ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA DE INGENIERÍA ELECTRÓNICA

"*ANÁLISIS DE LAS TÉCNICAS DEL RECONOCIMIENTO DE FIRMAS MANUSCRITAS DIGITALIZADAS PARA SEGURIDAD Y DISEÑO DE UN PROTOTIPO DE SOFTWARE*".

TESIS DE GRADO

PREVIA OBTENCIÓN DEL TÍTULO DE

INGENIERO EN ELECTRÓNICA Y COMPUTACIÓN.

PRESENTADO POR:

SEGUNDO ROSENDO CHAVEZ ARIAS.

RIOBAMBA - ECUADOR

2010

A Dios el guía espiritual y la fuerza en los momentos mas oportunos.

A mi familia, porque gracias a su apoyo, se logro alcanzar el trabajo establecido.

A mi tutor y colaborador ya que por medio de ellos pude resolver los problemas encontrados en la ejecución y desarrollo.

Y a todas las personas que de una u otra forma me ayudaron a conseguir este fin.

NOMBRE	FIRMA	FECHA
Dr. Ms.c. Romeo Rodríguez **DECANO DE LA FACULTAD** **DE INFORMÁTICA Y** **ELECTRÓNICA**	……	……
Ing. Paúl Romero **DIRECTOR DE LA** **ESCUELA DE INGENIERÍA** **ELECTRÓNICA.**	……	……
Ing. Wladimir Castro **DIRECTOR DE TESIS**	……	……
Ing. Paúl Romero **MIEMBRO DEL TRIBUNAL**	……	……
Lcdo. Carlos Rodríguez **DIRECTOR DPTO.** **DOCUMENTACION**	……	……
NOTA DE LA TESIS	……	

“Yo, **SEGUNDO ROSENDO CHAVEZ ARIAS**, soy el responsable de las ideas, doctrinas y resultados expuestos en esta Tesis de Grado, y el patrimonio intelectual de la misma pertenecen a la Escuela Superior Politécnica de Chimborazo”

Segundo Rosendo Chávez Arias

ÍNDICE GENERAL

CAPÍTULO I: FORMULACIÓN GENERAL DEL PROYECTO
Antecedentes, Justificación, Objetivos, e Hipótesis.

CAPÍTULO II: GENERALIDADES
Firmas Manuscritas, Lógica Difusa y Algoritmos.

CAPÍTULO III: ANALISIS DEL SISTEMA

CAPÍTULO IV: REQUISITOS, IMPLEMENTACION Y DESARROLLO

CAPÍTULO V COMPROBACIÓN DE LA HIPOTESIS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

CAPITULO I

FORMULACIÓN GENERAL DEL PROYECTO

1. ANTECEDENTES

Con la llegada de la era de los ordenadores personales y sobre todo de las tecnologías digitales, se plantearon una serie de nuevas necesidades que surgieron con la utilización de estos productos. Una de estas necesidades era la autenticación de los diferentes usuarios a través de estás tecnologías. Gracias a esta autenticación se convertía en relativamente fiable el uso de servicios a los usuarios destinados a facilitar los trámites a los mismos.

Existen multitud de mecanismos destinados a la verificación de los diferentes usuarios, como pueden ser tanto las técnicas basadas en firmas digitales o electrónicas, como el uso de otra serie de técnicas destinadas a la

verificación e identificación de firmas manuscritas digitalizadas. Aunque en la actualidad está muy extendido el uso de las firmas digitales, el uso conjunto

de técnicas de reconocimiento de firmas manuscritas digitalizadas puede aportar una serie de mejoras bastante considerables, tanto en la fiabilidad como en aspectos tan importantes para el usuario como la comodidad del usuario identificado.

1.2 JUSTIFICACION DEL PROYECTO DE TESIS O MEMORIA

Entre los sistemas de de identificación aportados por el DNI, uno de los más notables es el de la firma manuscrita, que aparece en el frontal de cada documento. Esta firma constituía hasta el momento un mecanismo por el cual se identificaba unívocamente al poseedor del DNI, esto es, en caso de que se necesitara comprobar la rúbrica de un firmante en un instante determinado, siempre se ha recurrido a comprobar de manera visual que dicha firma coincidía en gran medida con la que aparece en su Documento Nacional de Identidad.

Hoy en día, gracias a la aparición de las nuevas tecnologías digitales como Internet y el crecimiento de las posibilidades de los ciudadanos (por ejemplo la cantidad de trámites online a los que se tiene acceso como la declaración de la renta), se nos plantea la necesidad de que dichos ciudadanos cuando realizan cualquier operación importante a través de la red, no puedan falsificar su información personal. Por ejemplo se debe conseguir evitar que un determinado usuario consiga datos privados de otra persona (como datos fiscales o financieros) o simplemente que mediante una suplantación de datos, sea capaz de acceder a información que no está

autorizado a obtener.

Para intentar evitar esta serie de fraudes, surgen una serie de mecanismos que han ido evolucionando a lo largo de los años:

El primer intento se produjo con el uso de los conocidos usuario/contraseña, que consiste en que cada usuario tenga un identificador y una contraseña asociada, a través de los cuales acceda unívocamente a los servicios que desee realizar. Sin embargo este tipo de verificación no es nada fiable ya que dichos nombres de usuarios y contraseñas son relativamente fáciles de conseguir. Una primera mejora se intentó aconsejando a los usuarios que escogieran contraseñas seguras (es decir, difíciles de descifrar con un demonio informático), pero la única mejora que se conseguía con esto era retrasar un poco el tiempo de obtención del password.

Esto ocasionó que empezaran a aparecer mecanismos más evolucionados como la de la firma electrónica (o certificado digital) y las firmas manuscritas digitalizadas.

La firma electrónica consiguió evitar los problemas de desciframiento de contraseñas antes mencionados con el uso del ***algoritmo*** de encriptación

A grandes rasgos, en el certificado digital se envía la información de la persona encriptada con una ***clave pública***, lo que hace que sea visible para todos los organismos. Además la información que el usuario recibe viene también encriptada con dicha clave. La seguridad de este sistema consiste en que para

poder desencriptar dicha información se ha de estar en posesión de la ***clave privada***, que solo la posee el propietario del certificado digital. Esta clave es compleja de descifrar, ya que la encriptación del algoritmo es muy segura, siendo necesario para poder lograr dicho objetivo usar un sistema muy potente, además de emplear un tiempo tan elevado que motiva que cualquier tercera persona que pretenda conseguir información desestime esta opción.

Por otra parte y paralelamente se desarrolló un sistema basado en las firmas manuscritas, que consistía en obtener un mecanismo que permitiera a partir de la firma o rúbrica de la persona la información necesaria para asegurar si ésta se corresponde con la firma real de la misma o se trata de una falsificación. Esta rama se dividió en dos investigaciones fundamentales, la primera de ellas pretendía comprobar la veracidad a partir del trazo que iba realizando el usuario instantáneamente (hablamos de técnicas ***on-line***) mientras que la segunda buscaba no necesitar la presencia en tiempo real de la persona sino que se pudiese decidir a través de una firma realizada en cualquier instante de tiempo (en este caso se trata de las firmas ***off-line***).

En la actualidad existen implementados varios sistemas de reconocimiento de firmas manuscritas (tanto ***off-line***, como ***on-line***) que intentan comprobar la veracidad de un determinado par de firmas, es decir, verificar si ambas firmas corresponden a una misma persona. Dichos sistemas necesitan de un análisis automatizado de las imágenes que contienen las firmas.

El proceso de análisis de una imagen consiste en comprobar, identificar,

clasificar y comparar ciertos patrones característicos embebidos en la imagen que se está procesando. Como norma general los sistemas de reconocimiento de firma ***off-line*** siguen dos pasos fundamentales a la hora de realizar cualquier comprobación, dichos pasos descritos a grandes rasgos se pueden agrupar en *extracción de características* de la imagen y *comparación de las características* de las imágenes que se están verificando. Siguiendo este modelo pueden derivarse múltiples tipos de analizadores, dependiendo de los algoritmos que se usen para cada una de dichas partes. Por ejemplo supongamos que la extracción de características se hace a través del estudio de la concavidad de las curvas de la firma y la comparación a través de la distancia estadística de ambas, o simplemente que usemos el mismo sistema de comparación estadística y se escoja un sistema diferente de extracción de características.

Por ello, los reconocedores ***off-line*** de firmas manuscritas suelen estar compuesto de ese par de funciones:

- **Extracción de características**, usando para ello un algoritmo determinado se obtiene una serie de resultados a partir de la imagen tratada, que varían con cada tipo de firma que se esté estudiando.

- **Estudio comparativo**, de las características obtenidas en cada firma, comparando los resultados a través de un determinado algoritmo que variará según el reconocedor que se esté utilizando.

Dependiendo la fiabilidad de los algoritmos usados en cada una de las partes,

se obtendrá un resultado más o menos fiable, siempre dentro de la fiabilidad que proporciona el marco de las firmas manuscritas.

1.3 OBJETIVOS

1.3.1 OBJETIVOS GENERALES:

- Seguridad para el reconocimiento de firmas manuscritas digitalizadas por medio del diseño de un software.

1.3.2 OBJETIVOS ESPECIFICOS:

- Establecer las mejores opciones para el desarrollo de un software que brinde aplicaciones de seguridad.
- Realizar el control y comparación de firmas, con procesos en dos casillas la una para firma original almacenada y la otra para la nueva.
- Aplicación de nuevas tecnologías y recursos Informáticos para el desarrollo de este software.

1.4 HIPÓTESIS

Con el diseño e implementación de “Seguridad por medio de software para el reconocimiento de firmas manuscritas digitalizadas”, se pretende dar una solución flexible y económica para comprobar la autenticidad de un determinado usuario.

CAPITULO II

GENERALIDADES

Firmas Manuscritas, Lógica Difusa y Algoritmos.

2.1- Introducción a la Firma Electrónica

El objetivo de la firma electrónica es realizar un aporte al mundo de los documentos electrónicos con la misma función que aporta la firma manuscrita en un documento impreso, para identificar al autor del mismo.

Por medio de la firma electrónica se puede garantizar la identidad del firmante y que el documento no ha sido modificado después de ser firmado.

Para garantizar la identidad del firmante se emplea la tecnología de par de claves vinculada a los datos identificativos del titular del certificado.

De este modo, cuando se firma un documento se emplea un número único que sólo pertenece al firmante. El receptor del documento verifica la firma con la parte pública de la clave, de este modo, si el proceso de validación es positivo, debe concluirse que el firmante del documento es el titular del certificado.

La integridad del documento no se refiere al hecho de validar el contenido, sino de garantizar que el documento no ha sido modificado tras su firma. Para garantizar esto no es necesario que un tercero custodie una copia del documento sino que se realiza generando un código único del documento a partir de su estructura interna en el momento de ser firmado. Cualquier alteración del contenido del documento provocará que al aplicar de nuevo la función de generación de código único sea imposible reproducir el original, por tanto, quedará rota la integridad del contenido.

La firma electrónica avanzada demandaba la propiedad de no repudio, que jurídicamente implica que el firmante no pueda negar haber firmado. Entre otros, los elementos que garantizan el no repudio son los siguientes:

a. La **clave privada** vinculada al certificado y que confiere la veracidad a los documentos firmados sólo esté en posesión del firmante desde el mismo momento de genera dichas claves y vincula a sus datos.

b. El **certificado y los dispositivos de firma** empleados se deben basar en tecnologías y procesos seguros que eviten el uso o sustracción de la clave por parte de terceros.

c. El **certificado esté activo** en el momento de ser empleado.

d. Que los **receptores** de documentos firmados dispongan de un instrumento de verificación seguro que no permita suplantar identidades del firmante o de la Autoridad de Certificación que realiza la validación.

Inconveniente de la firma electrónica

Existe un inconveniente de la firma electrónica y reside principalmente en la necesidad de que el computador desde donde se haga cualquier

transacción que necesite de dicha firma deba contener el certificado de la persona que la va a llevar a cabo. Por tanto a la hora de que a dicho usuario se le olvide borrar dicho certificado del sistema, con lo cual cualquier persona que acceda al mismo podrá realizar operaciones suplantando sus datos. Además el certificado podría ser guardado para seguir realizando operaciones fraudulentas desde otro dispositivo y la única solución de la persona poseedora del certificado será solicitar uno nuevo.

Por lo que es poco probable de que el usuario recuerde si ha olvidado su certificado y más aún que compruebe si en realidad lo ha hecho, pues debería volver al mismo computador y puede que alguien haya borrado dicho archivo después de copiárselo para su posterior uso.

2.2.- Introducción a la Firma Manuscrita Digitalizada

El ser humano desde siempre ha tenido la preocupación de desarrollar sistemas de identificación personal, esto es, sistemas que nos digan que una persona es quien dice ser. La motivación ha sido muy variada y originada por circunstancias muy diferentes, tales como la autentificación de mensajes o de transacciones comerciales.

Entre los mencionados sistemas de identificación quizá el mas importante es el de la firma manuscrita, puesto que la identificación del autor del documento no necesita su presencia física y una vez firmado, no puede negar que prestó su consentimiento al contenido del documento.

Un problema en el reconocimiento de firmas manuscritas es la robustez del sistema ante falsificaciones. Las firmas falsificadas se clasifican en dos:

- Las ***cuidadosas***, en las que el falsificador es un experto y se le permite practicar con la firma antes de efectuar la falsificación.
- Las ***poco elaboradas*** en las que el falsificador no ha practicado demasiado con la firma original. Esta distinción se realiza porque el 95% de las falsificaciones realizadas para el uso fraudulento de tarjetas de crédito o cheque bancarios son falsificaciones de este último tipo.

Los sistemas de verificación de firmas se dividen generalmente en dos categorías: ***sistemas on-line*** y ***sistemas off-line***.

En los ***sistemas on-line*** se requiere la presencia física del firmante y utilizan características dinámicas para la verificación de la firma tales como la inclinación del bolígrafo, la presión ejercida en cada momento del trazo, la velocidad del trazo, la secuencia de direcciones tomadas por el trazo, etc. Para obtener este tipo de identificación en el momento de establecer la identidad del firmante se necesitan tableros digitalizadoras, bolígrafos electrónicos, etc.

Los ***sistemas off-line*** parten de la firma estampada en un documento, y tras escanear la imagen, se extraen características geométricas de la firma para verificar la identidad del firmante. Un problema de estos sistemas off-line es que se pierde la información dinámica o secuencial de la firma. Esta pérdida de información hace que estos sistemas, a pesar de no precisar la presencia física del firmante y requerir un equipamiento más económico, funcionen peor en cuanto a tasa de aciertos que los on-line.

2.3.- Descripción del Analizador de Firmas

Este proyecto se basa en la implementación de ***sistemas de reconocimiento de firmas manuscritas digitalizadas off-line***.

El software consta de 2 etapas o fases diferenciadas.

a) Extracción de características.

b) Aplicación de los diferentes algoritmos de comparación.

En cada una de estas partes existen varias subdivisiones.

1.4.- Lógica Difusa

La lógica difusa ha cobrado una gran relevancia por la variedad de áreas en las cuales es aplicada.

Estas van desde el control de complejos procesos industriales, hasta el

diseño de dispositivos artificiales de deducción automática, pasando por el procesamiento digital de señales así como también de sistemas de diagnostico.

La lógica difusa trata de crear aproximaciones matemáticas en la solución de ciertos tipos de problemas. Pretende producir resultados exactos a partir de datos imprecisos, por lo cual es particularmente útil en aplicaciones electrónicas o computacionales.

Las técnicas difusas han sido aplicadas en varias áreas del procesamiento de imágenes, como son filtrados, interpolación, detección de bordes y morfología, además de tener numerosas aplicaciones.

La lógica difusa maneja la incertidumbre presente en la estructura de un conjunto de datos. Los elementos de un conjunto difuso son pares ordenados que indican el valor del elemento y su grado de pertenencia.

Para un conjunto difuso, se tiene que el elemento x pertenece al conjunto A con un grado de pertenencia $\mu_A(x)$, donde $\mu_A(x) \in [0, 1]$. Por lo tanto, una variable puede ser caracterizada por diferentes valores lingüísticos, cada uno de los cuales representa un conjunto difuso.

Por ejemplo, la velocidad puede ser caracterizada por valores lingüísticos como Bajo, Medio y Alto, que representan una velocidad aproximadamente menor que 40 km/h, una velocidad cercana a 55 km/h y una velocidad sobre 70 km/h, respectivamente. Estos términos pueden ser conjuntos difusos con funciones de pertenencia como las mostradas en la figura II.1

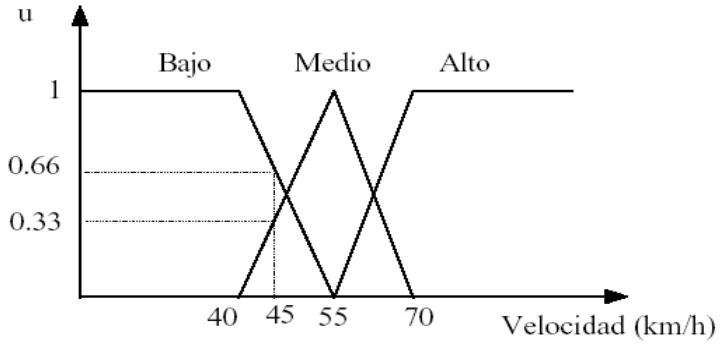


Figura II.1: Funciones de pertenencia para tres tipos de velocidad

Por lo tanto, si la velocidad es 45 km/h, existen grados de pertenencia 0.66, 0.33 y 0 a los conjuntos difusos Bajo, Medio y Alto respectivamente.

2.4.1 Modelos difusos lingüísticos

Estos modelos se basan en un conjunto de reglas heurísticas donde las variables lingüísticas de las entradas y salidas se representan por conjuntos difusos.

La figura II.2 muestra las principales componentes de un modelo difuso lingüístico: interfaz de fusificación, base de conocimiento, motor de inferencia e interfaz de defusificación.

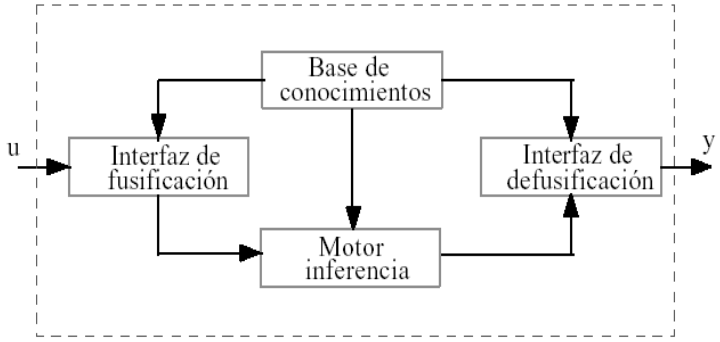


Figura II.2: Componentes de un modelo difuso

– Interfaz de fusificación. Este elemento transforma las variables de entrada del modelo (u) en variables difusas. Para esta interfaz se deben tener definidos los rangos de variación de las variables de entrada, así como los conjuntos difusos asociados con sus respectivas funciones de pertenencia.

2.4.2 Reglas Difusas Si....Entonces

Los conjuntos y los operadores difusos son los sujetos y predicados de la lógica difusa.

Las reglas:

Si....Entonces son usadas para formular las expresiones condicionales que abarca la lógica difusa:

Si x es A Entonces y es B

Donde A y B son los valores lingüísticos dados por los conjuntos definidos en los rangos de los universos de discurso llamados X e Y, respectivamente.

La parte Si de la regla x es A es llamada el antecedente o premisa, mientras la parte Entonces de la regla y es B es llamada la consecuencia o conclusión

2.4.3.- **Filtros Difusos para imágenes digitales**

Una de las técnicas de procesamiento mas utilizadas para mejorar la calidad de una imagen digital, es la aplicación de algoritmos que permitan eliminar las señales indeseadas, que por causa del método de adquisición aparecen en la imagen. Este tipo de técnica recibe el nombre de filtrado de imágenes, la cual es clasificada en dos grandes grupos, Filtrado Lineal y Filtrado No-Lineal.

En el procesamiento digital de imágenes, las técnicas que generan una matriz imagen al aplicarle un proceso lineal a la imagen de entrada son conocidas como técnicas lineales, las cuales realizan una combinación lineal de los píxeles pertenecientes a una pequeña vecindad del punto en estudio, con una matriz bidimensional que describe el proceso aplicar: esta técnicas es llamada convolución.

La matriz que describe el proceso lineal a aplicar, tiene como principal característica su tamaño, el cual depende exclusivamente de la aplicación que se quiera implantar. Dicha matriz recibe el nombre ventana o máscara. La principal ventaja de los filtros lineales es que llevan a cabo el proceso de inverso, es decir, hallar la imagen original a partir de la imagen filtrada, es inmediata usando por ejemplo la transformaciónde Fourier.

Los filtros no lineales son aquellos filtros cuyo funcionamiento no puede ser modelado por una convolución. Mejoran en algunos casos el filtrado lineal pero no siempre son la solución. Un filtro no lineal comúnmente usado es el de mediana, el cual sustituye cada valor de un píxel en la imagen, por la

mediana de los valores vecinos a él. La vecindad se determina usando una ventana que suele filtros difusos, los cuales involucran técnicas difusas para realizar el procesamiento de la imagen.

2.4.3.1. Filtro Difuso de reducción de Ruido

Las técnicas difusas para reducción de ruido en imágenes digitales, tratan principalmente con ruido impulsivo. Estos filtros difusos funcionan mejor que los filtros de esquema de rango (tales como los filtros de mediana), no obstante la mayoría de las técnicas difusas no están específicamente diseñadas para ruido gaussiano, o no producen resultados convincentes cuando son aplicadas para tratar este tipo de ruido. Para la construcción del filtro difuso se debe tener en cuenta su aspecto principal:

- El filtro estima una derivada difusa con el fin de ser menos sensible a variaciones locales debidas a estructuras de la imagen como son los bordes presentes en esta.

Para cada píxel que es procesado, se calcula en una primera etapa su derivada difusa, mientras que en la segunda etapa se elaboran 16 reglas, que determinan un término de corrección. Estas reglas hacen uso de la derivada difusa como entrada. Los conjuntos difusos son empleados para representar las variables lingüísticas: Pequeño, Positivo y Negativo.

Filtro Difuso

La idea general tras el filtro es la de promediar un píxel usando los valores de los píxeles del vecindario, pero simultáneamente cuidando las estructuras importantes de la imagen como son sus bordes.

Lo principal del filtro es distinguir entre las variaciones locales a causa del ruido y las debidas a los bordes presentes en la imagen. A fin de lograr esto, para cada píxeles se obtiene un valor que expresa el grado en el que la derivada en cierta dirección n es pequeña. Tal valor es obtenido para cada dirección correspondiente al vecindario del pixel procesado. La construcción del filtro están basada entonces en la observación de que una pequeña derivada difusa es causada por ruido, mientras que una gran derivada difusa

es causada probablemente por un borde en la imagen. En consecuencia, para cada direcciónse debe aplicar dos reglas difusas para tener en cuenta esta observación (y poder distinguir entre variaciones locales de ruido y estructuras de la imagen), y esto determina la contribución de los valores del vecindario de píxeles. El resultado de estas reglas es defusificado y se obtiene un tiempo de corrección para procesar el valor del pixel.

Estimación de la Derivada Difusa

Para filtrar debemos tener buenas indicaciones de los bordes y para hallar los bordes debemos filtrar.

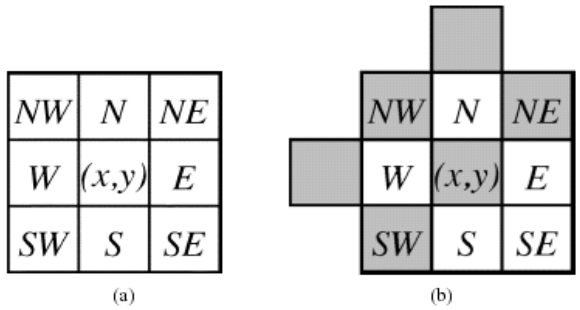


Figura II.3:(a) vecindario del píxel central (soy), (b) píxeles indicados en gris son usados para obtener la derivada difusa del píxel (x,y) en la dirección NW

2.4.3.2.- Detector de Bordes Difuso

Los bordes en una imagen se definen como sitios donde hay una variación significante en los niveles de gris en alguna dirección.La detección de bordes es la mas común aproximación para la detección de discontinuidades significativas en niveles de gris. Los algoritmos de detección de bordes localizan y acentúa en bordes. El principal propositode detectar bordes es la segmentación de escenas para la identificación de objetos en una imagen. Un segundo proposito es la conversión de una imagen en escala de grises a una en blanco y negro.

Existen diferentes estrategias para detectar bordes como lo son Sobel, el cual usa una mascara de 3x3 al igual que Prewit. También existen otros filtros como el Laplaciano del Gaussiano.

Los beneficios de emplear un clasificador difuso para la detección de bordes son que requieren un bajo, tienen baja sensibilidad al ruido, son isotrópicos, es decir que su comportamiento es el mismo sin importar en que direcciónse encuentre el borde, y de fácil modelado.

El clasificador difuso propuesto, opera sobre un conjunto de 4 características extraídas de un vecindario de 3x3 para cada pixel. El paradigma es simple, computacionalmente eficiente y es isotrópico. Cada pixel en la imagen es mapeado a blanco o negro. El detector de bordes basado en el clasificador difuso produce líneas blancas sobre un fondo negro.

2.4.3.3.- Metodología

Para el vecindario de 3x3 del píxel central p5, definimos cuatro direcciones: horizontal, vertical y dos diagonales. La diferencia de magnitudes de niveles de gris entre p5 y todos sus vecinos se designa por X1, X2, X3 y X4 y se calculan como sigue:

$$X1 = |p1 - p5| + |p9 - p5|$$
$$X2 = |p2 - p5| + |p8 - p5|$$
$$X3 = |p3 - p5| + |p7 - p5|$$
$$X4 = |p4 - p5| + |p6 - p5|$$

donde $|pi - pj|$ representa el valor absoluto de la diferencia de intensidades de gris. La figura I.4 muestra el vecindario de 3x3 y las direcciones mencionadas arriba. Se diferencian en este modelo cuatro clases de bordes y una clase no borde. Cada clase representa un par de bordes típicos, los cuales comparten el mismo vector de características c es 4-dimensional y asumido por X1, X2, X3 and X4. Los bordes representados por cada clase y su vector de características son mostrados en la figura I.5

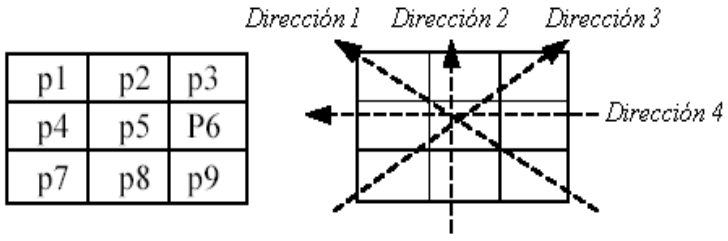


Figura II.4.- vecindario 3x3 y direcciones de bordes

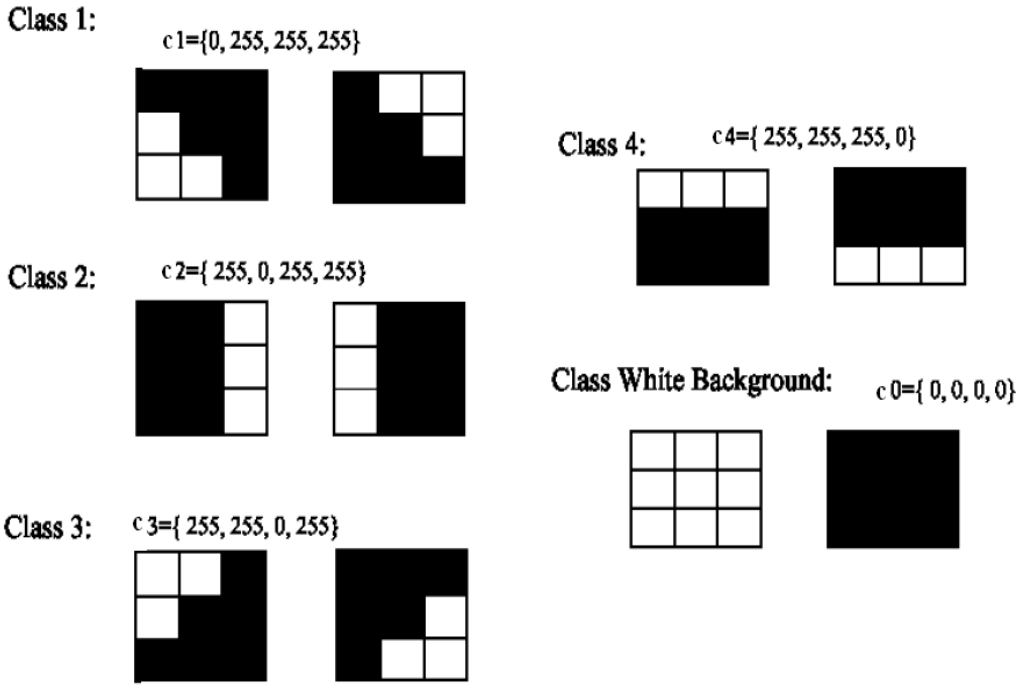


Figura II.5: Clases de bordes

Los vectores de características están dados de la siguiente forma:

Clase Blanca: c_0= **{** L, L, L, L**}**

Borde tipo 1: c_1=**{** L, H, H, H**}**

Borde tipo 2: c_2=**{** H, L, H, H**}**

Borde tipo 3: c_3=**{** H, H, L, H**}**

Borde tipo 4: c_4=**{** H, H, H, L**}**

Un clasificador difuso es un sistema que acepta vectores de características o verdades difusas para características pertenecientes a conjuntos difusos. La salida proporciona la clase a la cual pertenece ese vector de características o esa verdad difusa. Usualmente, los componentes de salida individua- les son verdades difusas y estas indican a que clase corresponde la salida.

La figura II.6 muestra el clasificador difuso empleado en el detector de bordes.

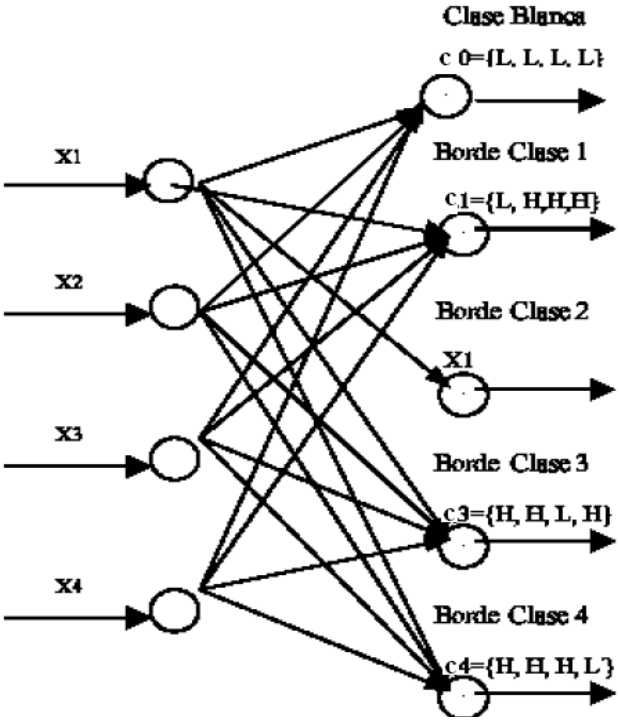


Figura II.6: Clasificador difuso competitivo

En las figuras II.7 y II. 8 se observa los respectivos diagramas de flujo.

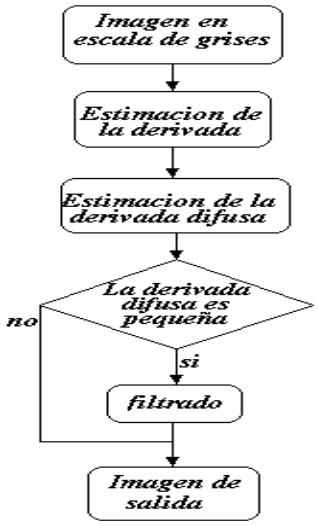


Figura II.7: Diagrama de flujo para el filtro difuso de reducción de ruido

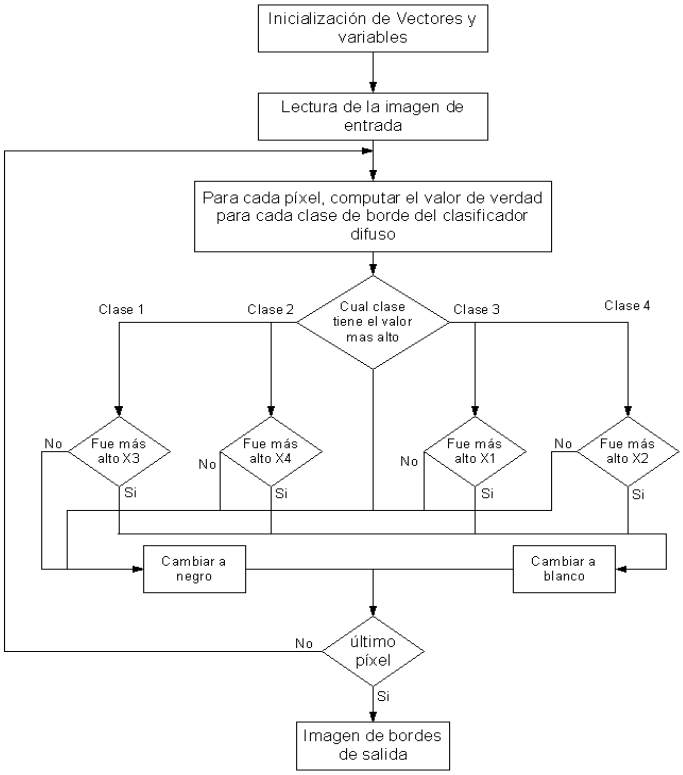


Figura II.8: Diagrama de flujo para el detector de bordes difuso

2.5.-Algoritmos de Tratamientos de Imágenes

2.5.1.- Algoritmo Sobel

El concepto fundamental para llevar a cabo el algoritmo de Sobel es la derivada, ya que los bordes representan realmente cambios de intensidad en una localización espacial determinada. Se pueden manejar dos órdenes de derivada: la primera y segunda derivada. En la Fig. 9, se observa con respecto al perfil de intensidad de línea horizontal y el cálculo de la primera derivada. El mismo comportamiento predecible aparece cuando se aplica la segunda derivada.

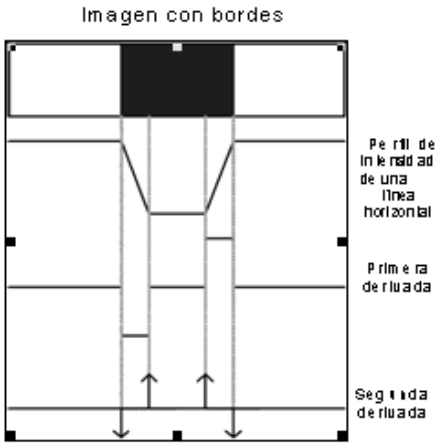


Figura II.9. Detección de bordes mediante operadores derivativos

El vector gradiente de una función f(x,y) indica la dirección y magnitud del cambio máximo en la imagen, y está dado por la primera derivada. Para nuestro caso, el cálculo del gradiente se realiza mediante máscaras de 3x3 y convolucionando a estas con la imagen a procesar. Las máscaras de Sobel (Fig. 10) se utilizan para realizar el cálculo del gradiente de Gx y Gy.

$$a)\begin{bmatrix} z_1 & z_2 & z_3 \\ z_4 & z_5 & z_6 \\ z_7 & z_8 & z_9 \end{bmatrix} \quad b)\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \quad c)\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Fig. II. 10 (a) Región de la imagen de 3x3, **(b)** Máscara usada para obtener Gx en el punto central de la región de 3x3, **(c)** Máscara usada para obtener Gy en el mismo punto

Tanto Gx como Gy, reciben el nombre de operadores Sobel. El requisito básico de un operador de derivación es que la suma de los coeficientes de la máscara sea nula, para que la derivada de una zona uniforme de la imagen sea cero.

La Fig. II.10, describe la implementación del pseudo código del algoritmo de Sobel.

Sea :
Mask = Sobel_X
Mask1 = Sobel_Y
m=1
Para cada i,j:
$\nabla x = 0; \quad \nabla y = 0;$
Para *p = - m:m*
Para *q = - m:m*
$\nabla x = \nabla x + image\_in(i+p,j+q) * Mask(p+m+1, q+m+1);$
$\nabla y = \nabla y + image\_in(i+p,j+q) * Mask1(p+m+1, q+m+1);$
fin
Fin
$Imangen\_out(x, y) = \sqrt{(\nabla_x(i, j))^2 + (\nabla_y(i, j))^2}$
fin

Fig. II .10. El Algoritmo Sobel implementa una convolución entre la imagen de entrada y la máscara Sobel Gx y Gy.

CAPITULO 3:

ANALISIS DEL SISTEMA

3.1.- Extracción de características de firmas

La extracción de características se realiza a través del algoritmo GSC[3] (Gradient, Structural, Concavity features). Dicho algoritmo obtiene una serie de datos gracias a la aplicación por separado de cada una de las anteriores características a través de un tratamiento directo de la imagen que contiene la firma manuscrita.

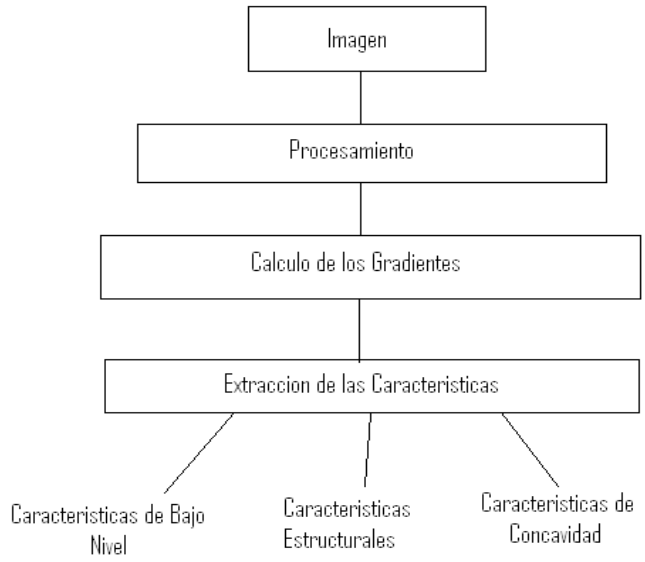


Figura III.1.- Proceso que sigue las imágenes que se van a comparar

Preprocesamiento de la imagen

Para realizar el preprocesamiento de la imagen para que posteriormente la misma siga un determinado estándar. Se normaliza la imagen con el fin de que todas las imágenes tratadas sean de una dimensión determinada.

Un aspecto importante es que el usuario puede contener una imagen a color de su firma (por ejemplo si el usuario firmó sobre un papel con un bolígrafo de color azul o rojo).

Se debe de pasar dicha imagen a escala de grises puesto que en primer lugar, toda firma al ser la representación escrita de un trazo, se realiza bajo un mismo color. Como el color en el que se realice la firma es irrelevante, Se asume que la imagen esta en escala de grises.

Para esto, la imagen real que se introduzca pueda estar en escala de grises, o en colores vivos para lo cual se realizara el cambio a escala de grises aunque justo antes de comenzar la extracción de las características, cada píxel de la imagen se trata como un píxel normal de una imagen en blanco y negro dependiendo si el valor de dicho píxel sobrepasa un umbral.

Figura III.2.- Muestra un ejemplo del paso a escala de negro de una imagen a color.

Como se observa las características quedan intactas en cuestión de los rasgos.

3.1.1.- Obtención del gradiente

Para poder conseguir las características de la imagen es necesario extraer una matriz de gradientes de cada una de ellas, el gradiente indica la dirección del píxel de la firma que se está tratando. Para el cálculo del gradiente se utiliza el operador de Sobel.

El **operador Sobel** aplicado sobre una imagen digital, calcula el gradiente de la intensidad de brillo de cada punto (píxel) dando la dirección del mayor incremento posible (de negro a blanco) además calcula el momento de cambio en esa dirección y devuelve un vector o una matriz. El resultado muestra la forma que cambia una imagen en cada punto analizado, y a su vez en cuanta medida un punto determinado representa un borde en la imagen además de la orientación a la que tiende ese borde.

Matemáticamente, el gradiente de una función de dos variables para cada punto es un vector bidimensional cuyos componentes están dados por las primeras derivadas de las direcciones verticales y horizontales. Para cada punto de la **imagen**, el **gradiente** del vector apunta en dirección del incremento máximo posible de intensidad, y la magnitud del **gradiente** del vector corresponde a la cantidad de cambio de intensidad de esa dirección.

Figura III.3.- Muestra una imagen en escala de grises junto con su imagen de

gradiente asociada.

El gradiente mide la magnitud y dirección de los mayores cambios de intensidad en un bloque de 8 de un píxel dado. El gradiente es calculado mediante el operador de Sobel, concretamente usando conjuntamente el operador Sobel vertical y el horizontal. Dichos operadores consisten en matrices de tamaño 3x3 que son aplicadas a cada zona de la imagen.

-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

horizontal (Gx) y vertical (Gy).

Figura III.4.- Valores de las matrices de Sobel

A partir de la imagen dada, de tamaño D_1 x D_2, cada píxel de la vecindad de 8 vecinos es computado con las casillas Gx y Gy para determinar los componentes X e Y de S_X y S_Y respectivamente.

Estos componentes se calculan matemáticamente mediante las siguientes expresiones a partir de la imagen de entrada *I* ():

$$S_x(i,j) = I(i-1,j+1) + 2I(i,j+1) + I(i+1,j+1) - I(i-1,j-1) - 2I(i,j-1) - I(i+1,j-1)$$

$$S_y(i,j) = I(i-1,j-1) + 2I(i-1,j) + I(i-1,j+1) - I(i+1,j-1) - 2I(i+1,j) - I(i+1,j+1)$$

Los índices ***i***, ***j*** determinan el número de fila y columna respectivamente de un determinado píxel de la imagen.

La magnitud del gradiente y la dirección del mismo se calculan a partir del

operador de Sobel. Así se obtiene el valor de cada píxel ***S_X (i, j)*** y ***S_Y (i, j)*** de la imagen, la obtención de la magnitud del gradiente se lleva a cabo mediante la expresión matemática:

$$r(i,j) = \sqrt{s_x^2(i,j) + s_y^2(i,j)}$$

Por su parte el cálculo de la dirección del gradiente del píxel se lleva a cabo aplicando la siguiente expresión:

$$\theta(i,j) = tg^{-1}\frac{S_y(i,j)}{S_x(i,j)}$$

La dirección del gradiente se calcula como arco tangente del operador Sobel vertical de la vecindad dividido por el operador Sobel horizontal de dicha vecindad. De estas dos expresiones, la más importante es la segunda ya que a partir de la dirección del gradiente se podrá obtener la mayoría de las características importantes de la imagen que será de gran utilidad.

El mapa de gradiente entrega información, relevante es este hecho, que llegados a este punto se esta en condiciones de comenzar con la extracción de características de la imagen.

Se puede hacer una idea de la información que contiene el mapa de direcciones del gradiente observando la siguiente secuencia en la Figura III.5

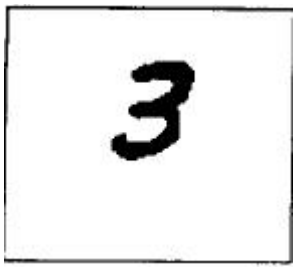
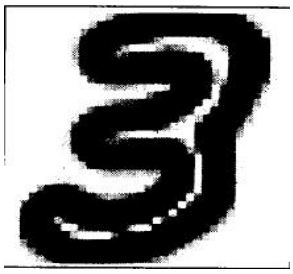


Imagen Original



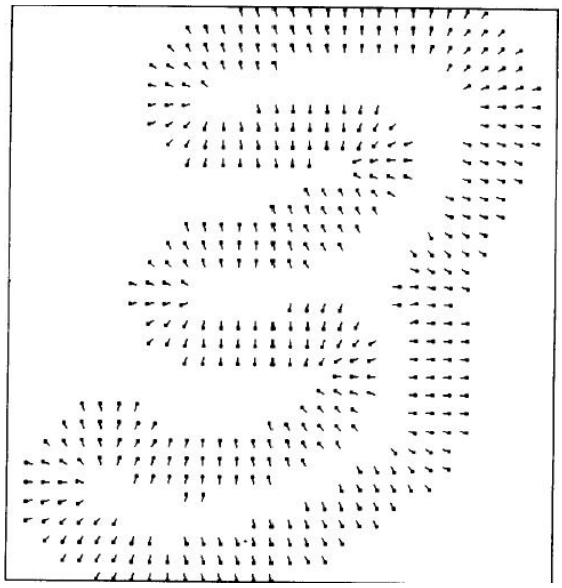


Imagen de la magnitud del gradiente Imagen de direcciones del gradiente

Figura III.5.- Características de Bajo nivel de un Gradiente.

El conjunto de características de bajo nivel posee características. La primera consiste en que particiona el mapa del gradiente en bloques, y la segunda realiza una cuantización de las direcciones de los gradientes obtenidos. La idea reside en que se traten las regiones de la imagen por separado para garantizar que los datos que se obtienen del análisis estén localizados en una misma región y se evite que datos idénticos de diferentes zonas de la imagen puedan llegar a verificar que dos firmas pertenecen a una firma real de una persona sin que esto sea cierto.

Una vez realizado el seccionamiento del mapa de direcciones de gradiente, se extrae las características locales de cada región, con lo que el resultado más fiable de los datos obtenidos para la posterior comparación de las rúbricas.

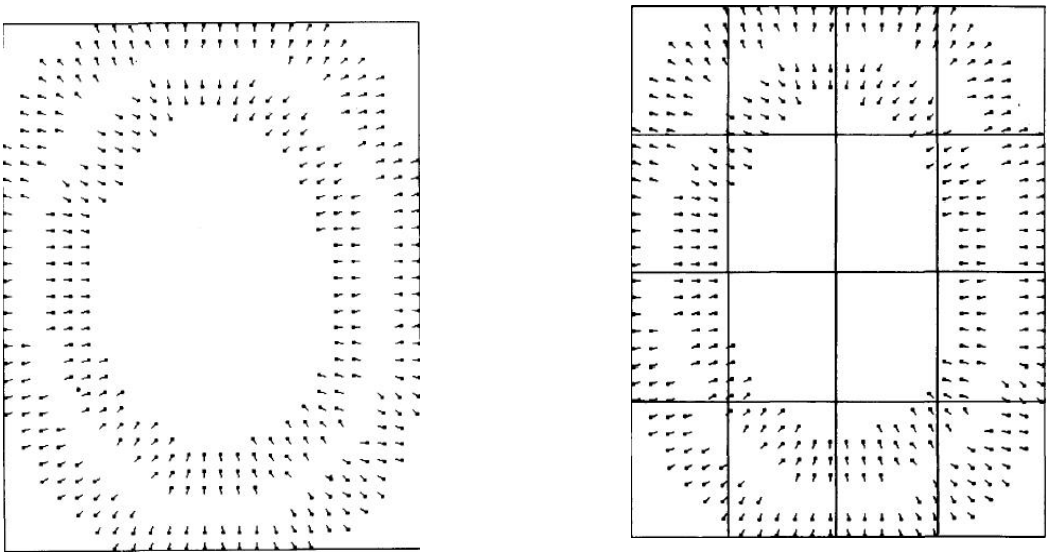


Figura III.6.- Particionamiento de una imagen. En un panel de 4x4

Partición del Mapa del Gradiente en regiones.

La partición del mapa del gradiente se lleva a cabo para encontrar variaciones locales de contorno. Tal como se observa en la figura 6, la partición del mapa de direcciones del gradiente, aísla en este ejemplo las zonas donde las direcciones son diagonales, las cuales se encuentran cada una en una zona o región de las que se ha particionado la imagen.

Cuantización de las direcciones del gradiente.

La cuantización de los gradientes en la imagen se debe tener en cuenta para la extracción de características de la imagen. Dicha cuantización se realizará en un número K de rangos de modo que el conjunto total de rangos en los que se mueva el gradiente sean 360. Por tanto, suponiendo un número de rangos k=12, los posibles rangos de direcciones serían 0º - 30º, 30º - 60º, 60º-90º… 330º - 360º. Se puede observar que después de la cuantización del gradiente, las nuevas direcciones corresponderán a enteros en el rango [1, 12]. La dirección del gradiente viene dada por la expresión

$$\theta(i,j) = tg^{-1}\frac{S_y(i,j)}{S_x(i,j)} \quad .$$

y la función arco tangente devuelve un valor del rango [-90, 90], no se podría conseguir llenar el vector con rangos mayores que 90. Sin embargo como matemáticamente se cumple que:

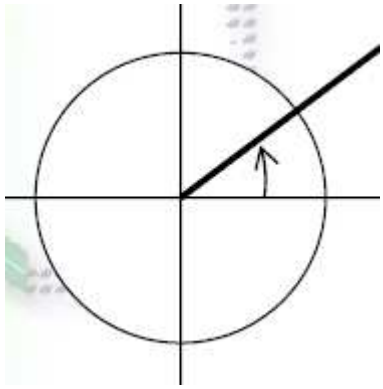
$$tg(\theta) = \frac{S_y(i,j)}{S_x(i,j)}$$

Por tanto, conociendo el signo de $S_y(i,j)$ y de $S_x(i,j)$ podemos ver si el angulo pertenece al primer, segundo, tercer o cuarto cuadrante.

Así pues según las circunstancias se tendrá que calcular el ángulo de una forma determinada:

a) Si $S_y(i,j) > 0$ y $S_x(i,j) > 0 \rightarrow$ Estamos ante un angulo del primer cuadrante, con lo que no seria necesario con ningún tratamiento ya que el angulo resultante de la arcotangente seria un angulo real, grafica del ejercicio.

$$\theta(i,j) = tg^{-1}\frac{S_y(i,j)}{S_x(i,j)}$$



Grafica III.7 : Grafica del la formula

b) En el caso de que $S_x(i,j) < 0 \rightarrow$ como el coseno es negativo, significa que en su representación se eextiende hacia el segundo o tercer cuadrante de la circunferencia dependiendo, de si el angulo obtenido es mayor o menor que cero la solución es por tanto la siguiente:

$$\theta(i,j) = tg^{-1}\frac{S_y(i,j)}{S_x(i,j)} + 180$$

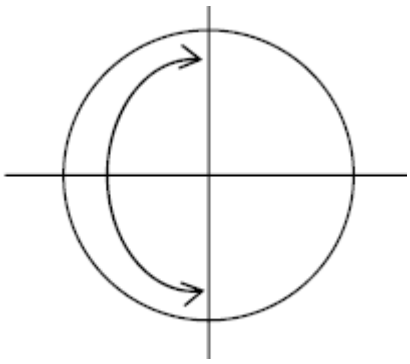


Figura III. 8 : Grafica de la Ecuacion.

c) Si concretamente tenemos el caso de que $S_y(i,j) < 0$ y $S_x(i,j) > 0 \rightarrow$ Estamos en el caso de que el seno es negativo, y también el coseno, por lo tanto estamos claramente ante un angulo perteneciente al cuarto cuadrante, ya que el angulo obtenido es negativo, para pasarlo a un angulo real del cuarto cuadrante se aplicara la siguiente formula.

$$\theta(i,j) = tg^{-1}\frac{S_y(i,j)}{S_x(i,j)} + 360$$

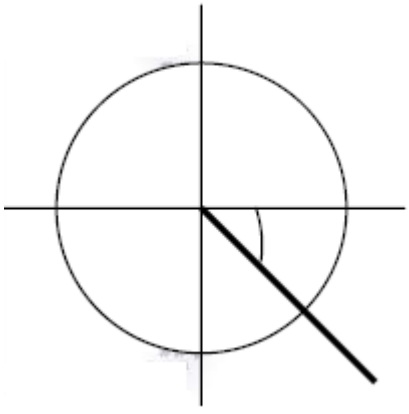


Figura III.9: Figura refernte a la formula

Características direccionales binarias.

Indican la presencia o ausencia de píxeles en un rango de direcciones del gradiente en cada partición. Las direcciones del gradiente de todos los píxeles se cuantizan en los K rangos. Exactamente K características son calculadas en cada bloque o sección de la imagen. Por lo tanto un total de $N_1 \times N_2 \times K$ características son computadas sobre la imagen completa.

Las características direccionales binarias son calculadas para cada partición a través de un índice incremental sobre el número de rangos K. Es decir para cada sección de la imagen tendemos un vector asociado de K posiciones en el que $F_i(k)$ se incrementará en 1 por cada píxel del cuadrante cuya dirección del gradiente esté dentro del rango de direcciones [(360/K)*k, (360/K)*(k+1)].

Características de valores reales direccionales.

Se basa en el porcentaje de píxeles en cada rango de direcciones de cada sección de la imagen. Una vez extraído el conjunto de características anteriores, el cálculo de estas nuevas es automático puesto que lo único que se necesita son los mismos vectores que antes con el mismo número de posiciones, con la diferencia de que en cada posición de este nuevo vector, se encuentra representado el porcentaje de píxeles que tiene la dirección del gradiente. Con lo cual el vector que se obtiene puede aprovecharse del anterior cambiando los valores de las posiciones.

Características Estructurales

Representan la presencia o ausencia de elementos L estructurales en el contorno de cada partición, es decir, estas características representan arcos y líneas de diferentes orientaciones.

Este tipo de características fijan su valor a través de características de L

vecindad de un píxel. En este caso usamos una vecindad de L = 8, es decir, los 8 vecinos del píxel actual. La figura 17 muestra la relación de rangos que puede tomar cada gradiente de cada píxel suponiendo que K = 12.

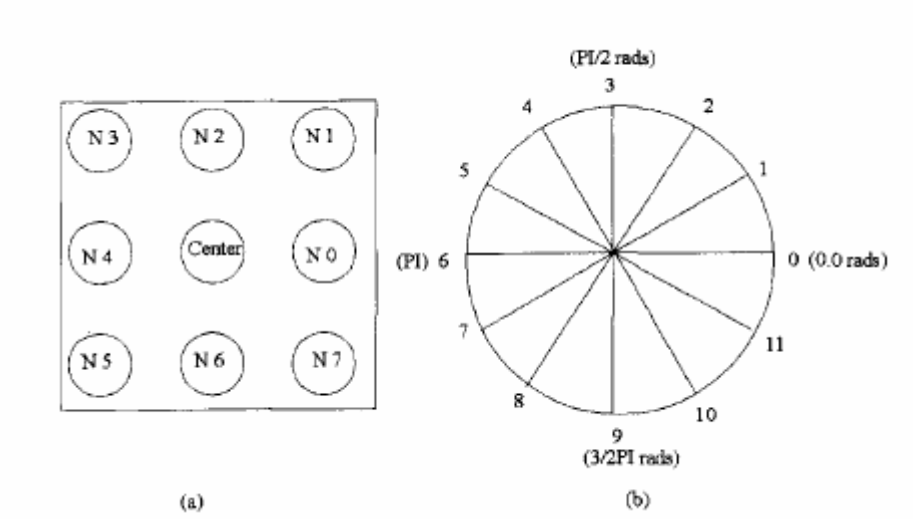


Figura III.10 a) muestra una 8 vecindad de un píxel central (Center)
Figura III. 11b) muestra los rangos en los que puede encontrarse el gradiente de un píxel

Rule	Description	Neighbor 1 (Range)	Neighbor 2 (Range)
1	Horizontal line, type 1	*N* 0 (2,3,4)	*N* 4 (2,3,4)
2	Horizontal line, type 2	*N* 0 (8,9,10)	*N* 4 (8,9,10)
3	Vertical line, type 1	*N* 2 (5,6,7)	*N* 6 (5,6,7)
4	Vertical line, type 2	*N* 2 (1,0,11)	*N* 6 (1,0,11)
5	Diagonal rising, type 1	*N* 5 (4,5,6)	*N* 1 (4,5,6)
6	Diagonal rising, type 2	*N* 5 (10,11,0)	*N* 1 (10,11,0)
7	Diagonal falling, type 1	*N* 3 (1,2,3)	*N* 7 (1,2,3)
8	Diagonal falling, type 2	*N* 3 (7,8,9)	*N* 7 (7,8,9)
9	Corner 1	*N* 2 (5,6,7)	*N* 0 (8,9,10)
10	Corner 2	*N* 6 (5,6,7)	*N* 0 (2,3,4)
11	Corner 3	*N* 4 (8,9,10)	*N* 2 (1,0,11)
12	Corner 4	*N* 6 (1,0,11)	*N* 4 (2,3,4)

Figura III.12 .- Muestra el cuadro de características estructurales que se pueden obtener a través de la firma.

Si se observa detenidamente, se conseguirá un conjunto de **doce**

características que se corresponden con los tipos de línea que pueden tener los vecinos del píxel considerado (tomando tipos de líneas horizontales, verticales, diagonales y esquinas).

El conjunto de datos obtenidos por tanto se corresponderá con un vector de doce posiciones por cada bloque o sección de la imagen (cada una referente a un tipo de característicaestructural) que almacenará el porcentaje de píxeles cuyos vecinos cumplen dicha característica estructural.

Características de Concavidad

Comprenden el conjunto de características del ***Algoritmo GSC***. Dicho conjunto puede ser seccionado en tres subclases de características que son: densidad del bloque, de trazos grandes y de concavidad. Se llegará a conseguir un vector que de datos que tendrá un tamaño de $N_1 \times N_2 \times 8$ bits, siendo:

N_1 el número de particiones horizontales en las que se divide la imagen que contiene la firma manuscrita.

N_2 el número de particiones verticales en las que se divide dicha imagen.

Coarse Píxel Density Features (características de densidad de píxeles).

Se engloba el agrupamiento general de píxeles de la imagen, se realiza un promedio general del número de píxeles negros que contiene cada bloque de la firma. Con esto se consigue aislar el porcentaje de firma que cae dentro de cada bloque de la imagen, lo que intuitivamente puede ser de gran utilidad a la hora de determinar cuál es el bloque de la imagen que tiene más peso cuando se realiza la comparación de dos firmas.

Por tanto determinaremos que en este apartado el conjunto de números de valores comprendidos entre 0 y 100 (que representan este tipo de características) será de tamaño N_1 x N_2.

Large Stroke Features (características de trazos largos).

Las características que se agrupa dentro de esta sección nos indica las líneas de la firma que conforman trazos largos horizontales y verticales. Se obtienen directamente a partir de las características estructurales. La presencia de un trazo largo es determinada testeando la cantidad de píxeles de cada bloque que forman líneas horizontales o verticales, es decir, el porcentaje de ellos que cumpla las características estructurales 1 ó 2 ó en su defecto 3 ó 4.

Por lo tanto se conseguirá un vector de enteros de tamaño $\mathbf{N_1}$ x $\mathbf{N_2}$ x **2**, cuyos valores estarán comprendidos en un rango de 0 a 100.

U/D/L/R/H Concavity Features (características de concavidad).

Las características de concavidad se calculan envolviendo la imagen en una estructura de estrella, es decir, para cada píxel considerando las características estructurales diagonales y de esquinas.

Con esto se consigue mostrar las líneas en ocho direcciones y determinar a través de la estructura de estrella cual de las líneas toca a un determinado píxel. Un rayo puede tocar un píxel de la imagen o un borde de la misma.

Una vez hecha la matriz de gradientes de la imagen, se extrae las características de la imagen. Estas son: Las de bajo nivel, las estructurales y las de concavidad.

3.1.2.- Características de Bajo Nivel

Las características de bajo nivel consiguen medir las variaciones de los gradientes de la imagen, para cada imagen a tratar, la extracción de este tipo de características indica el porcentaje de píxeles cuyo gradiente se encuentra dentro de un rango de direcciones determinado. Para conocer hasta dónde se extienden los píxeles más claros de la imagen partiendo de los píxeles de los bordes de la misma.

3.1.3.- Características Estructurales

Se comprueban características de vecindad de los píxeles de la imagen. El objetivo de este análisis es comprobar para cada píxel si pertenece a una línea horizontal, vertical, diagonal… o si se encuentra haciendo esquina en el trazo de la firma. Observando este tipo de características se agruparán los píxeles de la imagen que pertenecen a un tipo de línea determinada.

3.1.4.- Características de Concavidad

Esta característica averigua sobre la concavidad de la imagen, se dividen en:

***Características de densidad de píxeles (Coarse Pixel Density Features)*:**

Captura los grupos de píxeles existentes en la imagen, o el número de píxeles negros que existe en una zona determinada.

● *Características de trozos grandes (Large-Stroke Features)*: Se captura los trazos largos horizontales y verticales de los que está compuesta la imagen que se está tratando.

● *Características de concavidad (Concavity Features)*: Engloba a los píxeles de la imagen que pertenecen a una diagonal de un trazo de la misma o que forma parte de una de las esquinas que forma su dibujo.

3.2.- Análisis y de evaluación de similitudes de las firmas

Una vez extraídas las características que determinan las imágenes, se va a evaluar la similitud de firmas queconsiste en la aplicación de uno o varios procedimientos a través de los cuales podamos observar la variación que se produce entre los vectores de características de las firmas.

3.2.1.- Distancia Euclídea

Se denomina ***distancia Euclídea*** entre dos puntos $A(x_1, y_1)$ y $B(x_2, y_2)$ a la longitud del segmento de recta que tiene por extremos *A* y *B*.

Se expresa matemáticamente como:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Para el estudio realizado es, la distancia euclídea entre los dos vectores de características se corresponde únicamente con la raíz cuadrada de la diferencia de las posiciones de dichos vectores al cuadrado, es decir:

$$d = \sqrt{(f_1(x) - f_2(x))^2 + (f_1(x+1) - f_2(x+1))^2 + \ldots + (f_1(x+n) - f_2(x+n))^2}$$

Ventajas de este método.

Se trata de un método relativamente sencillo de calcular, ya que no requiere cálculos complejos de valores, por lo tanto entre las ventajas de la distancia euclídea se destaca:

- Es un método relativamente fácil de implementar.
- El tiempo de ejecución es relativamente rápido ya que no tiene que realizar cálculos complejos y constantes, ya que no varía con los vectores de características evaluados.

Inconvenientes de este método.

Al igual que las ventajas son la mayoría derivadas de la sencillez del método, esto mismo supone que aparezcan inconvenientes como son:

Todas características tienen el mismo peso, es decir, no existen características más determinantes como por ejemplo que el número de píxeles del mismo cuadrante de las dos firmas varíen de una forma considerable.

3.2.2.- Distancia estadística

El método de la distancia estadística se aplica sobre un vector de características GSC binario por lo que resulta relativamente adaptar el vector de porcentajes a esta técnica.

La única modificación que se realiza es considerar que la propiedad está activa en el vector en el caso de que el porcentaje de la característica de la i-ésima posición sea mayor que el 50%.

Esta técnica mide la similitud entre dos vectores binarios de imágenes usando la Correlación.

Se toma $S_{i,j}(i,j \in \{0,1\})$ como el número de posiciones del vector de características cuyo valor en el primer vector de características corresponde con *i* y su valor en el segundo vector de características coincide con el valor de *j*. Dado los dos vectores de características $X \in \wedge$ e $Y \in \wedge$, la similitud las imágenes obtenida a través de sus vectores de características depende de cuatro valores:

- s_{00}, el número de veces que la posición $X[indice]$ y la posición $Y[indice]$ de los vectores de características coincide y es igual a cero.
- s_{01}, el número de ocurrencias de los vectores de características tal que $X[indice]=0$ e $Y[indice]=1$.
- s_{10}, el número de ocurrencias de los vectores de características tal que $X[indice]=1$ e $Y[indice]=0$.
- s_{11}, el número de veces que la posición $X[indice]$ y la posición $Y[indice]$ de los vectores de características coincide y es igual a uno.

Se define la distancia estadística de X a Y $S(X,Y)$ como la medida de la correlación que se define en la siguiente expresión:

$$S(X,Y) = \left(\quad \right) \quad \backslash\backslash s_{10} + s_{11})(s_{01} + s_{00})(s_{11} + s_{01})(s_{00} + s_{10}))^{\cdot\cdot}$$

Ventajas de este método.

- Se trata de un método general, cuyo tiempo de cálculo es relativamente rápido.
- Se aproxima mucho a la solución y es sensible a los cambios en la firma.
- No necesita que las firmas tengan la misma localización dentro de donde se dibuja.

Desventajas de este método.

- Al igual que la mayoría de los métodos Off-line, aún tienen muchas deficiencias, aunque este se aproxima bastante a obtener una buena solución.

3.2.3.- Similitud por bloque

Este método es más específico que los dos anteriores, la idea consiste en evaluar cada sub-bloque de la imagen por separado comprobando la distancia euclídea que se deriva de cada uno.

Ventajas de este método.

- Es el más rápido de los tres métodos ya que no tiene que evaluar todos los bloques salvo en el caso de que se llegue a una solución satisfactoria

Inconvenientes de este método.

- No permite que las firmas esté impresas en diferentes lugares donde se representa o devolverá un resultado negativo.
- Es un método demasiado restrictivo.

3.3.- Requisitos Funcionales

Definen los servicios que debe ofrecer el sistema a los usuarios del mismo para alcanzar los objetivos. También almacenan las características de lo que no debe hacer el sistema.

Almacenar base de conocimientos

El sistema deberá *almacenar en una base de datos las firmas originales manuscritas de personas, con el fin de tener un referente a la hora de analizar la veracidad de una firma dada*

Analizar la veracidad de una firma manuscrita

El sistema deberá *permitir la evaluación de una firma manuscrita con objeto de verificar si dicha firma corresponde a una firma original de la persona o por el contrario se trata de una falsificación.*

La evaluación de dos firmas manuscritas debe poder permitir muchas posibilidades, entre ellas que la firma original se encuentre almacenada en una base de conocimientos, que las dos firmas procedan de imágenes a comparar o que alguna de las dos se obtenga inmediatamente a través de un firmado dinámico

Aplicación didáctica y clara

El sistema deberá ser una aplicación didáctica y poseer una interfaz clara con objeto de que el usuario sepa en cada momento como interactuar con él

Tratamiento de imágenes

El sistema deberá ser capaz de adaptar las imágenes para poder trabajar con ellas, por ello debe proporcionar al usuario herramientas que le permitan adaptar una imagen al tipo de imagen con el que trabaja la aplicación.

Con este requisito funcional se refiere a que el sistema trabaje con imágenes en escala de grises y de un tamaño máximo determinado, si el usuario posee imágenes a color o grandes, a través de herramientas adicionales, pueda adaptar su imagen a un tipo de imagen válida.

3.3.1.-Requisitos No Funcionales

Los requisitos no funcionales son condiciones que se le imponen al sistema a desarrollar relacionadas con aspectos principalmente de calidad: usabilidad, rendimiento, disponibilidad, fiabilidad, seguridad, compatibilidad con hardware o software, etc.

Este tipo de requisitos no aportan funcionalidad al software sino que su aportación particular es esencialmente aparte de la calidad ya mencionada, aspectos de eficiencia y claridad.

Los requisitos no funcionales que se han seguido en la elaboración de este proyecto son los que se describen a continuación:

Bajo costo en operaciones de ejecución y rapidez de respuesta

El sistema presenta un costo computacional bajo ya que no es necesario hacer varios recorridos por todos los píxeles de la imagen, lo que supondría un número de cálculos relativamente elevado. Para ello se aprovechara cada recorrido para realizar cálculos relativos a varias pasadas, con el fin de que el tiempo de ejecución sea lo más rápido posible.

Modularidad

El sistema presenta un diseño modular para permitir las posteriores

evoluciones del software incorporándole nueva funcionalidad para poder añadir nuevas características al software existente

Posibilidad de ampliación

El sistema es fácilmente ampliable, gracias a la modularidad procurada en la implementación del mismo. La ampliación resulta sencilla, únicamente añadiendo nuevas llamadas en el módulo principal a los nuevos módulos realizados

Integridad

El sistema presenta una alta integridad y facilidad de instalación. A través de un único programa de instalación (un ejecutable o .exe) se podrá hacer uso de todas las funciones del software.

Código fuente de fácil comprensión

El sistema posee un código fuente fácilmente comprensible.

3.3.2.-Otros Requisitos

Entre los requisitos mínimos que se nenecita para el correcto funcionamiento del programa en lo referente a Software y Hardware se encuentran los siguientes:

Software

- El Programa funciona correctamente bajo el sistema operativo Windows XP
- Se debe tener instalado el paquete de Windows Microsoft .NET Framework 1.1 para su correcto funcionamiento ya que se utilizan librerías que posee este paquete
- El Paquete de programación Visual Studio.net, herramienta con la cual se realiza los cálculos, el almacenamiento, y la interfaz grafica para los usuarios.

- Se debe tener instalado el Programa UC-Logic que sirve para la utilización de un lápiz electrónico medio por el cual se realizaran las rubricas a ser almacenadas y de comprobación.

Hardware

- Computador Pentium 4

  Mainboard intel 945gcnl
  Procesador 2.0 ghz
  ram 512 Mb
  disco duro 80 gb
  dvd writer
  lector de tarjetas múltiples
  fax
  monitor 15"
  mouse
  teclado
  parlantes
  pad mouse
  regulador

- Tablero Digital con su respectivo lápiz electrónico

CAPITULO 4:

IMPLEMENTACION Y DESARROLLO

La aplicación está compuesta por una serie de formularios (o ventanas) que son la parte gráfica de la misma, que interactúan con clases destinadas al cálculo del código. Esto es porque se ha seguido un diseño modular que facilita los posibles cambios futuros con el fin de que se pueda añadir funcionalidad.

Se opto por realizar una arquitectura dividida en capas de programación, siendo lo más apropiado un diseño en tres capas. Con esto conseguimos una serie de beneficios como son:

- La división en componentes reduce la complejidad, permite la reutilización y acelera el proceso de ensamblaje de software.
- Permite la incorporación de componentes de distintos fabricantes,
- Reducción de los costos y facilitando la construcción de estándares.
- El software se hace cada vez más rápido, de mejor calidad y a menor costo.
- Los costos de mantenimiento del software se reducen.

Esencialmente la arquitectura implementada consta de las siguientes capas:

Capa de datos. Es la encargada de almacenar físicamente los datos necesarios para la aplicación. En este nivel se encuentran las instrucciones de creación, modificación, acceso, y borrado de las bases de datos y/o archivos físicos.

Capa de negocios. Contiene la lógica que realiza el trabajo de la aplicación sin el acceso a la base de datos. Es una capa que sirve de comunicación entre la capa de datos y la de presentación, aunque las demás funciones de tratamiento interno de la aplicación también se desarrollan en este nivel.

Capa de presentación. Es la que se encarga de contener la interfaz gráfica de la aplicación. En ella se desarrolla la lógica de funciones cuya finalidad es presentar el entorno de trabajo al usuario.

En los siguientes apartados se detallaran las funciones más importantes englobadas dentro de la capa de lógica de negocio a la que pertenecen.

4.1.-Capa de datos

Se describe las clases del software cuya misión irá destinada al acceso físico a los datos. En la aplicación se cuenta con una base de datos de personas en las que se almacenarán para cada individuo requerido sus datos personales, junto con su firma si el usuario lo desease. Por ello es importante contar con una capa de datos que nos proporcionen las funciones de acceso a dicha base de datos.

Clase OleDbHelper.cs

Es la clase que se encarga de implementar los métodos de la capa de datos, y crea sentencias de acceso a una base de datos específica. Aquí es donde se comprende mejor la separación en capas, ya que si por casualidad se quiere realizar un cambio de base de datos y trabajar por ejemplo con MySQl o Postgress, en lugar de Access, sólo sería

necesario modificar esta clase ya que los métodos de acceso a la base de datos de la capa de negocio, únicamente llamarían al método correspondiente de OleDbHelper.

Sentencia ExecuteNonQuery

Tipo del resultado Integer : Representa el número de filas afectadas por la ejecución este método

Descripción: Ejecuta una sentencia de acceso a la base de datos (es decir, una query). Por ejemplo una inserción de datos en la base de datos.

Parámetros: connectionString : String -- *(una cadena de conexión válida para una conexión OleDbConnection, es decir, el proveedor de servicios de la base de datos, junto con la ruta de la misma)*

commandType : CommandType -- *(tipo de comando, que puede ser: stored procedure, text, etc.)*

commandText : String -- *(el nombre del procedimiento almacenado o la sentencia SQL a ejecutar en la base de datos)*

commandParameters : OleDbParameter[] -- *(un array de*

OleDbParameters usados para ejecutar la sentencia SQL)

Sentencia ExecuteDataset

Tipo del resultado DataSet: Conjunto de datos que contiene la colección de elementos generados por el comando

Descripción: Ejecuta una sentencia de consulta de la base de datos. Por ejemplo una sentencia de obtención de datos.

Parámetros: connectionString: String -- *(una cadena válida de conexión al objeto OleDbConeccion que conecta con la base de datos)*

commandType: CommandType -- *(tipo de comando, que puede ser: stored procedure, text, etc.)*

commandText: String -- *(el nombre del procedimiento almacenado o la sentencia SQL a ejecutar en la base de datos)*

commandParameters: OleDbParameter[] -- *(un array de OleDbParameters usados para ejecutar la sentencia SQL)*

4.2. -Capa de negocio

Esta capa es la encargada tanto de la implementación de los métodos que comunican la base de datos con la capa de presentación como de funciones de tratamiento interno de la aplicación. Por lo tanto se trata de la clase más importante que tiene que desarrollar ya que es la que va a realizar las funciones de tratamiento de las firmas manuscritas .

Clase Constantes.cs

Esta clase únicamente almacena las constantes que son necesarias a lo largo del programa. Su idea es hacer que los vectores de características de las firmas sean únicos con el fin de que toda función interna del programa que los necesite no tenga la obligación de almacenar memoria para un nuevo vector, ni sea necesario pasar parámetros por referencia en muchas de los métodos del proyecto. Además crea otro tipo de constantes necesarias para el software como el tamaño de cada bloque de la imagen a tratar durante la tapa de extracción de características.

Esto libera mucha memoria del programa, lo que supone que ganemos en **rapidez** y en **eficiencia**.

El único método que se implementa es el procedimiento **inicializa()** que establece los valores iniciales a las constantes de la clase.

Clase Imagenes.cs

Al igual que la clase anterior, es una clase general que almacena las imágenes que se van construyendo a partir de la firma manuscrita como por ejemplo la imagen Sobel o la de Gradiente. Su único método interesante es **inicializaMatricesImagen()**, que crea las matrices de la imagen con el tamaño necesario según el tamaño de la imagen de la firma que se haya establecido.

Clase recFirmaDAO.cs

La clase ***recFirmaDAO.cs*** es la encargada de implementar las funciones de acceso a la base de datos. Esta clase desarrolla los siguientes métodos:

salvaOActualizaPersona

Tipo del resultado bool : Informa de si se han podido salvar o actualizar los datos de la persona en la base de datos

Descripción: Realiza la inserción o actualización de los datos de una persona en la base.

Parámetros: persona : Persona -- *(persona que se desea incluir o actualizar en la base de datos)*

Tipo Funcion : String -- *(función que debe realizar, ya sea insertar nueva persona o actualizar sus datos)*

buscarPersona

Tipo del resultado: Persona : persona buscada

Descripción Realiza una búsqueda de la persona en la base de datos a través de su DNI y devuelve los datos de la persona junto con su firma si está estuviese almacenada

Parámetros persona : Persona -- *(la persona que se desea buscar)*

Comentarios La búsqueda se realiza a través del C.I de la persona.

- Si la persona no existe en la base de datos se devuelve null.

Clase Persona.cs

Esta clase es la encargada del tratamiento a nivel de capa de negocio de las personas con las que interactúa el software de esta aplicación.

Persona

Descripción: Este tipo de objetos representa a todas las personas que pueden ser añadidas, modificadas o extraídas de la base de datos de personas. Se ha diseñado en una clase para fomentar la alta cohesión y el bajo acoplamiento del software.

Atributos	Nombre	Tipo
	C.I	String
	Nombre	String
	Apellidos	String
	Email	String
	Dirección	String
	Código Postal	String
	Localidad	String
	Provincia	String
	Firma	byte[]

Para cada uno de los atributos de la clase existen un método Get y un método Set, que son los encargados de acceder a ellos desde las demás clases donde se necesitan. A través de estos atributos se almacena la información de las personas que se van a insertar o se buscan dentro de la base de datos.

Clase TransformaImagen.cs

Se trata de la clase más importante de la aplicación ya que es la que realiza la extracción de características de las firmas, lo que supone la mayor parte del tiempo de ejecución de los algoritmos. Contiene tanto las funciones de tratamiento de la imagen de la firma como las que se encargan de aplicar los diferentes algoritmos como Sobel, GSC, etc…

Se ha implementado así buscando el encapsulamiento de las funciones relacionadas, siempre intentando desarrollar un proyecto con alta cohesión y bajo acoplamiento.

Para la llamada a los métodos de la clase TransformaImagen se trabaja a través de un objeto de dicha clase, que es una imagen, todos los métodos de la clase trabajan sobre la imagen llamante, por lo cual nos ahorramos el paso de parámetros de la imagen cada vez que se llame a un método. Esto supone a su vez un ahorro importante en memoria requerida por el programa, lo que mejor la rapidez del mismo.

Los métodos más importantes de la clase TransformaImagen se describen a continuación:

ConvertirAGrises

Tipo del resultado void : Esta imagen no devuelve nada, ya que trabaja sobre la imagen sobre

la que se llama a la clase.

Descripción: Realiza la conversión de una imagen de entrada en una imagen en escala de grises.

Binarizar

Tipo del resultado void : Esta imagen no devuelve nada, ya que trabaja

sobre la imagen sobre la que se llama a la clase.

Descripción: Realiza la conversión de una imagen de entrada en una imagen en blanco y negro, además guarda la información en una matriz de booleanos.

Parámetros: pixels : bool[][] -- *(contendrá una matriz de booleans cuyo valor en la posicion i,j será true si el pixel i, j es negro)*

AplicarSobel

Tipo del resultado void : Esta imagen no devuelve nada, ya que trabaja sobre la imagen sobre la que se llama a la clase.

Descripción: Esta función aplica el operador de Sobel a la imagen de entrada,

modificándola, y almacenando la imagen resultante en una matriz para su posterior tratamiento.

Gradiente

Tipo del resultado void : Esta imagen no devuelve nada, ya que trabaja sobre la imagen sobre la que se llama a la clase.

Descripción: Lleva a cabo el cálculo del gradiente de la imagen, es decir, para cada píxel de la imagen calcula dirección de su ángulo. Este tratamiento se realiza a partir del operador de Sobel.

binaryFeatures

Tipo del resultado void : Esta imagen no devuelve nada, ya que trabaja sobre la imagen sobre la que se llama a la clase.

Descripción: Dentro de las características de gradiente, las características binarias son el primer grupo de características que se extraen.

Parámetros: fila : Integer -- *(fila del píxel actual que estamos tratando)*

columna : Integer -- *(columna del píxel actual que estamos considerando)*

valor : Double -- *(valor del gradiente del píxel que te está considerando)*

directionalFeatures

Tipo del resultado void : Esta imagen no devuelve nada, ya que trabaja sobre la imagen sobre la que se llama a la clase.

Descripción: Esta función realiza el cálculo de las características direccionales, que son el segundo grupo de características extraídas de las características de gradiente.

Comentarios: Se extraen directamente de las características binarias.

structuralFeatures

Tipo del resultado void : Esta imagen no devuelve nada, ya que trabaja sobre la imagen sobre la que se llama a la clase

Descripción Extrae las características estructurales de la imagen tratada.

concavityFeatures

Tipo del resultado void : Esta imagen no devuelve nada, ya que trabaja sobre la imagen sobre la que se llama a la clase.

Descripción: Esta función lleva a cabo el cálculo de las características de concavidad de

la imagen tratada, según el algotimo GSC.

rellenaVectoresCaractetisticas

Tipo del resultado void : Esta imagen no devuelve nada, ya que trabaja

sobre la imagen sobre la que se llama a la clase.

Descripción: Rellena el vector de características de la imagen extraídas con el algoritmo GSC.

CAPITULO V

COMPROBACIÓN DE LA HIPOTESIS

5.1. HIPOTESIS

Con el diseño e implementación de “Seguridad por medio de software para el reconocimiento de firmas manuscritas digitalizadas”, se pretende dar una solución flexible y económica para comprobar la autenticidad de un determinado usuario

5.2 Información Proporcionada

Existen multitud de mecanismos destinados a la verificación de los diferentes usuarios, como pueden ser tanto las técnicas basadas en firmas digitales o electrónicas, como el uso de otra serie de técnicas destinadas a la verificación e identificación de firmas manuscritas digitalizadas. Aunque en la actualidad está muy extendido el uso de las firmas digitales, el uso conjunto de técnicas de reconocimiento de firmas manuscritas digitalizadas puede

aportar una serie de mejoras bastante considerables, tanto en la fiabilidad como en aspectos tan importantes para el usuario como la comodidad del usuario identificado.

5.2.1 Sistemas Tradicionales

Entre los sistemas de de identificación aportados por el DNI, uno de los más notables es el de la firma manuscrita, que aparece en el frontal de cada documento. Esta firma constituía hasta el momento un mecanismo por el cual se identificaba unívocamente al poseedor del DNI, esto es, en caso de que se necesitara comprobar la rúbrica de un firmante en un instante determinado, siempre se ha recurrido a comprobar de manera visual que dicha firma coincidía en gran medida con la que aparece en su Documento Nacional de Identidad.

Hoy en día, gracias a la aparición de las nuevas tecnologías digitales como Internet y el crecimiento de las posibilidades de los ciudadanos (por ejemplo la cantidad de trámites online a los que se tiene acceso como la declaración de la renta), se nos plantea la necesidad de que dichos ciudadanos cuando realizan cualquier operación importante a través de la red, no puedan falsificar su información personal. Por ejemplo se debe conseguir evitar que un determinado usuario consiga datos privados de otra persona (como datos fiscales o financieros) o simplemente que mediante una suplantación de datos, sea capaz de acceder a información que no está autorizado a obtener.

Para intentar evitar esta serie de fraudes, surgen una serie de

mecanismos que han ido evolucionando a lo largo de los años:

El primer intento se produjo con el uso de los conocidos usuario/contraseña, que consiste en que cada usuario tenga un identificador y una contraseña asociada, a través de los cuales acceda unívocamente a los servicios que desee realizar. Sin embargo este tipo de verificación no es nada fiable ya que dichos nombres de usuarios y contraseñas son relativamente fáciles de conseguir. Una primera mejora se intentó aconsejando a los usuarios que escogieran contraseñas seguras (es decir, difíciles de descifrar con un demonio informático), pero la única mejora que se conseguía con esto era retrasar un poco el tiempo de obtención del password.

Esto ocasionó que empezaran a aparecer mecanismos más evolucionados como la de la firma electrónica (o certificado digital) y las firmas manuscritas digitalizadas.

La firma electrónica consiguió evitar los problemas de desciframiento de contraseñas antes mencionados con el uso del ***algoritmo*** de encriptación

A grandes rasgos, en el certificado digital se envía la información de la persona encriptada con una ***clave pública***, lo que hace que sea visible para todos los organismos. Además la información que el usuario recibe viene también encriptada con dicha clave. La seguridad de este sistema consiste en que para poder desencriptar dicha información se ha de estar en posesión de la ***clave privada***, que solo la posee el propietario del certificado digital. Esta clave es compleja de descifrar, ya que la encriptación del algoritmo es muy segura, siendo necesario para poder lograr dicho objetivo usar un sistema muy potente, además de emplear un tiempo tan elevado que motiva que

cualquier tercera persona que pretenda conseguir información desestime esta opción.

Por otra parte y paralelamente se desarrolló un sistema basado en las firmas manuscritas, que consistía en obtener un mecanismo que permitiera a partir de la firma o rúbrica de la persona la información necesaria para asegurar si ésta se corresponde con la firma real de la misma o se trata de una falsificación. Esta rama se dividió en dos investigaciones fundamentales, la primera de ellas pretendía comprobar la veracidad a partir del trazo que iba realizando el usuario instantáneamente (hablamos de técnicas ***on-line***) mientras que la segunda buscaba no necesitar la presencia en tiempo real de la persona sino que se pudiese decidir a través de una firma realizada en cualquier instante de tiempo (en este caso se trata de las firmas ***off-line***).

En la actualidad existen implementados varios sistemas de reconocimiento de firmas manuscritas (tanto ***off-line***, como ***on-line***) que intentan comprobar la veracidad de un determinado par de firmas, es decir, verificar si ambas firmas corresponden a una misma persona. Dichos sistemas necesitan de un análisis automatizado de las imágenes que contienen las firmas.

El proceso de análisis de una imagen consiste en comprobar, identificar, clasificar y comparar ciertos patrones característicos embebidos en la imagen que se está procesando. Como norma general los sistemas de reconocimiento de firma ***off-line*** siguen dos pasos fundamentales a la hora de realizar cualquier comprobación, dichos pasos descritos a grandes rasgos se pueden agrupar en *extracción de características* de la imagen y *comparación de las características* de las imágenes que se están verificando. Siguiendo este modelo pueden

derivarse múltiples tipos de analizadores, dependiendo de los algoritmos que se usen para cada una de dichas partes. Por ejemplo supongamos que la extracción de características se hace a través del estudio de la concavidad de las curvas de la firma y la comparación a través de la distancia estadística de ambas, o simplemente que usemos el mismo sistema de comparación estadística y se escoja un sistema diferente de extracción de características.
Por ello, los reconocedores ***off-line*** de firmas manuscritas suelen estar compuesto de ese par de funciones:

5.2.2 **Procesos del sistema**

- **Extracción de características**, usando para ello un algoritmo determinado se obtiene una serie de resultados a partir de la imagen tratada, que varían con cada tipo de firma que se esté estudiando.
- **Estudio comparativo**, de las características obtenidas en cada firma, comparando los resultados a través de un determinado algoritmo que variará según el reconocedor que se esté utilizando.

Dependiendo la fiabilidad de los algoritmos usados en cada una de las partes, se obtendrá un resultado más o menos fiable, siempre dentro de la fiabilidad que proporciona el marco de las firmas manuscritas

5.3 Comprobación de la Hipótesis

Para la comprobación de la hipótesis anteriormente citada se realizo la evaluación en base a los resultados de tiempo que toma el o los procesos para

la verificación de información manejada para el control de las firmas manuscritas digitalizadas.

Con el esquema plateado y con la hipótesis analizada con los tipos de variables examinadas se ha decidido por realizar un estudio para determinar la **Distancia Euclídea Matemática, por peso, y por Distancia Estadística,** para de esta manera encontrar la Verificación y Validación de las Firmas**,** entre los tipos de procesos.

5.4 Determinación de las Variables a ser Analizadas

Se ha logrado identificar las variables que intervienen en el análisis para la comprobación de la hipótesis mencionada.

- **Variable Independiente:** Proceso Tradicional para la recopilación de la información, escrita, y verificación mediante la cedula de identidad.
- **Variable Dependiente:** Integración de Sistemas mediante software, y mediante el uso de dispositivos Hardware.

5.5.- Métodos y Técnicas Utilizadas

5.5.1 Métodos

Método Inductivo

A partir de la información recopilada se inducirá para llegar a un conocimiento general y amplio para llegar una solución óptima de desarrollo.

5.5.2 Técnicas

INVESTIGACIÓN

Actualmente se ha determinado que es una necesidad el saber realizar proyectos mediante un fundamento técnico como es la investigación para así garantizar buenos resultados.

ANALISIS

La información que se manejará es fundamental para establecer la viabilidad y factibilidad del proyecto.

APLICACIÓN

Es el resultado de la automatización para el control y medio de seguridad para verificar si la rubrica de un determinado usuario es el mismo.

5.5.3. Validación de la hipótesis

Si se puede contar con componentes tanto hardware y software que permita agilitar el trabajo, se esta en la capacidad de presentar la solución más optima y productiva, de esta manera se demuestra que "*la integración de un sistema de software y mediante el uso de dispositivos hardware mejoran la métodologia y el desarrollo de los procesos tradicionales de lectura y procesamiento de datos.*

5.5.4. COMPROBACION ESTADISTICA

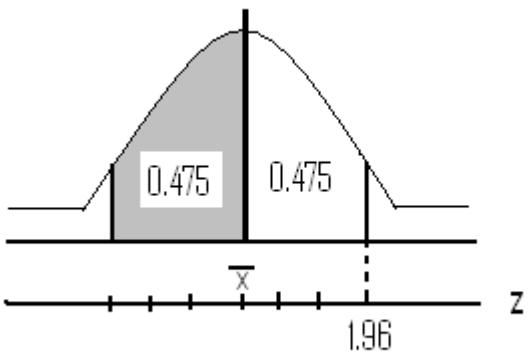
Con una población de 90 personas como referencia, las mismas realizarán el ingreso de datos por medio del software, se calculara el tamaño de la muestra, para saber a ciencia cierta cuantas personas deberán utilizar el software para las pruebas respectivas.

N = 90;

$$n = \frac{P(1-P)}{\frac{E^2}{Z^2} + \frac{P(1-P)}{N}}$$

Donde:

N es el tamaño de la población
n es el tamaño de la muestra
1-P es el rechazo
P es la aceptación
E es el error muestral



Z es el número de desviación típica
Datos:
N= 90
P= 0.5
1-P = 0.5
E = 0.05
Z =?
Nc= 95%

Según la tabla de áreas bajo la curva el valor de Z = 1.96

$$n = \frac{0.5(0.5)}{\frac{0.05^2}{1.96^2} + \frac{0.5(0.5)}{90}}$$

n = 26.60

n = 27 personas

Verificación para Corrección

$$\frac{n}{N} = \frac{27}{90} = 0.3 = 30\%$$

Se hace necesaria la corrección del tamaño de la muestra

$$nc = \frac{n * N}{N + n - 1}$$

$$nc = \frac{27 * 90}{90 + (27 - 1)}$$

nc = 20, 95=21 personas a investigar para tener un nivel de confianza de un 95 %

Luego de utilizar el software las 21 personas quedaron satisfechas y conformes, con la utilización del programa.

No teniendo ningún inconveniente con el manejo del tablero digital y programa en si.

RESUMEN

Utilizando las diferentes técnicas de muestreo y de análisis en el reconocimiento de firmas manuscritas digitalizadas se diseñó y construyó un software, buscando ayudar a prevenir su falsificación, con el fin de preservar y proteger los datos.

Fue diseñado el software en Visual Studio.net versión 2008, en el lenguaje de Visual Basic orientado a objetos para las máscaras o etiquetas, y con Acces, para control y creación de la base de datos. El ingreso de la firma únicamente se la realiza por medio de un tablero digitalizador o con el click derecho del mouse, El usuario ingresa sus datos personales y su firma, ésta se almacena en una base de datos con el fin de que cuando el usuario desee realizar alguna transacción, únicamente digitalice su firma, pasando el programa a realizar la respectiva comparación por medio de tres técnicas diferentes. En el diseño se aplicó método inductivo hasta encontrar la solución óptima mediante técnicas de investigación, análisis y aplicación.

Con este software creado se podría implementar en entidades públicas, empresas bancarias, dando un soporte de seguridad de hasta un 95% de grado de confianza.

Summary

Using the different sampling techniques and of analysis in the recognition of digitized hand written signatures was designed and it built a software, looking for to help to prevent their falsification, with the purpose of to preserve and to protect the data.

The software was designed in Visual Studio.net version 2008, in the language of Visual Basic guided to objects for the masks or labels, and with Access, for control and creation of the database. The entrance of the signature is only carried out it by means of a board digitalizador or with the right click of the mouse, The user enters his personal data and his signature, this is stored in a database with the purpose of that when the user wants to carry out some transaction, only digitize his signature, passing the program to carry out the respective comparison by means of three different techniques. In the design inductive method was applied until finding the good solution by means of technical of investigation, analysis and application.

With this created software you could implement in public entities, bank companies, giving a support of security of until 95% of grade of trust.

Conclusiones.

- La principal desventaja de la firma electrónica son los pasos que se deben realizar para poder usarla desde un equipo público.
- No es necesario que la persona transporte ningún tipo de información como un certificado digital.
- La aplicación realizada servirá como medio de seguridad, ya que se escoge el rango de error en uno de las 3 formas de comprobación.
- Podemos utilizarla la aplicación tanto en el sistema operativo xp o Vista, para el sistema operativo xp se puede utilizar tanto el tablero digital así como el mouse, pero para el sistema operativo Vista únicamente el mouse, ya que

necesitaríamos un parche para el tablero.

- Se comprobó la correcta compatibilidad tanto de software y de hardware.
- Los resultados fueron alentadores y positivos por parte de las personas que evaluaron el software.

Recomendaciones.

- Se recomienda tener instalado windows installer versión 3.1 o superior
- Se recomienda tener instalado Microsoft.net frameworks 2.0 o superior para poder instalar la aplicación
- Para poder instalar Visual studio 2008, es aconsejable instalar el sistema operativo xp de forma integra o total, sin obviar ningún componente, por lo que no se aconseja instalar cualquier tipo de sistemas operativo xp de versión de descargue gratuita.
- Si se desea utilizar el tablero digital para el sistema operativo windows vista se aconseja descargarse de internet el parche respectivo para que sea compatible con este sistema operativo.

Anexos.

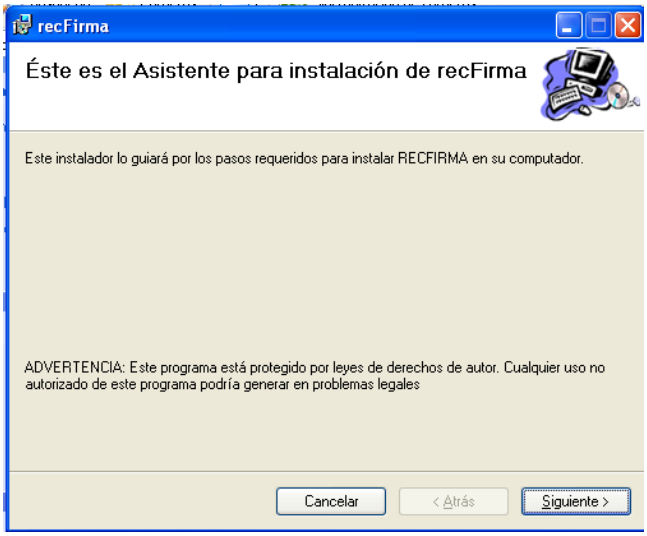
ANEXO A MANUAL DE USUARIO

Instalación de la Aplicación

Primero se da doble click en el paquete de instalación, que esta realizado en visual studio.net versión 2008.

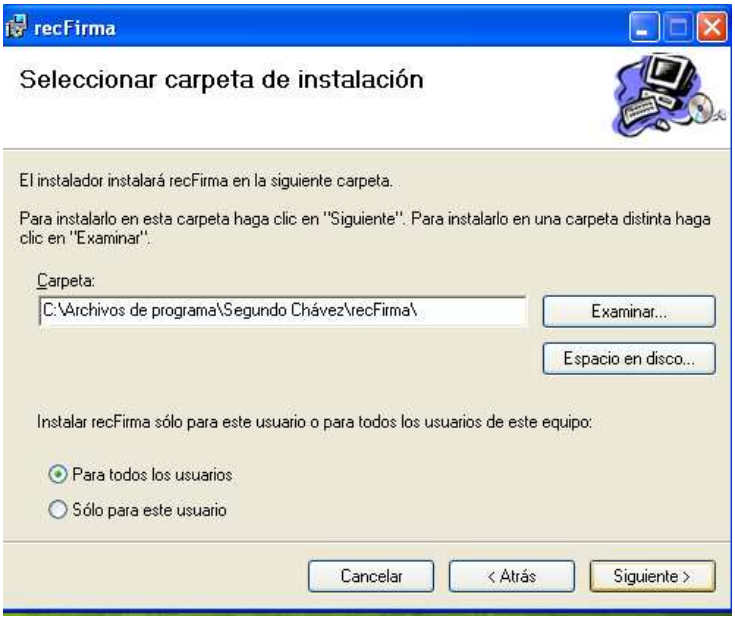
Iconos del Programa hecho ejecutable

Luego aparecerá la siguiente pantalla:



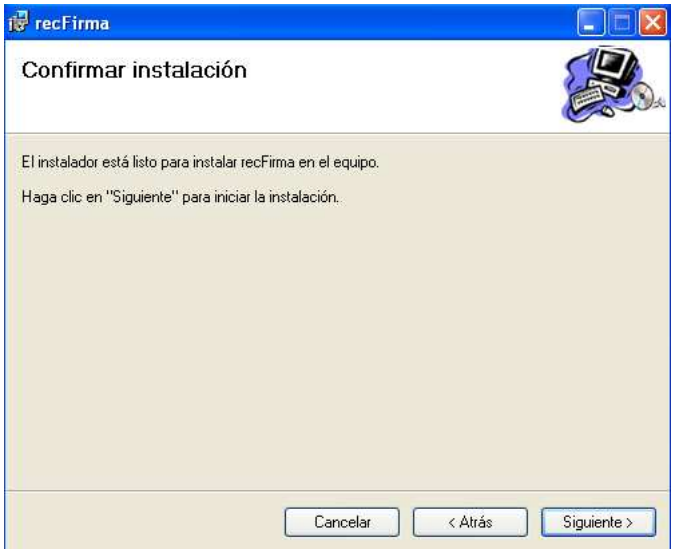
Asistente para la instalación.

Se da click en siguiente y aparecerá la ubicación en la que vamos a instalar el programa:



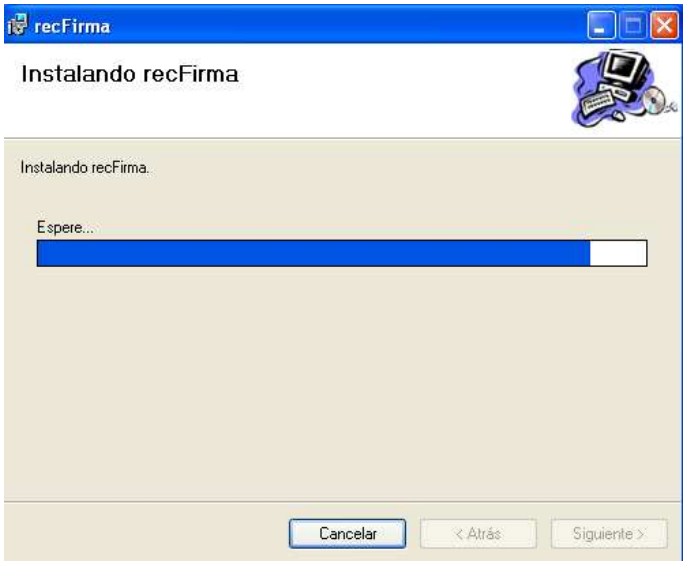
Selección de Carpeta de instalacion

Aparecerá la Pantalla de Confirmación de la Instalación.



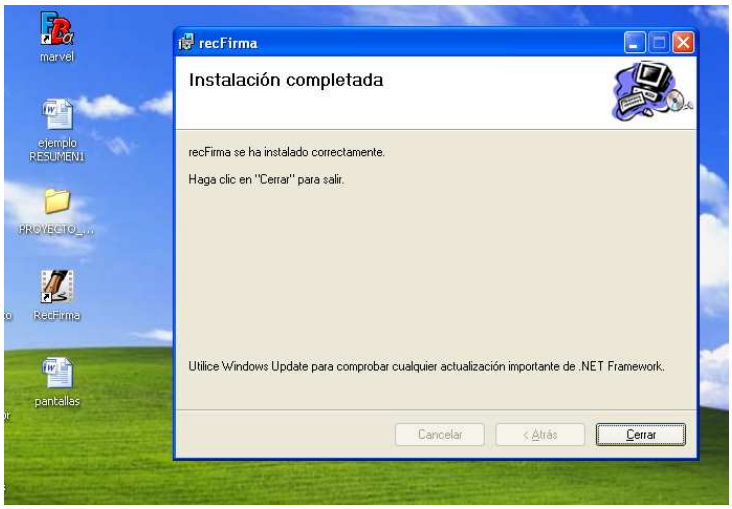
Confirmar la Instalacion

Instalando la Aplicación.



Instalando la Aplicación.

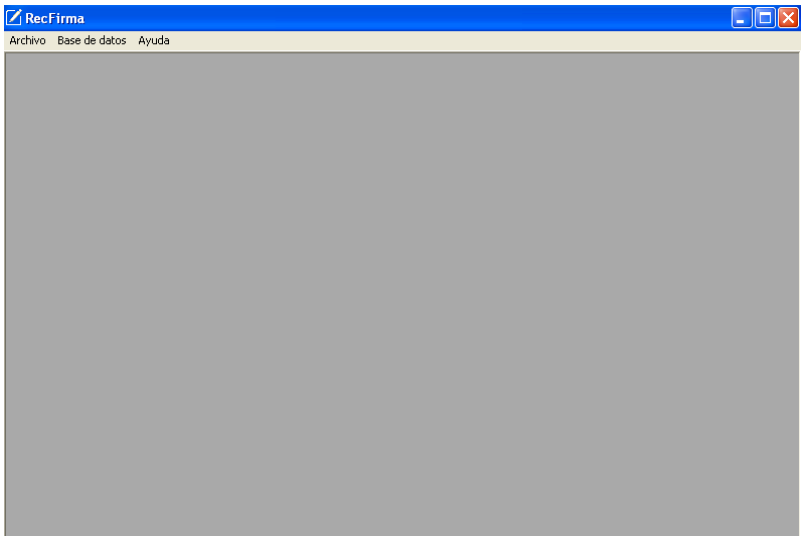
Finalización y colocación del acceso directo en el escritorio



Finalizacion de La instalación y Colocacion del Acceso Directo

Pantallas de Inserción de datos, y Pruebas.

Pantalla Principal en la que están presentes las Viñetas Archivo, en la se tiene como componente de Validar Firmas, la Viñeta Base de Datos las misma que tiene los componentes de Inserción, Editar y Consultar Personas y la Viñeta de Ayuda en la cual esta el Acerca de.

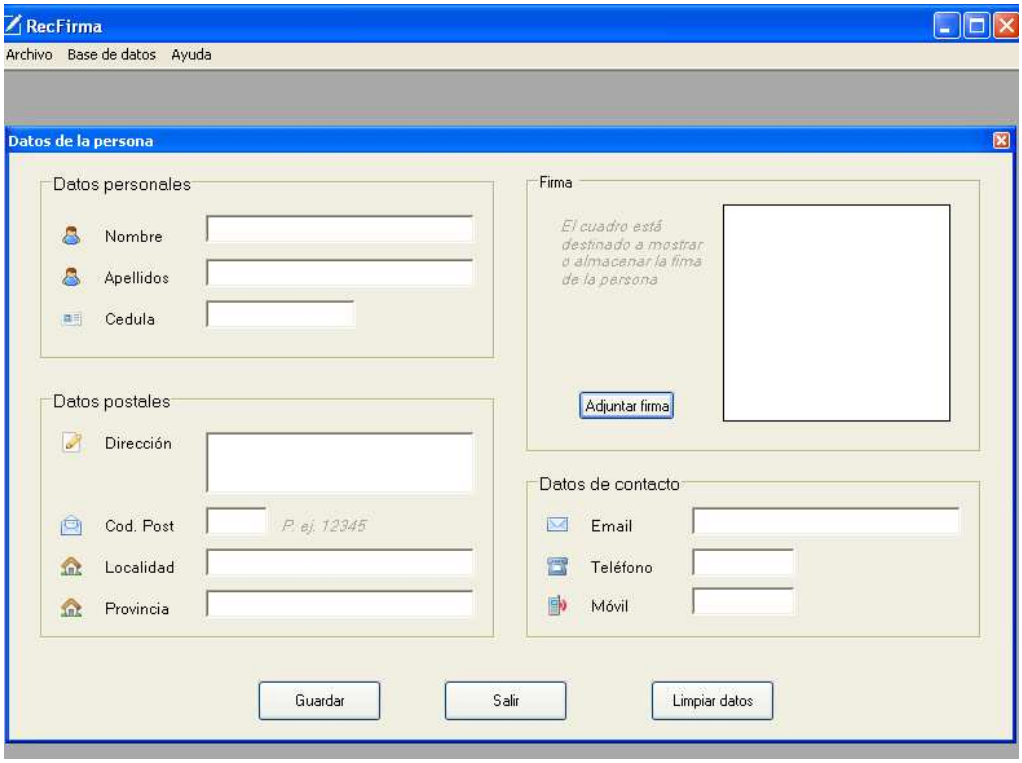


Pantalla Principal.

Viñeta Base de Datos

Opción de Inserción de Datos Personales

En esta pantalla básicamente se realiza la Inserción de los datos del usuario, que estan divididos de la siguiente manera Datos Personales y son: Nombre, Apellido, Numero de Cedula, Datos Postales y son: Direccion, Codigo Postal, Localidad Provincia, Datos de Contacto y son: Email, Telefono local, Celular o fono móvil, y la parte de inserción de la firma.

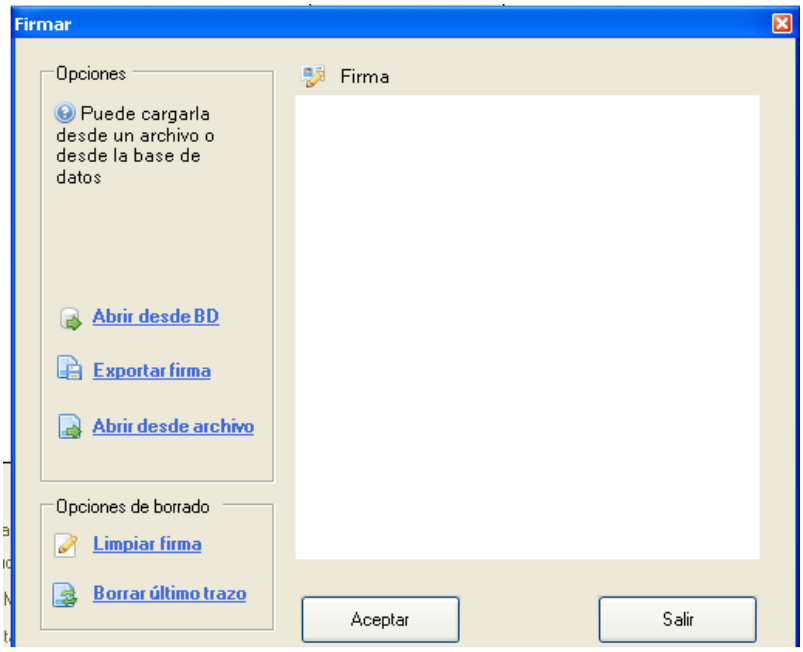


Pantalla del Ingreso de Datos

Cabe recalcar que los datos se almacenaran en una base hecha en Access, paquete que forma parte de visual studio.net.

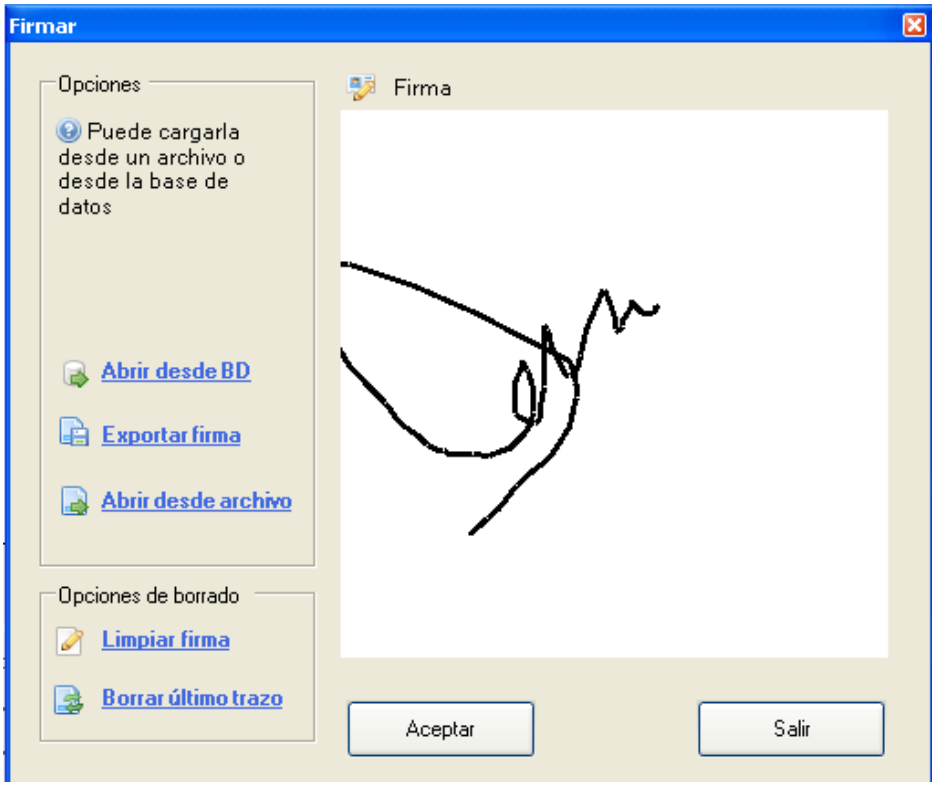
Ingreso de firma a la Base de Datos

En esta pantalla nos aparece las opciones de extraer las firmas desde una base de datos, de un archivo digital o exportándola directamente, para el ingreso de la firma se utiliza el tablero digital o en su defecto el click derecho del mouse, si tuviéramos algún tipo de equivocación o de error también tenemos la opción de eliminar el ultimo trazo o limpiar la pantalla completamente para poder realizar nuevamente la firma y de ahí guardarla.



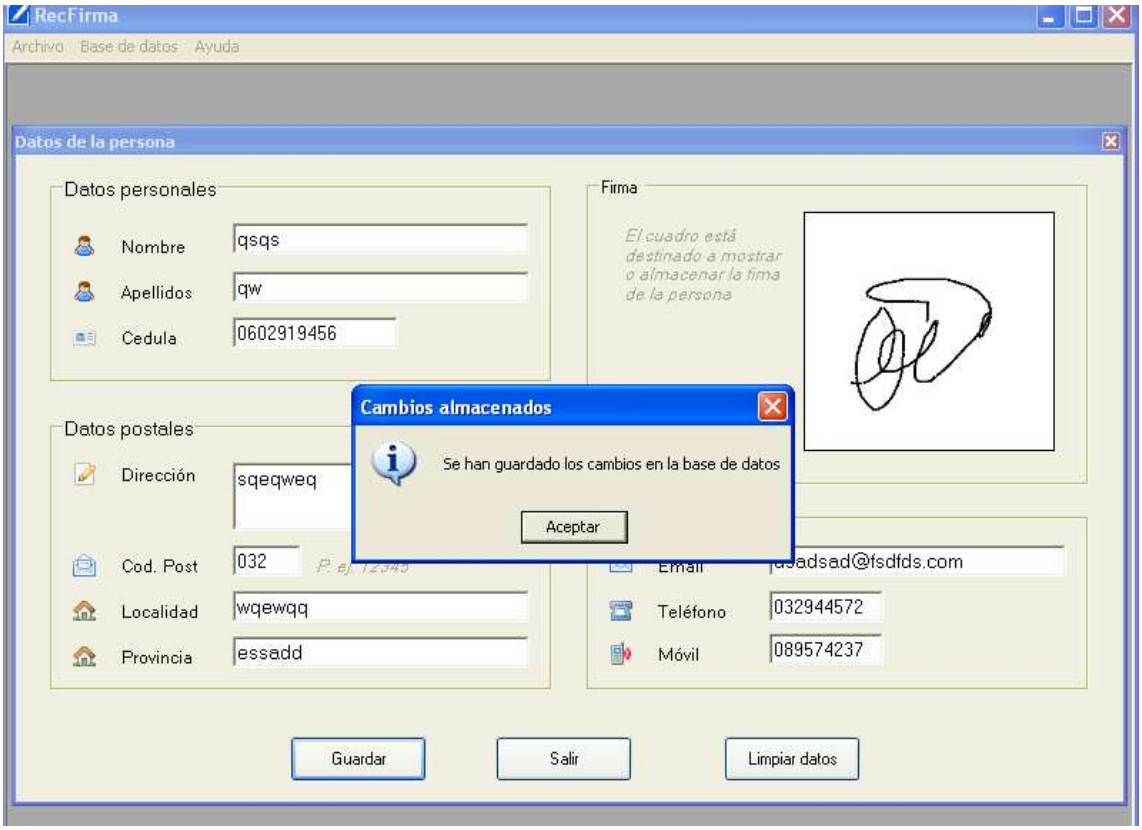
Pantalla de Ingreso de la Firma

Vista de una firma ingresada para su almacenamiento



Ingreso de la Firma.

La ventana queda de la siguiente manera con los datos.

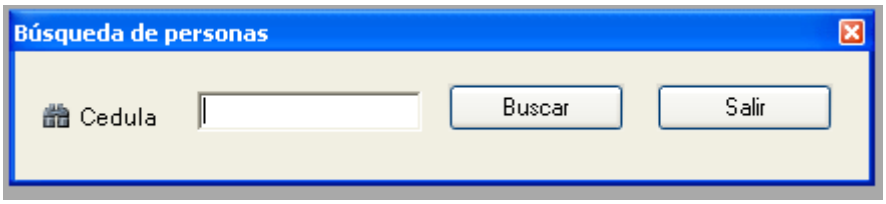


Datos Almacenados

De esta manera se almacena los datos en la base.

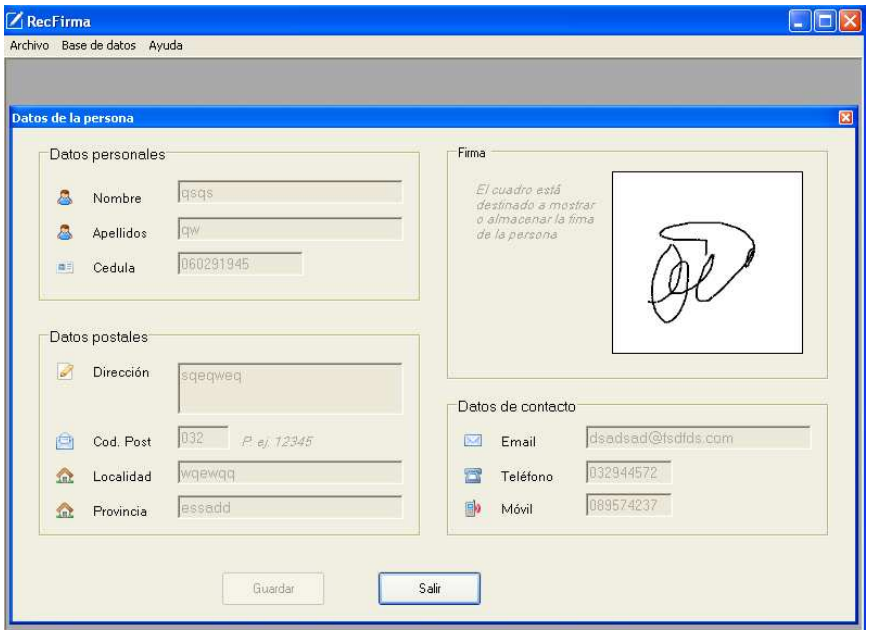
Opciones de Busqueda y Modificacion de Datos

Para la búsqueda y la Modificación de los datos se tiene la siguiente pantalla en común.



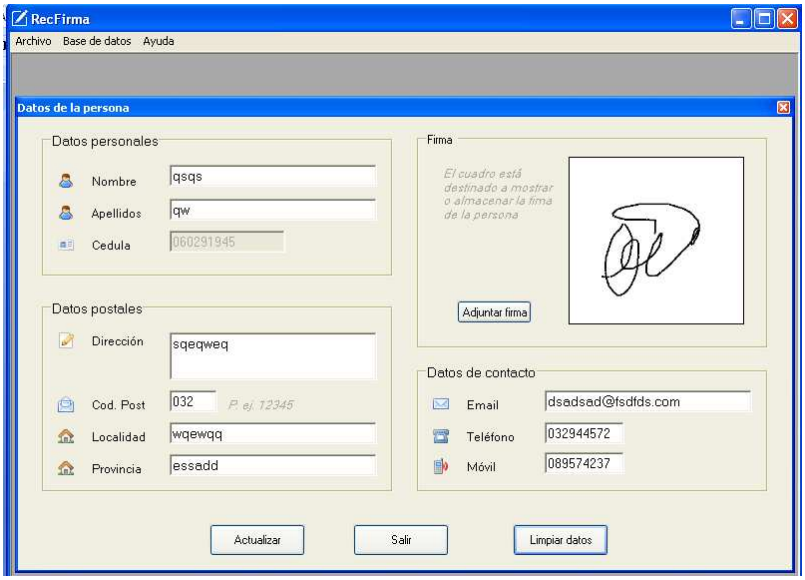
Ventana de Busqueda para actualizaciones y Consultas

Cuando sea únicamente consulta de datos del usuario aparecerá la pantalla de datos del mismo pero con las casillas desactivadas, esto es con el fin de no poder modificar los datos.



Pantalla de Consulta de Datos Personales

Pero cuando sea para modificación el usuario o digitador podrá ingresar normalmente los datos.

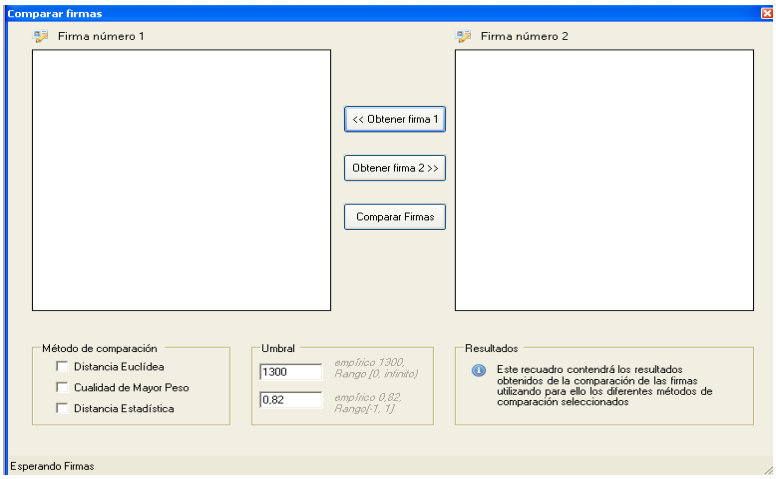


Pantalla de Actualizacion de Datos**.**

Viñeta Archivo

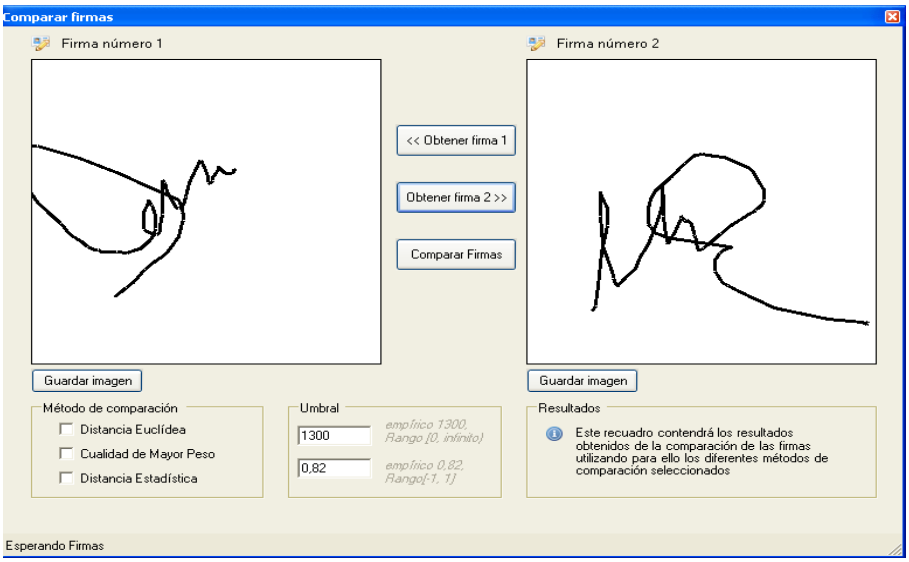
Comparación de las firmas Digitales

En esta opción nos aparecerá una ventana en la cual podremos ingresar la firma digital directamente por medio del tablero digital o en su defecto por medio del mouse, y compararla con la base de datos.



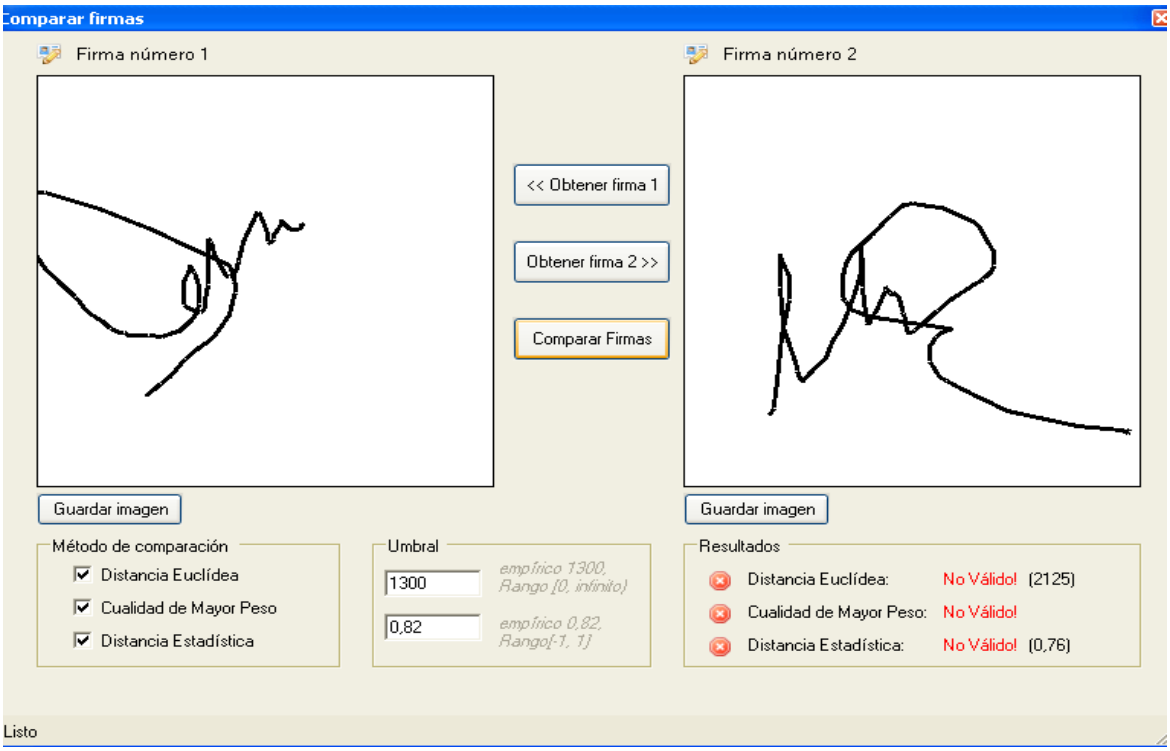
Pantalla de Comparacion de Firmas

Se ingresa las firmas digitales como datos y se tiene



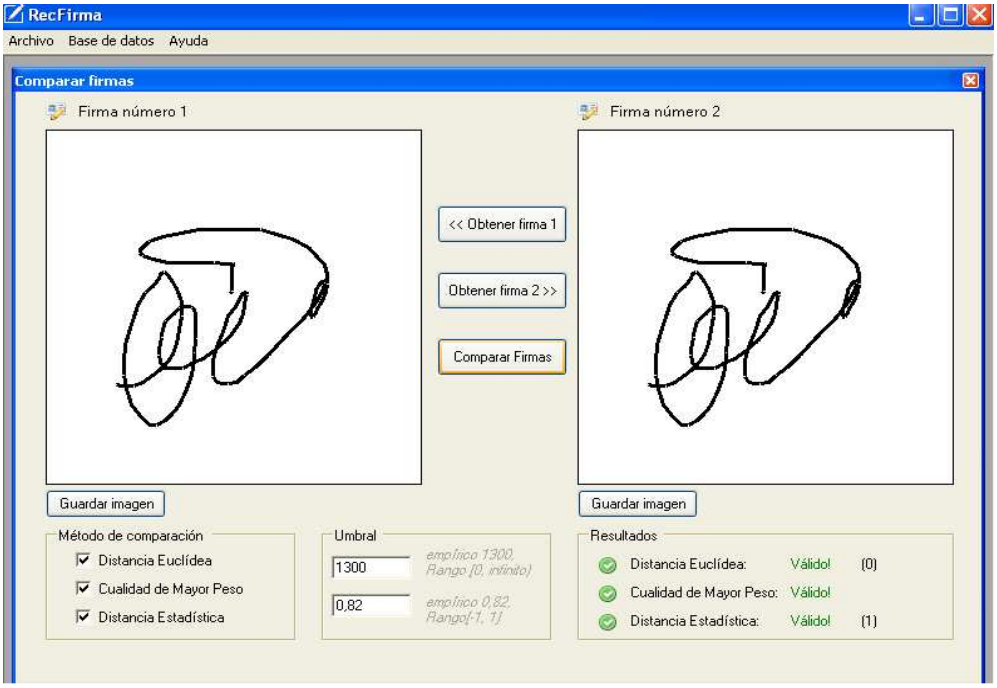
Comparacion de Firmas

Para la comparación de las firmas se la puede realizar de forma individual o de forma colectiva es decir un tipo de comprobación, con las 2 o las 3 como en la pantalla siguiente.



Muestra de Firmas Erradas, realizadas 3 tipos de comparación distinta

Lógicamente que si las firmas son las mismas nuestra pantalla dará en resultados como valida.

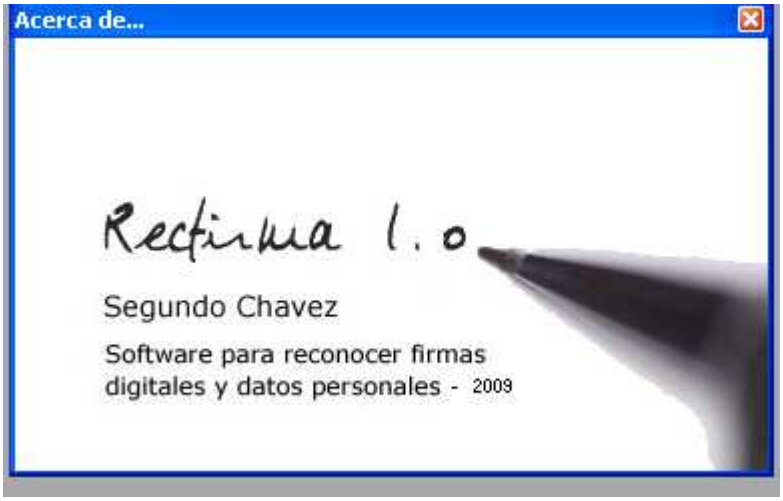


Visualizacion de Ingreso de Firmas Correctas

Viñeta Ayuda

Opcion Acerca de

En este apartado es para dar el crédito del Autor.



Creditos

ANEXO B CODIGO FUENTE
FORMULARIO PRINCIPAL

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using RecFirma.Clases;
using RecFirma.Formularios;
namespace RecFirma.Formularios
{
	public class PrincipalFrm : System.Windows.Forms.Form
	{	private Slusser.Components.MdiClientController mdiClientController;
		private System.Windows.Forms.MainMenu mainMenu1;
	private System.Windows.Forms.MenuItem menuItem1;
		private System.Windows.Forms.MenuItem menuItem3;
		private System.Windows.Forms.MenuItem menuItem4;
	private System.Windows.Forms.MenuItem menuItem5;
	private IContainer components;
		private System.Windows.Forms.MenuItem menuItem12;
		private System.Windows.Forms.MenuItem menuItem6;
		private System.Windows.Forms.MenuItem menuItem7;
		private System.Windows.Forms.MenuItem menuItem13;
		private System.Windows.Forms.MenuItem menuItem14;
		private System.Windows.Forms.MenuItem menuItem15;
		public recFirmaDAO DAO;
		public PrincipalFrm()
		{
			InitializeComponent();
			DAO = new recFirmaDAO();
		}
		#region Codigo generado por .NET
		/// <summary>
		/// Limpiar los recursos que se estén utilizando.
		/// </summary>
		protected override void Dispose( bool disposing )
		{	if( disposing )
			{	if (components != null)
				{	components.Dispose();
				}
			}
			base.Dispose( disposing );
		}
		#region Código generado por el Diseñador de Windows Forms
		/// <summary>
		/// Método necesario para admitir el Diseñador. No se puede modificar
		/// el contenido del método con el editor de código.
		/// </summary>
		private void InitializeComponent()
		{
	this.components = new System.ComponentModel.Container();
	System.Windows.Forms.MenuItem menuItem2;
	System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(PrincipalFrm));
	this.mdiClientController = new Slusser.Components.MdiClientController();
	this.mainMenu1 = new System.Windows.Forms.MainMenu(this.components);
```

```
this.menuItem1 = new System.Windows.Forms.MenuItem();
this.menuItem12 = new System.Windows.Forms.MenuItem();
this.menuItem5 = new System.Windows.Forms.MenuItem();
this.menuItem6 = new System.Windows.Forms.MenuItem();
this.menuItem7 = new System.Windows.Forms.MenuItem();
this.menuItem13 = new System.Windows.Forms.MenuItem();
this.menuItem14 = new System.Windows.Forms.MenuItem();
this.menuItem15 = new System.Windows.Forms.MenuItem();
this.menuItem3 = new System.Windows.Forms.MenuItem();
this.menuItem4 = new System.Windows.Forms.MenuItem();
menuItem2 = new System.Windows.Forms.MenuItem();
this.SuspendLayout();
//
// mdiClientController
//
this.mdiClientController.BackColor = System.Drawing.Color.DarkGray;
this.mdiClientController.ImageAlign = System.Drawing.ContentAlignment.MiddleRight;
this.mdiClientController.ParentForm = this;
//
// mainMenu1
//
this.mainMenu1.MenuItems.AddRange(new System.Windows.Forms.MenuItem[] {
this.menuItem1,
this.menuItem7,
this.menuItem3,
menuItem2});
//
// menuItem1
//
this.menuItem1.Index = 0;
this.menuItem1.MenuItems.AddRange(new System.Windows.Forms.MenuItem[] {
this.menuItem12});
this.menuItem1.Text = "Archivo";
//
// menuItem12
//
this.menuItem12.Index = 0;
this.menuItem12.Text = "Validar Firmas";
this.menuItem12.Click += new System.EventHandler(this.menuItem12_Click);
//
// menuItem2
//
menuItem2.Enabled = false;
menuItem2.Index = 3;
menuItem2.MenuItems.AddRange(new System.Windows.Forms.MenuItem[] {
this.menuItem5,
this.menuItem6});
menuItem2.Text = "Herramientas";
menuItem2.Visible = false;
//
// menuItem5
//
this.menuItem5.Index = 0;
this.menuItem5.Text = "Convertir a Blanco y Negro";
this.menuItem5.Click += new System.EventHandler(this.menuItem5_Click);
//
// menuItem6
//
```

```
this.menuItem6.Index = 1;
this.menuItem6.Text = "Redimensionar imagenes";
this.menuItem6.Click += new System.EventHandler(this.menuItem6_Click_1);
//
// menuItem7
//
this.menuItem7.Index = 1;
this.menuItem7.MenuItems.AddRange(new System.Windows.Forms.MenuItem[] {
this.menuItem13,
this.menuItem14,
this.menuItem15});
this.menuItem7.Text = "Base de datos";
//
// menuItem13
//
this.menuItem13.Index = 0;
this.menuItem13.Text = "Insertar persona";
this.menuItem13.Click += new System.EventHandler(this.menuItem13_Click);
//
// menuItem14
//
this.menuItem14.Index = 1;
this.menuItem14.Text = "Editar persona";
this.menuItem14.Click += new System.EventHandler(this.menuItem14_Click);
//
// menuItem15
//
this.menuItem15.Index = 2;
this.menuItem15.Text = "Consultar persona";
this.menuItem15.Click += new System.EventHandler(this.menuItem15_Click);
//
// menuItem3
//
this.menuItem3.Index = 2;
this.menuItem3.MenuItems.AddRange(new System.Windows.Forms.MenuItem[] {
this.menuItem4});
this.menuItem3.Text = "Ayuda";
//
// menuItem4
//
this.menuItem4.Index = 0;
this.menuItem4.Text = "Acerca de...";
this.menuItem4.Click += new System.EventHandler(this.menuItem4_Click);
//
// PrincipalFrm
//
this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);
this.ClientSize = new System.Drawing.Size(792, 566);
this.Icon = ((System.Drawing.Icon)(resources.GetObject("$this.Icon")));
this.IsMdiContainer = true;
this.Menu = this.mainMenu1;
this.Name = "PrincipalFrm";
this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
this.Text = "RecFirma";
this.ResumeLayout(false);
            }
            #endregion
            /// <summary>
```

```
		/// Punto de entrada principal de la aplicación.
		/// </summary>
		[STAThread]
		static void Main()
		{
			DateTime t = DateTime.Now;
			//TimeSpan tiempo = new TimeSpan();
	long milisegundos1 = Environment.TickCount, milisegundos2 =
Environment.TickCount;
			EsperaFrm f = new EsperaFrm();
			f.Show();
			f.Refresh();
			while((milisegundos2-milisegundos1) < 3000)
			{
				milisegundos2 = Environment.TickCount;
			}
			f.Close();
			Application.Run(new PrincipalFrm());
		}
		#endregion
		//Manejador Evento click del boton
		private void aBlancoYNegro_Click(object sender, System.EventArgs e)
		{
			pasaABlancoYNegro(sender, e);
		}

		private void menuItem4_Click(object sender, System.EventArgs e)
		{
			AcercaDeFrm f = new AcercaDeFrm(this);
			f.Show();
		}
		private void menuItem5_Click(object sender, System.EventArgs e)
		{
			pasaABlancoYNegro(sender, e);
		}
		private void pasaABlancoYNegro(object sender, System.EventArgs e)
		{
			TransformaFrm f = new TransformaFrm(this);
			f.Show();
		}
		private void menuItem12_Click(object sender, System.EventArgs e)
		{
			ComparaFirmasFrm f = new ComparaFirmasFrm(this);
			f.Show();
		}
		private void menuItem6_Click_1(object sender, System.EventArgs e)
		{
			ReducirImagenFrm f = new ReducirImagenFrm(this);
			f.Show();
		}
		private void menuItem13_Click(object sender, System.EventArgs e)
		{	PersonaFrm f = new PersonaFrm(this, 1, null);
			f.Show();
		}

		private void menuItem14_Click(object sender, System.EventArgs e)
		{
	Constantes.formularioCarga = false;
```

```
        FirmarFrm.CambioImagen = false;
                        BuscarFrm f = new BuscarFrm(this, 2);
                        f.Show();
                }
                private void menuItem15_Click(object sender, System.EventArgs e)
                {
        Constantes.formularioCarga = false;
        FirmarFrm.CambioImagen = false;
                        BuscarFrm f = new BuscarFrm(this, 3);
                        f.Show();
                }
        }
}
```

FORMULARIO PERSONA

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.IO;
using RecFirma.Clases;
namespace RecFirma.Formularios
{
        public class PersonaFrm : System.Windows.Forms.Form, Firmable
        {       #region Atributos generados por .NET
                private System.ComponentModel.Container components = null;
                private System.Windows.Forms.TextBox nombreTb;
                private System.Windows.Forms.TextBox apellidosTb;
                private System.Windows.Forms.Label label1;
                private System.Windows.Forms.Label label2;
                private System.Windows.Forms.Label label3;
                private System.Windows.Forms.TextBox emailTb;
                private System.Windows.Forms.TextBox telefonoTb;
                private System.Windows.Forms.TextBox movilTb;
                private System.Windows.Forms.Label label4;
                private System.Windows.Forms.Label label5;
                private System.Windows.Forms.Label label6;
                private System.Windows.Forms.TextBox direccionTb;
                private System.Windows.Forms.Label label7;
                private System.Windows.Forms.TextBox cpTb;
                private System.Windows.Forms.TextBox localidadTb;
                private System.Windows.Forms.Label label8;
                private System.Windows.Forms.TextBox provinciaTb;
                private System.Windows.Forms.Label label9;
                private System.Windows.Forms.OpenFileDialog abrirImagenOfd;
                private System.Windows.Forms.Label label10;
                private System.Windows.Forms.TextBox cdiTb;
                private Joaqs.UI.XpButton abrirBtn;
                private Joaqs.UI.XpButton limpiarBtn;
                private Joaqs.UI.XpButton guardarBtn;
                private System.Windows.Forms.PictureBox firmaPtb;
                private Joaqs.UI.XpButton cancelarBtn;
                #endregion
                private PrincipalFrm formularioPrincipal;
                private int tipo;
```

```
		private System.Windows.Forms.GroupBox groupBox1;
		private System.Windows.Forms.PictureBox pictureBox1;
		private System.Windows.Forms.PictureBox pictureBox2;
	private System.Windows.Forms.PictureBox pictureBox3;
		private System.Windows.Forms.GroupBox groupBox2;
		private System.Windows.Forms.PictureBox pictureBox4;
		private System.Windows.Forms.PictureBox pictureBox5;
		private System.Windows.Forms.PictureBox pictureBox6;
		private System.Windows.Forms.GroupBox groupBox3;
		private System.Windows.Forms.Label label12;
		private System.Windows.Forms.PictureBox pictureBox7;
		private System.Windows.Forms.PictureBox pictureBox8;
		private System.Windows.Forms.PictureBox pictureBox9;
		private System.Windows.Forms.PictureBox pictureBox10;
		private System.Windows.Forms.GroupBox groupBox4;
	private System.Windows.Forms.Label label13;
	private recFirmaDataSet recFirmaDataSet1;
		private Persona persona = null;
		public PersonaFrm(PrincipalFrm f, int tipo, Persona persona)
		{
			InitializeComponent();
			// Primero asignamos al formulario actual el formulario donde se abre.
			this.formularioPrincipal = f;
			this.MdiParent = this.formularioPrincipal;
			this.tipo = tipo;
			this.persona = persona;
			inicializaPorTipo(tipo);
		}

		#region Codigo generado por .NET
		/// <summary>
		/// Limpiar los recursos que se estén utilizando.
		/// </summary>
		protected override void Dispose( bool disposing )
		{	if( disposing )
			{	if(components != null)
				{	components.Dispose();
				}
			}
			base.Dispose( disposing );
		}
		#region Código generado por el Diseñador de Windows Forms
		/// <summary>
		/// Método necesario para admitir el Diseñador. No se puede modificar
		/// el contenido del método con el editor de código.
		/// </summary>
		private void InitializeComponent()
		{
	System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(PersonaFrm));
	this.firmaPtb = new System.Windows.Forms.PictureBox();
	this.nombreTb = new System.Windows.Forms.TextBox();
	this.apellidosTb = new System.Windows.Forms.TextBox();
	this.label1 = new System.Windows.Forms.Label();
	this.label2 = new System.Windows.Forms.Label();
	this.emailTb = new System.Windows.Forms.TextBox();
	this.label3 = new System.Windows.Forms.Label();
	this.telefonoTb = new System.Windows.Forms.TextBox();
```

```
this.movilTb = new System.Windows.Forms.TextBox();
this.label4 = new System.Windows.Forms.Label();
this.label5 = new System.Windows.Forms.Label();
this.label6 = new System.Windows.Forms.Label();
this.direccionTb = new System.Windows.Forms.TextBox();
this.cpTb = new System.Windows.Forms.TextBox();
this.label7 = new System.Windows.Forms.Label();
this.localidadTb = new System.Windows.Forms.TextBox();
this.label8 = new System.Windows.Forms.Label();
this.provinciaTb = new System.Windows.Forms.TextBox();
this.label9 = new System.Windows.Forms.Label();
this.abrirImagenOfd = new System.Windows.Forms.OpenFileDialog();
this.cdiTb = new System.Windows.Forms.TextBox();
this.label10 = new System.Windows.Forms.Label();
this.abrirBtn = new Joaqs.UI.XpButton();
this.limpiarBtn = new Joaqs.UI.XpButton();
this.guardarBtn = new Joaqs.UI.XpButton();
this.cancelarBtn = new Joaqs.UI.XpButton();
this.groupBox1 = new System.Windows.Forms.GroupBox();
this.pictureBox3 = new System.Windows.Forms.PictureBox();
this.pictureBox2 = new System.Windows.Forms.PictureBox();
this.pictureBox1 = new System.Windows.Forms.PictureBox();
this.groupBox2 = new System.Windows.Forms.GroupBox();
this.pictureBox6 = new System.Windows.Forms.PictureBox();
this.pictureBox5 = new System.Windows.Forms.PictureBox();
this.pictureBox4 = new System.Windows.Forms.PictureBox();
this.groupBox3 = new System.Windows.Forms.GroupBox();
this.pictureBox10 = new System.Windows.Forms.PictureBox();
this.pictureBox9 = new System.Windows.Forms.PictureBox();
this.pictureBox8 = new System.Windows.Forms.PictureBox();
this.pictureBox7 = new System.Windows.Forms.PictureBox();
this.label12 = new System.Windows.Forms.Label();
this.groupBox4 = new System.Windows.Forms.GroupBox();
this.label13 = new System.Windows.Forms.Label();
this.recFirmaDataSet1 = new RecFirma.recFirmaDataSet();
((System.ComponentModel.ISupportInitialize)(this.firmaPtb)).BeginInit();
this.groupBox1.SuspendLayout();
((System.ComponentModel.ISupportInitialize)(this.pictureBox3)).BeginInit();
((System.ComponentModel.ISupportInitialize)(this.pictureBox2)).BeginInit();
((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).BeginInit();
this.groupBox2.SuspendLayout();
((System.ComponentModel.ISupportInitialize)(this.pictureBox6)).BeginInit();
((System.ComponentModel.ISupportInitialize)(this.pictureBox5)).BeginInit();
((System.ComponentModel.ISupportInitialize)(this.pictureBox4)).BeginInit();
this.groupBox3.SuspendLayout();
((System.ComponentModel.ISupportInitialize)(this.pictureBox10)).BeginInit();
((System.ComponentModel.ISupportInitialize)(this.pictureBox9)).BeginInit();
((System.ComponentModel.ISupportInitialize)(this.pictureBox8)).BeginInit();
((System.ComponentModel.ISupportInitialize)(this.pictureBox7)).BeginInit();
this.groupBox4.SuspendLayout();
((System.ComponentModel.ISupportInitialize)(this.recFirmaDataSet1)).BeginInit();
this.SuspendLayout();
//
// firmaPtb
//
this.firmaPtb.BackColor = System.Drawing.Color.White;
this.firmaPtb.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;
this.firmaPtb.Location = new System.Drawing.Point(152, 24);
```

```
            this.firmaPtb.Name = "firmaPtb";
            this.firmaPtb.Size = new System.Drawing.Size(176, 168);
            this.firmaPtb.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
            this.firmaPtb.TabIndex = 1;
            this.firmaPtb.TabStop = false;
            //
            // nombreTb
            //
            this.nombreTb.Location = new System.Drawing.Point(128, 32);
            this.nombreTb.MaxLength = 50;
            this.nombreTb.Name = "nombreTb";
            this.nombreTb.Size = new System.Drawing.Size(208, 22);
            this.nombreTb.TabIndex = 2;
            //
            // apellidosTb
            //
            this.apellidosTb.Location = new System.Drawing.Point(128, 66);
            this.apellidosTb.MaxLength = 150;
            this.apellidosTb.Name = "apellidosTb";
            this.apellidosTb.Size = new System.Drawing.Size(208, 22);
            this.apellidosTb.TabIndex = 3;
            //
            // label1
            //
            this.label1.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label1.Location = new System.Drawing.Point(48, 40);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(72, 16);
            this.label1.TabIndex = 4;
            this.label1.Text = "Nombre";
            //
            // label2
            //
            this.label2.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label2.Location = new System.Drawing.Point(48, 72);
            this.label2.Name = "label2";
            this.label2.Size = new System.Drawing.Size(56, 16);
            this.label2.TabIndex = 5;
            this.label2.Text = "Apellidos";
            //
            // emailTb
            //
            this.emailTb.Location = new System.Drawing.Point(128, 26);
            this.emailTb.MaxLength = 60;
            this.emailTb.Name = "emailTb";
            this.emailTb.Size = new System.Drawing.Size(208, 22);
            this.emailTb.TabIndex = 6;
            //
            // label3
            //
            this.label3.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label3.Location = new System.Drawing.Point(48, 32);
            this.label3.Name = "label3";
            this.label3.Size = new System.Drawing.Size(56, 16);
            this.label3.TabIndex = 7;
```

```
            this.label3.Text = "Email";
            //
            // telefonoTb
            //
            this.telefonoTb.Location = new System.Drawing.Point(128, 58);
            this.telefonoTb.MaxLength = 9;
            this.telefonoTb.Name = "telefonoTb";
            this.telefonoTb.Size = new System.Drawing.Size(80, 22);
            this.telefonoTb.TabIndex = 8;
            //
            // movilTb
            //
            this.movilTb.Location = new System.Drawing.Point(128, 88);
            this.movilTb.MaxLength = 9;
            this.movilTb.Name = "movilTb";
            this.movilTb.Size = new System.Drawing.Size(80, 22);
            this.movilTb.TabIndex = 9;
            //
            // label4
            //
            this.label4.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label4.Location = new System.Drawing.Point(48, 64);
            this.label4.Name = "label4";
            this.label4.Size = new System.Drawing.Size(56, 16);
            this.label4.TabIndex = 10;
            this.label4.Text = "Teléfono";
            //
            // label5
            //
            this.label5.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label5.Location = new System.Drawing.Point(48, 94);
            this.label5.Name = "label5";
            this.label5.Size = new System.Drawing.Size(56, 16);
            this.label5.TabIndex = 11;
            this.label5.Text = "Móvil";
            //
            // label6
            //
            this.label6.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label6.Location = new System.Drawing.Point(48, 32);
            this.label6.Name = "label6";
            this.label6.Size = new System.Drawing.Size(64, 16);
            this.label6.TabIndex = 13;
            this.label6.Text = "Dirección";
            //
            // direccionTb
            //
            this.direccionTb.Location = new System.Drawing.Point(128, 32);
            this.direccionTb.MaxLength = 300;
            this.direccionTb.Multiline = true;
            this.direccionTb.Name = "direccionTb";
            this.direccionTb.Size = new System.Drawing.Size(208, 48);
            this.direccionTb.TabIndex = 14;
            //
            // cpTb
```

```
            //
            this.cpTb.Location = new System.Drawing.Point(128, 90);
            this.cpTb.MaxLength = 5;
            this.cpTb.Name = "cpTb";
            this.cpTb.Size = new System.Drawing.Size(48, 22);
            this.cpTb.TabIndex = 15;
            //
            // label7
            //
            this.label7.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label7.Location = new System.Drawing.Point(48, 96);
            this.label7.Name = "label7";
            this.label7.Size = new System.Drawing.Size(64, 16);
            this.label7.TabIndex = 16;
            this.label7.Text = "Cod. Postal";
            //
            // localidadTb
            //
            this.localidadTb.Location = new System.Drawing.Point(128, 122);
            this.localidadTb.MaxLength = 100;
            this.localidadTb.Name = "localidadTb";
            this.localidadTb.Size = new System.Drawing.Size(208, 22);
            this.localidadTb.TabIndex = 17;
            //
            // label8
            //
            this.label8.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label8.Location = new System.Drawing.Point(48, 128);
            this.label8.Name = "label8";
            this.label8.Size = new System.Drawing.Size(64, 16);
            this.label8.TabIndex = 18;
            this.label8.Text = "Localidad";
            //
            // provinciaTb
            //
            this.provinciaTb.Location = new System.Drawing.Point(128, 154);
            this.provinciaTb.MaxLength = 50;
            this.provinciaTb.Name = "provinciaTb";
            this.provinciaTb.Size = new System.Drawing.Size(208, 22);
            this.provinciaTb.TabIndex = 19;
            //
            // label9
            //
            this.label9.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label9.Location = new System.Drawing.Point(48, 160);
            this.label9.Name = "label9";
            this.label9.Size = new System.Drawing.Size(64, 16);
            this.label9.TabIndex = 20;
            this.label9.Text = "Provincia";
            //
            // cdiTb
            //
            this.cdiTb.Location = new System.Drawing.Point(128, 98);
            this.cdiTb.MaxLength = 10;
            this.cdiTb.Name = "cdiTb";
```

```
            this.cdiTb.Size = new System.Drawing.Size(116, 22);
            this.cdiTb.TabIndex = 21;
            this.cdiTb.KeyPress += new
System.Windows.Forms.KeyPressEventHandler(this.cdiTb_KeyPress);
            //
            // label10
            //
            this.label10.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label10.Location = new System.Drawing.Point(48, 104);
            this.label10.Name = "label10";
            this.label10.Size = new System.Drawing.Size(64, 16);
            this.label10.TabIndex = 22;
            this.label10.Text = "Cedula";
            //
            // abrirBtn
            //
            this.abrirBtn.Location = new System.Drawing.Point(40, 168);
            this.abrirBtn.Name = "abrirBtn";
            this.abrirBtn.Size = new System.Drawing.Size(75, 23);
            this.abrirBtn.TabIndex = 23;
            this.abrirBtn.Text = "Adjuntar firma";
            this.abrirBtn.Click += new System.EventHandler(this.abrirBtn_Click);
            //
            // limpiarBtn
            //
            this.limpiarBtn.Location = new System.Drawing.Point(496, 408);
            this.limpiarBtn.Name = "limpiarBtn";
            this.limpiarBtn.Size = new System.Drawing.Size(96, 32);
            this.limpiarBtn.TabIndex = 24;
            this.limpiarBtn.Text = "Limpiar datos";
            this.limpiarBtn.Click += new System.EventHandler(this.limpiarBtn_Click);
            //
            // guardarBtn
            //
            this.guardarBtn.Location = new System.Drawing.Point(192, 408);
            this.guardarBtn.Name = "guardarBtn";
            this.guardarBtn.Size = new System.Drawing.Size(96, 32);
            this.guardarBtn.TabIndex = 25;
            this.guardarBtn.Text = "Guardar";
            this.guardarBtn.Click += new System.EventHandler(this.guardarBtn_Click);
            //
            // cancelarBtn
            //
            this.cancelarBtn.Location = new System.Drawing.Point(336, 408);
            this.cancelarBtn.Name = "cancelarBtn";
            this.cancelarBtn.Size = new System.Drawing.Size(96, 32);
            this.cancelarBtn.TabIndex = 27;
            this.cancelarBtn.Text = "Salir";
            this.cancelarBtn.Click += new System.EventHandler(this.cancelarBtn_Click);
            //
            // groupBox1
            //
            this.groupBox1.Controls.Add(this.pictureBox3);
            this.groupBox1.Controls.Add(this.pictureBox2);
            this.groupBox1.Controls.Add(this.pictureBox1);
            this.groupBox1.Controls.Add(this.nombreTb);
            this.groupBox1.Controls.Add(this.cdiTb);
```

```
            this.groupBox1.Controls.Add(this.label10);
            this.groupBox1.Controls.Add(this.label2);
            this.groupBox1.Controls.Add(this.label1);
            this.groupBox1.Controls.Add(this.apellidosTb);
            this.groupBox1.Font = new System.Drawing.Font("Microsoft Sans Serif", 9.75F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.groupBox1.Location = new System.Drawing.Point(24, 16);
            this.groupBox1.Name = "groupBox1";
            this.groupBox1.Size = new System.Drawing.Size(352, 144);
            this.groupBox1.TabIndex = 28;
            this.groupBox1.TabStop = false;
            this.groupBox1.Text = "Datos personales";
            this.groupBox1.Enter += new System.EventHandler(this.groupBox1_Enter);
            //
            // pictureBox3
            //
            this.pictureBox3.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox3.Image")));
            this.pictureBox3.Location = new System.Drawing.Point(16, 104);
            this.pictureBox3.Name = "pictureBox3";
            this.pictureBox3.Size = new System.Drawing.Size(16, 16);
            this.pictureBox3.TabIndex = 25;
            this.pictureBox3.TabStop = false;
            this.pictureBox3.Click += new System.EventHandler(this.pictureBox3_Click);
            //
            // pictureBox2
            //
            this.pictureBox2.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox2.Image")));
            this.pictureBox2.Location = new System.Drawing.Point(16, 72);
            this.pictureBox2.Name = "pictureBox2";
            this.pictureBox2.Size = new System.Drawing.Size(16, 16);
            this.pictureBox2.TabIndex = 24;
            this.pictureBox2.TabStop = false;
            this.pictureBox2.Click += new System.EventHandler(this.pictureBox2_Click);
            //
            // pictureBox1
            //
            this.pictureBox1.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox1.Image")));
            this.pictureBox1.Location = new System.Drawing.Point(16, 40);
            this.pictureBox1.Name = "pictureBox1";
            this.pictureBox1.Size = new System.Drawing.Size(16, 16);
            this.pictureBox1.TabIndex = 23;
            this.pictureBox1.TabStop = false;
            //
            // groupBox2
            //
            this.groupBox2.Controls.Add(this.pictureBox6);
            this.groupBox2.Controls.Add(this.pictureBox5);
            this.groupBox2.Controls.Add(this.pictureBox4);
            this.groupBox2.Controls.Add(this.label5);
            this.groupBox2.Controls.Add(this.label4);
            this.groupBox2.Controls.Add(this.movilTb);
            this.groupBox2.Controls.Add(this.telefonoTb);
            this.groupBox2.Controls.Add(this.label3);
            this.groupBox2.Controls.Add(this.emailTb);
```

```
            this.groupBox2.Font = new System.Drawing.Font("Microsoft Sans Serif", 9.75F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.groupBox2.Location = new System.Drawing.Point(400, 248);
            this.groupBox2.Name = "groupBox2";
            this.groupBox2.Size = new System.Drawing.Size(352, 128);
            this.groupBox2.TabIndex = 29;
            this.groupBox2.TabStop = false;
            this.groupBox2.Text = "Datos de contacto";
            //
            // pictureBox6
            //
            this.pictureBox6.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox6.Image")));
            this.pictureBox6.Location = new System.Drawing.Point(16, 94);
            this.pictureBox6.Name = "pictureBox6";
            this.pictureBox6.Size = new System.Drawing.Size(16, 16);
            this.pictureBox6.TabIndex = 28;
            this.pictureBox6.TabStop = false;
            //
            // pictureBox5
            //
            this.pictureBox5.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox5.Image")));
            this.pictureBox5.Location = new System.Drawing.Point(16, 64);
            this.pictureBox5.Name = "pictureBox5";
            this.pictureBox5.Size = new System.Drawing.Size(16, 16);
            this.pictureBox5.TabIndex = 27;
            this.pictureBox5.TabStop = false;
            //
            // pictureBox4
            //
            this.pictureBox4.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox4.Image")));
            this.pictureBox4.Location = new System.Drawing.Point(16, 32);
            this.pictureBox4.Name = "pictureBox4";
            this.pictureBox4.Size = new System.Drawing.Size(16, 16);
            this.pictureBox4.TabIndex = 26;
            this.pictureBox4.TabStop = false;
            //
            // groupBox3
            //
            this.groupBox3.Controls.Add(this.pictureBox10);
            this.groupBox3.Controls.Add(this.pictureBox9);
            this.groupBox3.Controls.Add(this.pictureBox8);
            this.groupBox3.Controls.Add(this.pictureBox7);
            this.groupBox3.Controls.Add(this.label12);
            this.groupBox3.Controls.Add(this.direccionTb);
            this.groupBox3.Controls.Add(this.label6);
            this.groupBox3.Controls.Add(this.label8);
            this.groupBox3.Controls.Add(this.provinciaTb);
            this.groupBox3.Controls.Add(this.label9);
            this.groupBox3.Controls.Add(this.localidadTb);
            this.groupBox3.Controls.Add(this.label7);
            this.groupBox3.Controls.Add(this.cpTb);
            this.groupBox3.Font = new System.Drawing.Font("Microsoft Sans Serif", 9.75F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.groupBox3.Location = new System.Drawing.Point(24, 184);
            this.groupBox3.Name = "groupBox3";
```

```
            this.groupBox3.Size = new System.Drawing.Size(352, 192);
            this.groupBox3.TabIndex = 30;
            this.groupBox3.TabStop = false;
            this.groupBox3.Text = "Datos postales";
            //
            // pictureBox10
            //
            this.pictureBox10.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox10.Image")));
            this.pictureBox10.Location = new System.Drawing.Point(16, 160);
            this.pictureBox10.Name = "pictureBox10";
            this.pictureBox10.Size = new System.Drawing.Size(16, 16);
            this.pictureBox10.TabIndex = 31;
            this.pictureBox10.TabStop = false;
            //
            // pictureBox9
            //
            this.pictureBox9.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox9.Image")));
            this.pictureBox9.Location = new System.Drawing.Point(16, 128);
            this.pictureBox9.Name = "pictureBox9";
            this.pictureBox9.Size = new System.Drawing.Size(16, 16);
            this.pictureBox9.TabIndex = 30;
            this.pictureBox9.TabStop = false;
            //
            // pictureBox8
            //
            this.pictureBox8.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox8.Image")));
            this.pictureBox8.Location = new System.Drawing.Point(16, 96);
            this.pictureBox8.Name = "pictureBox8";
            this.pictureBox8.Size = new System.Drawing.Size(16, 16);
            this.pictureBox8.TabIndex = 29;
            this.pictureBox8.TabStop = false;
            //
            // pictureBox7
            //
            this.pictureBox7.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox7.Image")));
            this.pictureBox7.Location = new System.Drawing.Point(16, 32);
            this.pictureBox7.Name = "pictureBox7";
            this.pictureBox7.Size = new System.Drawing.Size(16, 16);
            this.pictureBox7.TabIndex = 28;
            this.pictureBox7.TabStop = false;
            //
            // label12
            //
            this.label12.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F,
System.Drawing.FontStyle.Italic, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label12.ForeColor = System.Drawing.SystemColors.GrayText;
            this.label12.Location = new System.Drawing.Point(184, 96);
            this.label12.Name = "label12";
            this.label12.Size = new System.Drawing.Size(100, 16);
            this.label12.TabIndex = 27;
            this.label12.Text = "P. ej. 12345";
            //
            // groupBox4
            //
```

```
            this.groupBox4.Controls.Add(this.label13);
            this.groupBox4.Controls.Add(this.firmaPtb);
            this.groupBox4.Controls.Add(this.abrirBtn);
            this.groupBox4.Location = new System.Drawing.Point(400, 16);
            this.groupBox4.Name = "groupBox4";
            this.groupBox4.Size = new System.Drawing.Size(352, 216);
            this.groupBox4.TabIndex = 31;
            this.groupBox4.TabStop = false;
            this.groupBox4.Text = "Firma";
            //
            // label13
            //
            this.label13.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F,
System.Drawing.FontStyle.Italic, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label13.ForeColor = System.Drawing.SystemColors.GrayText;
            this.label13.Location = new System.Drawing.Point(24, 32);
            this.label13.Name = "label13";
            this.label13.Size = new System.Drawing.Size(120, 64);
            this.label13.TabIndex = 27;
            this.label13.Text = "El cuadro está destinado a mostrar o almacenar la fima de la
persona";
            //
            // recFirmaDataSet1
            //
            this.recFirmaDataSet1.DataSetName = "recFirmaDataSet";
            this.recFirmaDataSet1.SchemaSerializationMode =
System.Data.SchemaSerializationMode.IncludeSchema;
            //
            // PersonaFrm
            //
            this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);
            this.BackColor = System.Drawing.SystemColors.ControlLight;
            this.ClientSize = new System.Drawing.Size(776, 454);
            this.Controls.Add(this.groupBox4);
            this.Controls.Add(this.groupBox3);
            this.Controls.Add(this.groupBox2);
            this.Controls.Add(this.groupBox1);
            this.Controls.Add(this.cancelarBtn);
            this.Controls.Add(this.guardarBtn);
            this.Controls.Add(this.limpiarBtn);
            this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedToolWindow;
            this.Name = "PersonaFrm";
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Datos de la persona";
            this.Closed += new System.EventHandler(this.PersonaFrm_Closed);
            ((System.ComponentModel.ISupportInitialize)(this.firmaPtb)).EndInit();
            this.groupBox1.ResumeLayout(false);
            this.groupBox1.PerformLayout();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox3)).EndInit();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox2)).EndInit();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).EndInit();
            this.groupBox2.ResumeLayout(false);
            this.groupBox2.PerformLayout();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox6)).EndInit();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox5)).EndInit();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox4)).EndInit();
            this.groupBox3.ResumeLayout(false);
            this.groupBox3.PerformLayout();
```

```
((System.ComponentModel.ISupportInitialize)(this.pictureBox10)).EndInit();
((System.ComponentModel.ISupportInitialize)(this.pictureBox9)).EndInit();
((System.ComponentModel.ISupportInitialize)(this.pictureBox8)).EndInit();
((System.ComponentModel.ISupportInitialize)(this.pictureBox7)).EndInit();
this.groupBox4.ResumeLayout(false);
((System.ComponentModel.ISupportInitialize)(this.recFirmaDataSet1)).EndInit();
this.ResumeLayout(false);
        }
        #endregion
        #endregion
        private void abrirBtn_Click(object sender, System.EventArgs e)
        {
                FirmarFrm f = new FirmarFrm(formularioPrincipal, this, this);
                f.Show();
                this.Enabled = false;
        }
        // Esta funcion limpia los datos de la pantalla
        private void limpiarBtn_Click(object sender, System.EventArgs e)
        {
                firmaPtb.Image = null;
                //Component c;
                if(tipo == 1) //Solo limpiamos el Cdi si no estamos editando la persona
                        cdi.Text = "";
                nombreTb.Text = "";
                apellidosTb.Text = "";
                emailTb.Text = "";
                telefonoTb.Text ="";
                movilTb.Text = "";
                direccionTb.Text = "";
                cpTb.Text = "";
                localidadTb.Text = "";
                provinciaTb.Text = "";
        }
        // Esta funcion tiene que guardar los datos en la base de datos
        private void guardarBtn_Click(object sender, System.EventArgs e)
        {
                string cadena = "";
                if(tipo == 1) // Estamos en modo crear nuevo usuario
                {
               // Creamos la persona que vamos a almacenar en la base de datos
                        persona = new Persona();
                        cadena = Constantes.TIPO_GUARDAR;
                }
                else if(tipo == 2) // Estamos en modo actualizar usuario
                {
                        cadena = Constantes.TIPO_ACTUALIZAR;
                }
                persona.Cdi = cdiTb.Text;
                persona.Nombre = nombreTb.Text;
                persona.Apellidos = apellidosTb.Text;
                persona.Email = emailTb.Text;
                persona.Telefono = telefonoTb.Text;
                persona.Movil = movilTb.Text;
                persona.Direccion = direccionTb.Text;
                persona.CodigoPostal = cpTb.Text;
                persona.Localidad = localidadTb.Text;
                persona.Provincia = provinciaTb.Text;
```

```
                // Carga una imagen en una matriz de bytes del campo firma de la clase
Persona
                        // Esto es necesario para poder almacenarla como objeto OLE
                       if(firmaPtb.Image != null && FirmarFrm.CambioImagen)
                        // Si se ha metido una imagen para la firma
                        {       MemoryStream ms = new MemoryStream();
                                // Cargamos la imagen en formato bmp en un MemoryStream
                                firmaPtb.Image.Save(ms,
System.Drawing.Imaging.ImageFormat.Bmp);
                                // Pasamos el MemoryStream a un array de bytes
                                persona.Firma = ms.ToArray();
                        }
                        if(!formularioPrincipal.DAO.salvaOActualizaPersona(persona,
cadena))
                        {
MessageBox.Show(this, "No se han podido guardar los cambios", "Error de base de datos",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                        }
                        else
                        {
MessageBox.Show(this, "Se han guardado los cambios en la base de datos", "Cambios
almacenados", MessageBoxButtons.OK, MessageBoxIcon.Information);
            this.Close();
                        }
                }
                private void PersonaFrm_Closed(object sender, System.EventArgs e)
                {
                        if(tipo == 2)
                                this.guardarBtn.Text = "Salvar";
                }
                private void cancelarBtn_Click(object sender, System.EventArgs e)
                {
                        this.Close();
                }
                #region Miembros de Firmable
                public void setFirma(Bitmap bitmap)
                {
                        firmaPtb.Image = bitmap;
                }
                #endregion
                #region Funciones auxiliares
                private void inicializaPorTipo(int tipo)
                {       if(tipo == 1) // Estamos en modo crear un nuevo usuario
                        {       this.guardarBtn.Enabled = true;
                        }
                        else if(tipo == 2) // Estamos en modo actualizar datos de usuario
                        {       this.guardarBtn.Text = "Actualizar";
                                this.guardarBtn.Enabled = true;
                                rellenaCampos();
                        }
                        else // Estamos en modo consultar datos de los usuarios
                        {       this.guardarBtn.Enabled = false;
                                this.limpiarBtn.Hide();
                                this.limpiarBtn.Enabled = false;
                                this.abrirBtn.Hide();
                                this.abrirBtn.Enabled = false;
                                rellenaCampos();
                        }
```

```
                }
                private void rellenaCampos()
                {
                // Hacemos un recorrido sobre los campos mostrando los datos de la persona
                        cdiTb.Text = persona.Cdi;
                        nombreTb.Text = persona.Nombre;
                        apellidosTb.Text = persona.Apellidos;
                        emailTb.Text = persona.Email;
                        telefonoTb.Text = persona.Telefono;
                        movilTb.Text = persona.Movil;
                        direccionTb.Text = persona.Direccion;
                        cpTb.Text = persona.CodigoPostal;
                        localidadTb.Text = persona.Localidad;
                        provinciaTb.Text = persona.Provincia;
                        // Ahora metemos en el PictureBox el array de bits que contiene
                        // la imagen que está en la base de datos
                        if (persona.Firma!=null)
                        {
                                //Transformar los bytes en una imagen de nuevo
                                MemoryStream ms = new MemoryStream();
                                ms.Write(persona.Firma, 0 ,persona.Firma.GetUpperBound(0)
+ 1);
                                firmaPtb.Image=System.Drawing.Image.FromStream(ms);
        if(firmaPtb.Image.Height < firmaPtb.Height && firmaPtb.Image.Width <
firmaPtb.Width)
                                        firmaPtb.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.CenterImage;
                                ms.Close();
                        }
        // Hacemos que el Cdi no sea modificable por la persona en modo edicion o mostrar
datos
                        cdiTb.Enabled = false;
                        if(tipo == 3)
                        {
                                nombreTb.Enabled = false;
                                apellidosTb.Enabled = false;
                                emailTb.Enabled = false;
                                telefonoTb.Enabled = false;
                                movilTb.Enabled = false;
                                direccionTb.Enabled = false;
                                cpTb.Enabled = false;
                                localidadTb.Enabled = false;
                                provinciaTb.Enabled = false;
                        }
                }
                #endregion
                private void pictureBox2_Click(object sender, System.EventArgs e)
                {
                }
                private void pictureBox3_Click(object sender, System.EventArgs e)
                {
                }
        private void groupBox1_Enter(object sender, EventArgs e)
        {
        }
        private void cdiTb_KeyPress(object sender, KeyPressEventArgs e)
        {
//          if (e.KeyChar == 45)
```

```
    }
        }
}
```

FORMULARIO TRANSFORMA

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Drawing.Imaging;
using RecFirma.Clases;
namespace RecFirma.Formularios
{
	public class TransformaFrm : System.Windows.Forms.Form
	{	private System.ComponentModel.IContainer components = null;
		private System.Windows.Forms.OpenFileDialog abrirImagen;
		private Joaqs.UI.XpButton abrirBtn;
		private System.Windows.Forms.PictureBox originalPtb;
		private System.Windows.Forms.PictureBox resultadoPtb;
		private Joaqs.UI.XpButton xpButton1;
		private Joaqs.UI.XpButton xpButton2;
		private System.Windows.Forms.SaveFileDialog salvarImagen;
		private System.Windows.Forms.PictureBox histogramaPtb;
		private Joaqs.UI.XpButton xpButton3;
		private System.Windows.Forms.Label label1;
		private System.Windows.Forms.Label label2;
		private System.Windows.Forms.PictureBox originalIconoPtb;
		private System.Windows.Forms.PictureBox histogramaIconoPtb;
		private System.Windows.Forms.PictureBox blancoynegroIconoPtb;
		private PrincipalFrm formularioPrincipal;
		public TransformaFrm(PrincipalFrm f)
		{
			InitializeComponent();

			// Primero asignamos al formulario actual el formulario donde se abre.
			this.formularioPrincipal = f;
			this.MdiParent = this.formularioPrincipal;
		}
		#region Codigo generado por .NET
		/// <summary>
		/// Limpiar los recursos que se estén utilizando.
		/// </summary>
		protected override void Dispose( bool disposing )
		{		if( disposing )
			{		if(components != null)
				{	components.Dispose();
				}
			}
			base.Dispose( disposing );
		}
		#region Código generado por el Diseñador de Windows Forms
		/// <summary>
		/// Método necesario para admitir el Diseñador. No se puede modificar
		/// el contenido del método con el editor de código.
		/// </summary>
		private void InitializeComponent()
```

```
        {
            System.Resources.ResourceManager resources = new
            System.Resources.ResourceManager(typeof(TransformaFrm));
            this.originalPtb = new System.Windows.Forms.PictureBox();
            this.abrirImagen = new System.Windows.Forms.OpenFileDialog();
            this.abrirBtn = new Joaqs.UI.XpButton();
            this.resultadoPtb = new System.Windows.Forms.PictureBox();
            this.xpButton1 = new Joaqs.UI.XpButton();
            this.xpButton2 = new Joaqs.UI.XpButton();
            this.salvarImagen = new System.Windows.Forms.SaveFileDialog();
            this.histogramaPtb = new System.Windows.Forms.PictureBox();
            this.xpButton3 = new Joaqs.UI.XpButton();
            this.label1 = new System.Windows.Forms.Label();
            this.originalIconoPtb = new System.Windows.Forms.PictureBox();
            this.histogramaIconoPtb = new System.Windows.Forms.PictureBox();
            this.blancoynegroIconoPtb = new
System.Windows.Forms.PictureBox();
            this.label2 = new System.Windows.Forms.Label();
            this.SuspendLayout();
            //
            // originalPtb
            //
            this.originalPtb.BackColor = System.Drawing.Color.White;
            this.originalPtb.BorderStyle =
System.Windows.Forms.BorderStyle.FixedSingle;
            this.originalPtb.Location = new System.Drawing.Point(16, 56);
            this.originalPtb.Name = "originalPtb";
            this.originalPtb.Size = new System.Drawing.Size(264, 300);
            this.originalPtb.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.StretchImage;
            this.originalPtb.TabIndex = 0;
            this.originalPtb.TabStop = false;
            //
            // abrirBtn
            //
            this.abrirBtn.Location = new System.Drawing.Point(88, 392);
            this.abrirBtn.Name = "abrirBtn";
            this.abrirBtn.Size = new System.Drawing.Size(100, 30);
            this.abrirBtn.TabIndex = 1;
            this.abrirBtn.Text = "Abrir imagen";
            this.abrirBtn.Click += new System.EventHandler(this.abrirBtn_Click);
            //
            // resultadoPtb
            //
            this.resultadoPtb.BackColor = System.Drawing.Color.White;
            this.resultadoPtb.BorderStyle =
System.Windows.Forms.BorderStyle.FixedSingle;
            this.resultadoPtb.Location = new System.Drawing.Point(480, 56);
            this.resultadoPtb.Name = "resultadoPtb";
            this.resultadoPtb.Size = new System.Drawing.Size(264, 300);
            this.resultadoPtb.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.StretchImage;
            this.resultadoPtb.TabIndex = 2;
            this.resultadoPtb.TabStop = false;
            //
            // xpButton1
            //
```

```
                        this.xpButton1.ImageAlign =
System.Drawing.ContentAlignment.MiddleRight;
                        this.xpButton1.Location = new System.Drawing.Point(304, 136);
                        this.xpButton1.Name = "xpButton1";
                        this.xpButton1.Size = new System.Drawing.Size(144, 32);
                        this.xpButton1.TabIndex = 3;
                        this.xpButton1.Text = "Convertir Imagen >>";
                        this.xpButton1.Click += new
System.EventHandler(this.xpButton1_Click);
                        //
                        // xpButton2
                        //
                        this.xpButton2.Location = new System.Drawing.Point(560, 392);
                        this.xpButton2.Name = "xpButton2";
                        this.xpButton2.Size = new System.Drawing.Size(100, 30);
                        this.xpButton2.TabIndex = 4;
                        this.xpButton2.Text = "Guardar Imagen";
                        this.xpButton2.Visible = false;
                        this.xpButton2.Click += new
System.EventHandler(this.xpButton2_Click);
                        //
                        // histogramaPtb
                        //
                        this.histogramaPtb.BackColor = System.Drawing.Color.White;
                        this.histogramaPtb.BorderStyle =
System.Windows.Forms.BorderStyle.FixedSingle;
                        this.histogramaPtb.Location = new System.Drawing.Point(480, 56);
                        this.histogramaPtb.Name = "histogramaPtb";
                        this.histogramaPtb.Size = new System.Drawing.Size(264, 300);
                        this.histogramaPtb.TabIndex = 5;
                        this.histogramaPtb.TabStop = false;
                        this.histogramaPtb.Visible = false;
                        //
                        // xpButton3
                        //
                        this.xpButton3.Location = new System.Drawing.Point(304, 256);
                        this.xpButton3.Name = "xpButton3";
                        this.xpButton3.Size = new System.Drawing.Size(144, 32);
                        this.xpButton3.TabIndex = 6;
                        this.xpButton3.Text = "Ver histograma >>";
                        this.xpButton3.Click += new
System.EventHandler(this.xpButton3_Click);
                        //
                        // label1
                        //
                        this.label1.Font = new System.Drawing.Font("Microsoft Sans Serif",
10F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((System.Byte)(0)));
                        this.label1.Location = new System.Drawing.Point(40, 32);
                        this.label1.Name = "label1";
                        this.label1.Size = new System.Drawing.Size(120, 16);
                        this.label1.TabIndex = 7;
                        this.label1.Text = "Imagen original";
                        //
                        // originalIconoPtb
                        //
                        this.originalIconoPtb.Image =
((System.Drawing.Image)(resources.GetObject("originalIconoPtb.Image")));
```

```
                this.originalIconoPtb.Location = new System.Drawing.Point(16, 32);
                this.originalIconoPtb.Name = "originalIconoPtb";
                this.originalIconoPtb.Size = new System.Drawing.Size(16, 16);
                this.originalIconoPtb.TabIndex = 8;
                this.originalIconoPtb.TabStop = false;
                //
                // histogramaIconoPtb
                //
                this.histogramaIconoPtb.Image =
((System.Drawing.Image)(resources.GetObject("histogramaIconoPtb.Image")));
                this.histogramaIconoPtb.Location = new System.Drawing.Point(480,
32);
                this.histogramaIconoPtb.Name = "histogramaIconoPtb";
                this.histogramaIconoPtb.Size = new System.Drawing.Size(16, 16);
                this.histogramaIconoPtb.TabIndex = 9;
                this.histogramaIconoPtb.TabStop = false;
                //
                // blancoynegroIconoPtb
                //
                this.blancoynegroIconoPtb.Image =
((System.Drawing.Image)(resources.GetObject("blancoynegroIconoPtb.Image")));
                this.blancoynegroIconoPtb.Location = new
System.Drawing.Point(480, 32);
                this.blancoynegroIconoPtb.Name = "blancoynegroIconoPtb";
                this.blancoynegroIconoPtb.Size = new System.Drawing.Size(16, 16);
                this.blancoynegroIconoPtb.TabIndex = 10;
                this.blancoynegroIconoPtb.TabStop = false;
                //
                // label2
                //
                this.label2.Font = new System.Drawing.Font("Microsoft Sans Serif",
10F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((System.Byte)(0)));
                this.label2.Location = new System.Drawing.Point(504, 32);
                this.label2.Name = "label2";
                this.label2.Size = new System.Drawing.Size(168, 16);
                this.label2.TabIndex = 11;
                this.label2.Text = "Imagen en blanco y negro";
                //
                // TransformaFrm
                //
                this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);
                this.BackColor = System.Drawing.SystemColors.Control;
                this.ClientSize = new System.Drawing.Size(770, 456);
                this.Controls.Add(this.label2);
                this.Controls.Add(this.blancoynegroIconoPtb);
                this.Controls.Add(this.histogramaIconoPtb);
                this.Controls.Add(this.originalIconoPtb);
                this.Controls.Add(this.label1);
                this.Controls.Add(this.xpButton3);
                this.Controls.Add(this.histogramaPtb);
                this.Controls.Add(this.xpButton2);
                this.Controls.Add(this.xpButton1);
                this.Controls.Add(this.resultadoPtb);
                this.Controls.Add(this.abrirBtn);
                this.Controls.Add(this.originalPtb);
                this.FormBorderStyle =
System.Windows.Forms.FormBorderStyle.FixedToolWindow;
```

```
                this.MaximizeBox = false;
                this.Name = "TransformaFrm";
                this.StartPosition =
System.Windows.Forms.FormStartPosition.CenterScreen;
                this.Text = "Pasar imagen a Blanco y Negro";
                this.ResumeLayout(false);
            }
            #endregion
            #endregion
            private void abrirBtn_Click(object sender, System.EventArgs e)
            {       abrirImagen.Filter = "Tipo de archivo (*.bmp; *.gif; *.jpg;
*.png)|*.bmp;*.gif;*.jpg;*.png|" + "All files (*.*)|*.*";
                if( abrirImagen.ShowDialog()==DialogResult.OK )
                {       try
                {  System.Windows.Forms.PictureBoxSizeMode modoVisualizacion;
                        string nombreFichero = abrirImagen.FileName;
                        Image imagen =
Image.FromFile(abrirImagen.FileName);
                        originalPtb.Image = imagen;
            // Si la imagen es mas pequeña que el cuadro de imagen, no le hacemos un
stretch
            if (imagen.Width <= originalPtb.Width && imagen.Height <=
originalPtb.Height)
                        {
                                modoVisualizacion =
System.Windows.Forms.PictureBoxSizeMode.CenterImage;
                                originalPtb.SizeMode = modoVisualizacion;
                                resultadoPtb.SizeMode = modoVisualizacion;
                        }
                        else
                        {
            modoVisualizacion =
System.Windows.Forms.PictureBoxSizeMode.StretchImage;
                                originalPtb.SizeMode = modoVisualizacion;
                        }
                    }
                    catch (Exception)
                    {
MessageBox.Show(this, "La imagen no se ha podido Cargar!!!", "Cargar Imagen",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                    }
                }
            }
            private void xpButton1_Click(object sender, System.EventArgs e)
            {       //Aqui va el tratamiento de la imagen
                if(this.originalPtb.Image!=null)
                {       xpButton2.Visible = true;
                    bool[][] pixels= new bool[originalPtb.Image.Width][];
                    for(int i=0; i<originalPtb.Image.Width; i++)
                    pixels[i] = new bool[originalPtb.Image.Height];
                    Bitmap copia = new Bitmap((Bitmap) this.originalPtb.Image);
                    TransformaImagen gris = new TransformaImagen(copia);
                    resultadoPtb.Image = null;
                    gris.Binarizar(pixels);
                    resultadoPtb.Visible = true;
                    resultadoPtb.Image = gris.Bitmap;
                    histogramaPtb.Visible = false;
                    label2.Text = "Imagen en blanco y negro";
```

```
                        histogramaIconoPtb.Visible = false;
                        blancoynegroIconoPtb.Visible = true;
                    }
                    else
MessageBox.Show(this, "Debe cargar una imagen antes de poder transformarla!!!", "Cargar
Imagen", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                }
                private void xpButton2_Click(object sender, System.EventArgs e)
                {
                    salvarImagen.Filter = "Archivo de mapa de bits (*.bmp)|*.bmp|"+
                        "Archivo de imagen (*.gif; *.jpg; *.png)|*.gif;*.jpg;*.png|" +
                            "All files (*.*)|*.*";
                        if( salvarImagen.ShowDialog()==DialogResult.OK)
                    {   try
                        {
        resultadoPtb.Image.Save(salvarImagen.FileName, ImageFormat.Bmp);
                        }
                        catch(Exception)
                        {
MessageBox.Show(this, "No se ha podido guardar la imagen!!!", "Guargar Imagen",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                        }
                    }
                }
                private void xpButton3_Click(object sender, System.EventArgs e)
                {   if(this.originalPtb.Image!=null)
                    {   // Ahora dibujamos el histograma de la imagen original
                        //************************************
                        Histograma histograma = new Histograma();
                        histograma.setImage( new Bitmap( this.originalPtb.Image ) );
                        histograma.calculaHistograma( histogramaPtb );
                        histogramaPtb.Visible = true;
                        resultadoPtb.Visible = false;
                        label2.Text = "Histograma";
                        blancoynegroIconoPtb.Visible = false;
                        histogramaIconoPtb.Visible = true;
                    }
                    else
MessageBox.Show(this, "Debe cargar una imagen antes de ver su histograma!!!", "Cargar
Imagen", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                }
        }
}
```

FORMULARIO REDUCIR IMAGEN

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Drawing.Imaging;
namespace RecFirma.Formularios
{   public class ReducirImagenFrm : System.Windows.Forms.Form
    {   private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.TextBox anchoImagenTb;
        private System.Windows.Forms.TextBox altoImagenTb;
        private System.ComponentModel.Container components = null;
```

```
		private Joaqs.UI.XpButton abrirImagenBtn;
		private System.Windows.Forms.PictureBox originalPtb;
		private Joaqs.UI.XpButton redimensionarBtn;
		private Joaqs.UI.XpButton guardarBtn;
		private System.Windows.Forms.OpenFileDialog abrirImagenOfd;
		private System.Windows.Forms.SaveFileDialog salvarSfd;
		private System.Windows.Forms.GroupBox groupBox1;
		private System.Windows.Forms.TrackBar barraReduccionTcb;
		private System.Windows.Forms.GroupBox groupBox2;
		private System.Windows.Forms.RadioButton redimensionConstanteRbt;
		private System.Windows.Forms.RadioButton redimensionManualRbt;
		private PrincipalFrm formularioPrincipal;
		// Variables para el cálculo de la redimensión de la imagen
		private int anchoImagen;
		private int altoImagen;
		private int posicionScrollAnterior;
		private int porcentajeAlto, porcentajeAncho;
		private System.Windows.Forms.GroupBox groupBox3;
		private System.Windows.Forms.GroupBox groupBox4;
		private System.Windows.Forms.PictureBox pictureBox1;
		private System.Windows.Forms.PictureBox pictureBox2;
		private System.Windows.Forms.PictureBox pictureBox3;
		private System.Windows.Forms.PictureBox pictureBox4;
		private Image imagenVirgen;
		public ReducirImagenFrm(PrincipalFrm f)
		{
			InitializeComponent();
			// Primero asignamos al formulario actual el formulario donde se abre.
			this.formularioPrincipal = f;
			this.MdiParent = this.formularioPrincipal;
		}
		#region Codigo generado por .NET
		/// <summary>
		/// Limpiar los recursos que se estén utilizando.
		/// </summary>
		protected override void Dispose( bool disposing )
		{	if( disposing )
			{	if(components != null)
				{	components.Dispose();
				}
			}
			base.Dispose( disposing );
		}
		#region Código generado por el Diseñador de Windows Forms
		/// <summary>
		/// Método necesario para admitir el Diseñador. No se puede modificar
		/// el contenido del método con el editor de código.
		/// </summary>
		private void InitializeComponent()
		{	System.Resources.ResourceManager resources = new
System.Resources.ResourceManager(typeof(ReducirImagenFrm));
			this.originalPtb = new System.Windows.Forms.PictureBox();
			this.anchoImagenTb = new System.Windows.Forms.TextBox();
			this.label1 = new System.Windows.Forms.Label();
			this.label2 = new System.Windows.Forms.Label();
			this.altoImagenTb = new System.Windows.Forms.TextBox();
			this.abrirImagenBtn = new Joaqs.UI.XpButton();
			this.abrirImagenOfd = new System.Windows.Forms.OpenFileDialog();
```

```
                this.redimensionarBtn = new Joaqs.UI.XpButton();
                this.guardarBtn = new Joaqs.UI.XpButton();
                this.salvarSfd = new System.Windows.Forms.SaveFileDialog();
                this.barraReduccionTcb = new System.Windows.Forms.TrackBar();
                this.groupBox1 = new System.Windows.Forms.GroupBox();
                this.pictureBox4 = new System.Windows.Forms.PictureBox();
                this.pictureBox3 = new System.Windows.Forms.PictureBox();
                this.groupBox2 = new System.Windows.Forms.GroupBox();
                this.redimensionManualRbt = new
System.Windows.Forms.RadioButton();
                this.redimensionConstanteRbt = new
System.Windows.Forms.RadioButton();
                this.groupBox3 = new System.Windows.Forms.GroupBox();
                this.groupBox4 = new System.Windows.Forms.GroupBox();
                this.pictureBox2 = new System.Windows.Forms.PictureBox();
                this.pictureBox1 = new System.Windows.Forms.PictureBox();

        ((System.ComponentModel.ISupportInitialize)(this.barraReduccionTcb)).BeginInit();
                this.groupBox1.SuspendLayout();
                this.groupBox2.SuspendLayout();
                this.groupBox3.SuspendLayout();
                this.groupBox4.SuspendLayout();
                this.SuspendLayout();
                //
                // originalPtb
                //
                this.originalPtb.BackColor = System.Drawing.Color.White;
                this.originalPtb.BorderStyle =
System.Windows.Forms.BorderStyle.FixedSingle;
                this.originalPtb.Location = new System.Drawing.Point(16, 24);
                this.originalPtb.Name = "originalPtb";
                this.originalPtb.Size = new System.Drawing.Size(300, 300);
                this.originalPtb.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.CenterImage;
                this.originalPtb.TabIndex = 0;
                this.originalPtb.TabStop = false;
                //
                // anchoImagenTb
                //
                this.anchoImagenTb.Enabled = false;
                this.anchoImagenTb.Location = new System.Drawing.Point(152, 32);
                this.anchoImagenTb.MaxLength = 5;
                this.anchoImagenTb.Name = "anchoImagenTb";
                this.anchoImagenTb.Size = new System.Drawing.Size(88, 21);
                this.anchoImagenTb.TabIndex = 1;
                this.anchoImagenTb.Text = "";
                //
                // label1
                //
                this.label1.Font = new System.Drawing.Font("Microsoft Sans Serif",
8.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((System.Byte)(0)));
                this.label1.Location = new System.Drawing.Point(40, 37);
                this.label1.Name = "label1";
                this.label1.Size = new System.Drawing.Size(104, 16);
                this.label1.TabIndex = 2;
                this.label1.Text = "Ancho de la imagen";
                //
```

```
                        // label2
                        //
                        this.label2.Font = new System.Drawing.Font("Microsoft Sans Serif",
8.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((System.Byte)(0)));
                        this.label2.Location = new System.Drawing.Point(40, 77);
                        this.label2.Name = "label2";
                        this.label2.Size = new System.Drawing.Size(104, 16);
                        this.label2.TabIndex = 3;
                        this.label2.Text = "Alto de la imagen";
                        //
                        // altoImagenTb
                        //
                        this.altoImagenTb.Enabled = false;
                        this.altoImagenTb.Location = new System.Drawing.Point(152, 72);
                        this.altoImagenTb.MaxLength = 5;
                        this.altoImagenTb.Name = "altoImagenTb";
                        this.altoImagenTb.Size = new System.Drawing.Size(88, 21);
                        this.altoImagenTb.TabIndex = 4;
                        this.altoImagenTb.Text = "";
                        //
                        // abrirImagenBtn
                        //
                        this.abrirImagenBtn.Location = new System.Drawing.Point(24, 336);
                        this.abrirImagenBtn.Name = "abrirImagenBtn";
                        this.abrirImagenBtn.Size = new System.Drawing.Size(120, 32);
                        this.abrirImagenBtn.TabIndex = 5;
                        this.abrirImagenBtn.Text = "Abrir Imagen";
                        this.abrirImagenBtn.Click += new
System.EventHandler(this.abrirImagenBtn_Click);
                        //
                        // redimensionarBtn
                        //
                        this.redimensionarBtn.Location = new System.Drawing.Point(376,
352);
                        this.redimensionarBtn.Name = "redimensionarBtn";
                        this.redimensionarBtn.Size = new System.Drawing.Size(112, 32);
                        this.redimensionarBtn.TabIndex = 6;
                        this.redimensionarBtn.Text = "Redimensionar";
                        this.redimensionarBtn.Visible = false;
                        this.redimensionarBtn.Click += new
System.EventHandler(this.redimensionarBtn_Click);
                        //
                        // guardarBtn
                        //
                        this.guardarBtn.Location = new System.Drawing.Point(512, 352);
                        this.guardarBtn.Name = "guardarBtn";
                        this.guardarBtn.Size = new System.Drawing.Size(104, 32);
                        this.guardarBtn.TabIndex = 7;
                        this.guardarBtn.Text = "Guardar";
                        this.guardarBtn.Visible = false;
                        this.guardarBtn.Click += new
System.EventHandler(this.guardarBtn_Click);
                        //
                        // barraReduccionTcb
                        //
                        this.barraReduccionTcb.Cursor =
System.Windows.Forms.Cursors.Hand;
```

```
                this.barraReduccionTcb.LargeChange = 3;
                this.barraReduccionTcb.Location = new System.Drawing.Point(16,
16);
                this.barraReduccionTcb.Maximum = 20;
                this.barraReduccionTcb.Name = "barraReduccionTcb";
                this.barraReduccionTcb.Size = new System.Drawing.Size(224, 45);
                this.barraReduccionTcb.TabIndex = 8;
                this.barraReduccionTcb.Scroll += new
System.EventHandler(this.barraReduccionTcb_Scroll);
                //
                // groupBox1
                //
                this.groupBox1.Controls.Add(this.pictureBox4);
                this.groupBox1.Controls.Add(this.pictureBox3);
                this.groupBox1.Controls.Add(this.altoImagenTb);
                this.groupBox1.Controls.Add(this.anchoImagenTb);
                this.groupBox1.Controls.Add(this.label1);
                this.groupBox1.Controls.Add(this.label2);
                this.groupBox1.Font = new System.Drawing.Font("Microsoft Sans
Serif", 9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((System.Byte)(0)));
                this.groupBox1.Location = new System.Drawing.Point(368, 16);
                this.groupBox1.Name = "groupBox1";
                this.groupBox1.Size = new System.Drawing.Size(256, 104);
                this.groupBox1.TabIndex = 9;
                this.groupBox1.TabStop = false;
                this.groupBox1.Text = "Dimensiones de la imagen";
                //
                // pictureBox4
                //
                this.pictureBox4.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox4.Image")));
                this.pictureBox4.Location = new System.Drawing.Point(16, 77);
                this.pictureBox4.Name = "pictureBox4";
                this.pictureBox4.Size = new System.Drawing.Size(24, 16);
                this.pictureBox4.TabIndex = 6;
                this.pictureBox4.TabStop = false;
                //
                // pictureBox3
                //
                this.pictureBox3.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox3.Image")));
                this.pictureBox3.Location = new System.Drawing.Point(16, 32);
                this.pictureBox3.Name = "pictureBox3";
                this.pictureBox3.Size = new System.Drawing.Size(16, 24);
                this.pictureBox3.TabIndex = 5;
                this.pictureBox3.TabStop = false;
                //
                // groupBox2
                //
                this.groupBox2.Controls.Add(this.redimensionManualRbt);
                this.groupBox2.Controls.Add(this.redimensionConstanteRbt);
                this.groupBox2.Font = new System.Drawing.Font("Microsoft Sans
Serif", 9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((System.Byte)(0)));
                this.groupBox2.Location = new System.Drawing.Point(368, 136);
                this.groupBox2.Name = "groupBox2";
                this.groupBox2.Size = new System.Drawing.Size(256, 100);
```

```
                        this.groupBox2.TabIndex = 10;
                        this.groupBox2.TabStop = false;
                        this.groupBox2.Text = "Tipo de redimensión";
                        this.groupBox2.Visible = false;
                        //
                        // redimensionManualRbt
                        //
                        this.redimensionManualRbt.Location = new System.Drawing.Point(16,
64);
                        this.redimensionManualRbt.Name = "redimensionManualRbt";
                        this.redimensionManualRbt.Size = new System.Drawing.Size(144,
24);
                        this.redimensionManualRbt.TabIndex = 1;
                        this.redimensionManualRbt.Text = "Redimension Manual";
                        this.redimensionManualRbt.CheckedChanged += new
System.EventHandler(this.redimensionManualRbt_CheckedChanged);
                        //
                        // redimensionConstanteRbt
                        //
                        this.redimensionConstanteRbt.Checked = true;
                this.redimensionConstanteRbt.Location = new System.Drawing.Point(16, 32);
                        this.redimensionConstanteRbt.Name = "redimensionConstanteRbt";
                        this.redimensionConstanteRbt.Size = new System.Drawing.Size(152,
24);
                        this.redimensionConstanteRbt.TabIndex = 0;
                        this.redimensionConstanteRbt.TabStop = true;
                        this.redimensionConstanteRbt.Text = "Redimensión constante";
                        this.redimensionConstanteRbt.CheckedChanged += new
System.EventHandler(this.redimensionConstanteRbt_CheckedChanged);
                        //
                        // groupBox3
                        //
                        this.groupBox3.Controls.Add(this.originalPtb);
                        this.groupBox3.Controls.Add(this.abrirImagenBtn);
                        this.groupBox3.Font = new System.Drawing.Font("Microsoft Sans
Serif", 9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((System.Byte)(0)));
                        this.groupBox3.Location = new System.Drawing.Point(16, 16);
                        this.groupBox3.Name = "groupBox3";
                        this.groupBox3.Size = new System.Drawing.Size(336, 384);
                        this.groupBox3.TabIndex = 11;
                        this.groupBox3.TabStop = false;
                        this.groupBox3.Text = "Imagen a redimensionar";
                        //
                        // groupBox4
                        //
                        this.groupBox4.Controls.Add(this.pictureBox2);
                        this.groupBox4.Controls.Add(this.pictureBox1);
                        this.groupBox4.Controls.Add(this.barraReduccionTcb);
                        this.groupBox4.Location = new System.Drawing.Point(368, 256);
                        this.groupBox4.Name = "groupBox4";
                        this.groupBox4.Size = new System.Drawing.Size(256, 64);
                        this.groupBox4.TabIndex = 12;
                        this.groupBox4.TabStop = false;
                        this.groupBox4.Text = "Barra de redimensión";
                        this.groupBox4.Visible = false;
                        //
                        // pictureBox2
```

```
                        //
                        this.pictureBox2.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox2.Image")));
                        this.pictureBox2.Location = new System.Drawing.Point(232, 32);
                        this.pictureBox2.Name = "pictureBox2";
                        this.pictureBox2.Size = new System.Drawing.Size(16, 16);
                        this.pictureBox2.TabIndex = 10;
                        this.pictureBox2.TabStop = false;
                        this.pictureBox2.Click += new
System.EventHandler(this.pictureBox2_Click);
                        //
                        // pictureBox1
                        //
                        this.pictureBox1.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox1.Image")));
                        this.pictureBox1.Location = new System.Drawing.Point(8, 32);
                        this.pictureBox1.Name = "pictureBox1";
                        this.pictureBox1.Size = new System.Drawing.Size(16, 16);
                        this.pictureBox1.TabIndex = 9;
                        this.pictureBox1.TabStop = false;
                        this.pictureBox1.Click += new
System.EventHandler(this.pictureBox1_Click);
                        //
                        // ReducirImagenFrm
                        //
                        this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);
                        this.BackColor = System.Drawing.SystemColors.ControlLight;
                        this.ClientSize = new System.Drawing.Size(640, 430);
                        this.Controls.Add(this.groupBox4);
                        this.Controls.Add(this.groupBox3);
                        this.Controls.Add(this.groupBox2);
                        this.Controls.Add(this.groupBox1);
                        this.Controls.Add(this.guardarBtn);
                        this.Controls.Add(this.redimensionarBtn);
                        this.FormBorderStyle =
System.Windows.Forms.FormBorderStyle.FixedToolWindow;
                        this.Icon =
((System.Drawing.Icon)(resources.GetObject("$this.Icon")));
                        this.Name = "ReducirImagenFrm";
                        this.StartPosition =
System.Windows.Forms.FormStartPosition.CenterScreen;
                        this.Text = "Redimensionar tamaño de imagen";

        ((System.ComponentModel.ISupportInitialize)(this.barraReduccionTcb)).EndInit();
                        this.groupBox1.ResumeLayout(false);
                        this.groupBox2.ResumeLayout(false);
                        this.groupBox3.ResumeLayout(false);
                        this.groupBox4.ResumeLayout(false);
                        this.ResumeLayout(false);
                }
                #endregion
                #endregion
                private void abrirImagenBtn_Click(object sender, System.EventArgs e)
                {       abrirImagenOfd.Filter = "Tipo de archivo (*.bmp; *.gif; *.jpg;
*.png)|*.bmp;*.gif;*.jpg;*.png|" + "All files (*.*)|*.*";
                        if( abrirImagenOfd.ShowDialog()==DialogResult.OK )
                {       try
                        {  string nombreFichero = abrirImagenOfd.FileName;
```

```
                                imagenVirgen = Image.FromFile(abrirImagenOfd.FileName);
                                originalPtb.Image = imagenVirgen;
                                        anchoImagen = originalPtb.Image.Width;
                                        altoImagen = originalPtb.Image.Height;
                        // Calculamos el porcentaje de alto y ancho que se va a modificar la
imagen
                                        porcentajeAlto = (altoImagen * 5)/100;
                                        porcentajeAncho = (anchoImagen * 5)/100;
                // Si la imagen es mas pequeña que el cuadro de imagen, no le hacemos un
stretch
        if (imagenVirgen.Width <= originalPtb.Width && imagenVirgen.Height <=
originalPtb.Height)
                                        {       int aux = (originalPtb.Width /
imagenVirgen.Width);
                                                if(aux >= 6)
                                                {
                                        porcentajeAncho = (300 - imagenVirgen.Width)/20;
                                        porcentajeAlto = (300 - imagenVirgen.Height)/20;
                                                        barraReduccionTcb.Value = 3;
                                                }
                                                else
                                                        barraReduccionTcb.Value = 10;
// Establecemos el valor del scroll anterior para conocer si se desplaza a la derecha o a la
izquierda
                                        posicionScrollAnterior = barraReduccionTcb.Value;
                                        }
                                        else
                                        {
                                                barraReduccionTcb.Value = 20;
// Establecemos el valor del scroll anterior para conocer si se desplaza a la derecha o a la
izquierda
                                                posicionScrollAnterior =
barraReduccionTcb.Value;
                                        }
                                        // Escribimos en pantalla el ancho y el alto de la
imagen
                                        anchoImagenTb.Text =
imagenVirgen.Width.ToString();
                                        altoImagenTb.Text = imagenVirgen.Height.ToString();

                                        this.groupBox2.Visible = true;
                                        this.groupBox4.Visible = true;
                                        this.guardarBtn.Visible = true;
                                }
                                catch (Exception)
                                {
                MessageBox.Show(this, "La imagen no se ha podido Cargar!!!", "Cargar
Imagen");
                                }
                        }
                }
                private void redimensionarBtn_Click(object sender, System.EventArgs e)
                {       redimensionaImagen();
                }

                private void guardarBtn_Click(object sender, System.EventArgs e)
                {
                        salvarSfd.Filter = "Archivo de mapa de bits (*.bmp)|*.bmp";
```

```
                        if( salvarSfd.ShowDialog() == DialogResult.OK )
                        {       bool salvada = true;
                                try
                                {
                                originalPtb.Image.Save(salvarSfd.FileName,
ImageFormat.Bmp);
                                }
                                catch(Exception)
                                {
                                        salvada = false;
                                }
                                        if(!salvada)
                                {
MessageBox.Show(this, "La imágen no se ha podido salvar", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                                }
                        }
                }
        private void redimensionManualRbt_CheckedChanged(object sender,
System.EventArgs e)
                {       groupBox4.Visible = false;
                        anchoImagenTb.Enabled = true;
                        altoImagenTb.Enabled = true;
                        redimensionarBtn.Visible = true;
                }
private void redimensionConstanteRbt_CheckedChanged(object sender, System.EventArgs e)
                {
                        groupBox4.Visible = true;
                        anchoImagenTb.Enabled = false;
                        altoImagenTb.Enabled = false;
                        redimensionarBtn.Visible = false;
                }
                private void barraReduccionTcb_Scroll(object sender, System.EventArgs e)
                {       int altoImagen = this.altoImagen, anchoImagen = this.anchoImagen;
int modifAncho = (porcentajeAncho * Math.Abs(posicionScrollAnterior -
barraReduccionTcb.Value));
int modifAlto = (porcentajeAlto * Math.Abs(posicionScrollAnterior -
barraReduccionTcb.Value));
                        // Calculamos las nuevas dimensiones;
                        if(posicionScrollAnterior > barraReduccionTcb.Value)
                        {       anchoImagen =  anchoImagen - modifAncho;
                                altoImagen = altoImagen - modifAlto;
                        }
                        else
                        {       anchoImagen =  anchoImagen + modifAncho;
                                altoImagen = altoImagen + modifAlto;
                        }
                        posicionScrollAnterior = barraReduccionTcb.Value;
                        // Si las dimensiones no son menores que 0
                        if(anchoImagen > 0 && altoImagen > 0)
                        {       // Primero redimensionamos la imagen con la nueva
dimensión
                                this.anchoImagen = anchoImagen;
                                this.altoImagen = altoImagen;
                                // Y después establecemos mostramos ese valor en el
TextBox
                                anchoImagenTb.Text = anchoImagen.ToString();
                                altoImagenTb.Text = altoImagen.ToString();
```

```
                        redimensionaImagen();
                    }
                    else
                    {       barraReduccionTcb.Value = posicionScrollAnterior;
                    }
                }

                private void redimensionaImagen()
                {       Bitmap result = new Bitmap( imagenVirgen,
Int32.Parse(this.anchoImagenTb.Text) , Int32.Parse(this.altoImagenTb.Text) );
                    using( Graphics g = Graphics.FromImage( (Image) result ) )
                        g.DrawImage( imagenVirgen, 0, 0,
float.Parse(this.anchoImagenTb.Text), float.Parse(this.altoImagenTb.Text) );
                    this.originalPtb.Image = result;
                }
                private void pictureBox2_Click(object sender, System.EventArgs e)
                {       if(barraReduccionTcb.Value < barraReduccionTcb.Maximum)
                    {       barraReduccionTcb.Value = barraReduccionTcb.Value + 1;
                        barraReduccionTcb_Scroll(sender, e);
                    }
                }
                private void pictureBox1_Click(object sender, System.EventArgs e)
                {
                    if(barraReduccionTcb.Value > barraReduccionTcb.Minimum)
                    {       barraReduccionTcb.Value = barraReduccionTcb.Value - 1;
                        barraReduccionTcb_Scroll(sender, e);
                    }
                }

        }
}
```

FORMULARIO FIRMAS

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using RecFirma.Clases;
namespace RecFirma.Formularios
{
        /// <summary>
        /// Descripción breve de Firmar.
        /// </summary>
        public class FirmarFrm : System.Windows.Forms.Form
        {
                #region atributos generados por .NET
    private System.ComponentModel.Container components = null;
    public static Controles_RecFirma.Signer signer;
                private Joaqs.UI.XpButton aceptarBtn;
                private Joaqs.UI.XpButton cancelarBtn;
                private System.Windows.Forms.LinkLabel abrirLlbl;
                private System.Windows.Forms.LinkLabel guardarLlbl;
                private System.Windows.Forms.OpenFileDialog abrirOfd;
                #endregion
                private PrincipalFrm formularioPrincipal;
                private Form formularioAnterior;
```

```
        private System.Windows.Forms.SaveFileDialog salvarSfd;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.PictureBox pictureBox1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.PictureBox pictureBox2;
        private System.Windows.Forms.GroupBox groupBox1;
        private System.Windows.Forms.LinkLabel buscarLbl;
        private System.Windows.Forms.LinkLabel linkLabel1;
        private System.Windows.Forms.LinkLabel limpiarLlbl;
        private System.Windows.Forms.GroupBox groupBox2;
        private System.Windows.Forms.PictureBox pictureBox3;
        private System.Windows.Forms.PictureBox pictureBox4;
        private System.Windows.Forms.PictureBox pictureBox5;
        private System.Windows.Forms.PictureBox pictureBox6;
        private System.Windows.Forms.PictureBox pictureBox7;
    private Firmable consumidorFirma;
    public static bool CambioImagen;
    public FirmarFrm(PrincipalFrm f, Form formularioAnterior, Firmable consumidorFirma)
    {
        InitializeComponent();

        // Primero asignamos al formulario actual el formulario donde se abre.
        this.formularioPrincipal = f;
        this.MdiParent = this.formularioPrincipal;
        this.formularioAnterior = formularioAnterior;
        this.consumidorFirma = consumidorFirma;
    }
        #region Codigo generado por .NET
        /// <summary>
        /// Limpiar los recursos que se estén utilizando.
        /// </summary>
        protected override void Dispose( bool disposing )
        {       if( disposing )
                {       if(components != null)
                        {       components.Dispose();
                        }
                }
                base.Dispose( disposing );
        }
        #region Código generado por el Diseñador de Windows Forms
        /// <summary>
        /// Método necesario para admitir el Diseñador. No se puede modificar
        /// el contenido del método con el editor de código.
        /// </summary>
        private void InitializeComponent()
        {
    System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(FirmarFrm));
        signer = new Controles_RecFirma.Signer();
        this.aceptarBtn = new Joaqs.UI.XpButton();
        this.cancelarBtn = new Joaqs.UI.XpButton();
        this.abrirLlbl = new System.Windows.Forms.LinkLabel();
        this.guardarLlbl = new System.Windows.Forms.LinkLabel();
        this.abrirOfd = new System.Windows.Forms.OpenFileDialog();
        this.salvarSfd = new System.Windows.Forms.SaveFileDialog();
        this.label1 = new System.Windows.Forms.Label();
        this.pictureBox1 = new System.Windows.Forms.PictureBox();
        this.label2 = new System.Windows.Forms.Label();
```

```
            this.pictureBox2 = new System.Windows.Forms.PictureBox();
            this.groupBox1 = new System.Windows.Forms.GroupBox();
            this.pictureBox5 = new System.Windows.Forms.PictureBox();
            this.pictureBox4 = new System.Windows.Forms.PictureBox();
            this.buscarLbl = new System.Windows.Forms.LinkLabel();
            this.pictureBox3 = new System.Windows.Forms.PictureBox();
            this.linkLabel1 = new System.Windows.Forms.LinkLabel();
            this.limpiarLlbl = new System.Windows.Forms.LinkLabel();
            this.groupBox2 = new System.Windows.Forms.GroupBox();
            this.pictureBox7 = new System.Windows.Forms.PictureBox();
            this.pictureBox6 = new System.Windows.Forms.PictureBox();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox2)).BeginInit();
            this.groupBox1.SuspendLayout();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox5)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox4)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox3)).BeginInit();
            this.groupBox2.SuspendLayout();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox7)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox6)).BeginInit();
            this.SuspendLayout();
            //
            // signer
            //
            signer.Location = new System.Drawing.Point(176, 32);
            signer.Name = "signer";
            signer.Size = new System.Drawing.Size(305, 305);
            signer.TabIndex = 1;
            //
            // aceptarBtn
            //
            this.aceptarBtn.Font = new System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.aceptarBtn.Location = new System.Drawing.Point(184, 360);
            this.aceptarBtn.Name = "aceptarBtn";
            this.aceptarBtn.Size = new System.Drawing.Size(104, 32);
            this.aceptarBtn.TabIndex = 2;
            this.aceptarBtn.Text = "Aceptar";
            this.aceptarBtn.Click += new System.EventHandler(this.aceptarBtn_Click);
            //
            // cancelarBtn
            //
            this.cancelarBtn.Font = new System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.cancelarBtn.Location = new System.Drawing.Point(376, 360);
            this.cancelarBtn.Name = "cancelarBtn";
            this.cancelarBtn.Size = new System.Drawing.Size(104, 32);
            this.cancelarBtn.TabIndex = 3;
            this.cancelarBtn.Text = "Salir";
            this.cancelarBtn.Click += new System.EventHandler(this.cancelarBtn_Click);
            //
            // abrirLlbl
            //
            this.abrirLlbl.Font = new System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.abrirLlbl.LinkColor = System.Drawing.Color.RoyalBlue;
            this.abrirLlbl.Location = new System.Drawing.Point(32, 228);
            this.abrirLlbl.Name = "abrirLlbl";
```

```
            this.abrirLlbl.Size = new System.Drawing.Size(112, 16);
            this.abrirLlbl.TabIndex = 4;
            this.abrirLlbl.TabStop = true;
            this.abrirLlbl.Text = "Abrir desde archivo";
            this.abrirLlbl.Click += new System.EventHandler(this.abrirLlbl_Click);
            //
            // guardarLlbl
            //
            this.guardarLlbl.Font = new System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.guardarLlbl.LinkColor = System.Drawing.Color.RoyalBlue;
            this.guardarLlbl.Location = new System.Drawing.Point(32, 192);
            this.guardarLlbl.Name = "guardarLlbl";
            this.guardarLlbl.Size = new System.Drawing.Size(100, 16);
            this.guardarLlbl.TabIndex = 6;
            this.guardarLlbl.TabStop = true;
            this.guardarLlbl.Text = "Exportar firma";
            this.guardarLlbl.Click += new System.EventHandler(this.guardarLlbl_Click);
            //
            // label1
            //
            this.label1.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label1.Location = new System.Drawing.Point(208, 16);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(48, 16);
            this.label1.TabIndex = 7;
            this.label1.Text = "Firma";
            //
            // pictureBox1
            //
            this.pictureBox1.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox1.Image")));
            this.pictureBox1.Location = new System.Drawing.Point(184, 16);
            this.pictureBox1.Name = "pictureBox1";
            this.pictureBox1.Size = new System.Drawing.Size(16, 16);
            this.pictureBox1.TabIndex = 8;
            this.pictureBox1.TabStop = false;
            //
            // label2
            //
            this.label2.Font = new System.Drawing.Font("Microsoft Sans Serif", 8.99F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label2.Location = new System.Drawing.Point(8, 24);
            this.label2.Name = "label2";
            this.label2.Size = new System.Drawing.Size(128, 63);
            this.label2.TabIndex = 9;
            this.label2.Text = "     Puede cargarla desde un archivo o desde la base de datos";
            //
            // pictureBox2
            //
            this.pictureBox2.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox2.Image")));
            this.pictureBox2.Location = new System.Drawing.Point(8, 24);
            this.pictureBox2.Name = "pictureBox2";
            this.pictureBox2.Size = new System.Drawing.Size(16, 16);
            this.pictureBox2.TabIndex = 0;
            this.pictureBox2.TabStop = false;
```

```
            //
            // groupBox1
            //
            this.groupBox1.Controls.Add(this.pictureBox5);
            this.groupBox1.Controls.Add(this.pictureBox4);
            this.groupBox1.Controls.Add(this.buscarLbl);
            this.groupBox1.Controls.Add(this.pictureBox2);
            this.groupBox1.Controls.Add(this.abrirLlbl);
            this.groupBox1.Controls.Add(this.guardarLlbl);
            this.groupBox1.Controls.Add(this.label2);
            this.groupBox1.Controls.Add(this.pictureBox3);
            this.groupBox1.Location = new System.Drawing.Point(16, 16);
            this.groupBox1.Name = "groupBox1";
            this.groupBox1.Size = new System.Drawing.Size(152, 272);
            this.groupBox1.TabIndex = 10;
            this.groupBox1.TabStop = false;
            this.groupBox1.Text = "Opciones";
            //
            // pictureBox5
            //
            this.pictureBox5.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox5.Image")));
            this.pictureBox5.Location = new System.Drawing.Point(11, 189);
            this.pictureBox5.Name = "pictureBox5";
            this.pictureBox5.Size = new System.Drawing.Size(16, 16);
            this.pictureBox5.TabIndex = 14;
            this.pictureBox5.TabStop = false;
            //
            // pictureBox4
            //
            this.pictureBox4.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox4.Image")));
            this.pictureBox4.Location = new System.Drawing.Point(11, 227);
            this.pictureBox4.Name = "pictureBox4";
            this.pictureBox4.Size = new System.Drawing.Size(16, 16);
            this.pictureBox4.TabIndex = 13;
            this.pictureBox4.TabStop = false;
            //
            // buscarLbl
            //
            this.buscarLbl.Font = new System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.buscarLbl.LinkColor = System.Drawing.Color.RoyalBlue;
            this.buscarLbl.Location = new System.Drawing.Point(32, 157);
            this.buscarLbl.Name = "buscarLbl";
            this.buscarLbl.Size = new System.Drawing.Size(88, 16);
            this.buscarLbl.TabIndex = 10;
            this.buscarLbl.TabStop = true;
            this.buscarLbl.Text = "Abrir desde BD";
            this.buscarLbl.LinkClicked += new
System.Windows.Forms.LinkLabelLinkClickedEventHandler(this.buscarLbl_LinkClicked);
            //
            // pictureBox3
            //
            this.pictureBox3.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox3.Image")));
            this.pictureBox3.Location = new System.Drawing.Point(11, 157);
            this.pictureBox3.Name = "pictureBox3";
```

```
            this.pictureBox3.Size = new System.Drawing.Size(16, 16);
            this.pictureBox3.TabIndex = 12;
            this.pictureBox3.TabStop = false;
            //
            // linkLabel1
            //
            this.linkLabel1.Font = new System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.linkLabel1.LinkColor = System.Drawing.Color.RoyalBlue;
            this.linkLabel1.Location = new System.Drawing.Point(32, 56);
            this.linkLabel1.Name = "linkLabel1";
            this.linkLabel1.Size = new System.Drawing.Size(104, 16);
            this.linkLabel1.TabIndex = 6;
            this.linkLabel1.TabStop = true;
            this.linkLabel1.Text = "Borrar último trazo";
            this.linkLabel1.LinkClicked += new
System.Windows.Forms.LinkLabelLinkClickedEventHandler(this.linkLabel1_LinkClicked);
            //
            // limpiarLlbl
            //
            this.limpiarLlbl.Font = new System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.limpiarLlbl.LinkColor = System.Drawing.Color.RoyalBlue;
            this.limpiarLlbl.Location = new System.Drawing.Point(32, 24);
            this.limpiarLlbl.Name = "limpiarLlbl";
            this.limpiarLlbl.Size = new System.Drawing.Size(80, 16);
            this.limpiarLlbl.TabIndex = 5;
            this.limpiarLlbl.TabStop = true;
            this.limpiarLlbl.Text = "Limpiar firma";
            this.limpiarLlbl.Click += new System.EventHandler(this.limpiarLlbl_Click);
            //
            // groupBox2
            //
            this.groupBox2.Controls.Add(this.pictureBox7);
            this.groupBox2.Controls.Add(this.pictureBox6);
            this.groupBox2.Controls.Add(this.linkLabel1);
            this.groupBox2.Controls.Add(this.limpiarLlbl);
            this.groupBox2.Location = new System.Drawing.Point(16, 296);
            this.groupBox2.Name = "groupBox2";
            this.groupBox2.Size = new System.Drawing.Size(152, 88);
            this.groupBox2.TabIndex = 11;
            this.groupBox2.TabStop = false;
            this.groupBox2.Text = "Opciones de borrado";
            //
            // pictureBox7
            //
            this.pictureBox7.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox7.Image")));
            this.pictureBox7.Location = new System.Drawing.Point(8, 56);
            this.pictureBox7.Name = "pictureBox7";
            this.pictureBox7.Size = new System.Drawing.Size(16, 16);
            this.pictureBox7.TabIndex = 16;
            this.pictureBox7.TabStop = false;
            //
            // pictureBox6
            //
            this.pictureBox6.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox6.Image")));
```

```
		this.pictureBox6.Location = new System.Drawing.Point(8, 24);
		this.pictureBox6.Name = "pictureBox6";
		this.pictureBox6.Size = new System.Drawing.Size(16, 16);
		this.pictureBox6.TabIndex = 15;
		this.pictureBox6.TabStop = false;
		//
		// FirmarFrm
		//
		this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);
		this.ClientSize = new System.Drawing.Size(504, 406);
		this.Controls.Add(this.groupBox1);
		this.Controls.Add(this.pictureBox1);
		this.Controls.Add(this.label1);
		this.Controls.Add(this.cancelarBtn);
		this.Controls.Add(this.aceptarBtn);
		this.Controls.Add(signer);
		this.Controls.Add(this.groupBox2);
		this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedToolWindow;
		this.Name = "FirmarFrm";
		this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
		this.Text = "Firmar";
		this.Closing += new
System.ComponentModel.CancelEventHandler(this.FirmarFrm_Closing);
		((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).EndInit();
		((System.ComponentModel.ISupportInitialize)(this.pictureBox2)).EndInit();
		this.groupBox1.ResumeLayout(false);
		((System.ComponentModel.ISupportInitialize)(this.pictureBox5)).EndInit();
		((System.ComponentModel.ISupportInitialize)(this.pictureBox4)).EndInit();
		((System.ComponentModel.ISupportInitialize)(this.pictureBox3)).EndInit();
		this.groupBox2.ResumeLayout(false);
		((System.ComponentModel.ISupportInitialize)(this.pictureBox7)).EndInit();
		((System.ComponentModel.ISupportInitialize)(this.pictureBox6)).EndInit();
		this.ResumeLayout(false);
			}
			#endregion
			#endregion
			private void FirmarFrm_Closing(object sender,
System.ComponentModel.CancelEventArgs e)
			{
				formularioAnterior.Enabled = true;
			}
			private void abrirLlbl_Click(object sender, System.EventArgs e)
			{
			  importarImagenDesdeArchivo();
			}
			private void limpiarLlbl_Click(object sender, System.EventArgs e)
			{
				signer.Limpiar();
			}
			private void guardarLlbl_Click(object sender, System.EventArgs e)
			{	salvarSfd.Filter = "Archivo de mapa de bits (*.bmp)|*.bmp";
				if( salvarSfd.ShowDialog() == DialogResult.OK )
				{	bool salvada = signer.SalvarFirma(salvarSfd.FileName);
					if(!salvada)
					{ MessageBox.Show(this, "La imágen no se ha podido
guardar", "Error", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
					}
				}
```

```
		}

		// Carga una firma buscando a la persona en la base de datos
		private void buscarLbl_LinkClicked(object sender,
System.Windows.Forms.LinkLabelLinkClickedEventArgs e)
		{	Constantes.formularioCarga = true;
			BuscarFrm f = new BuscarFrm(this.formularioPrincipal, 2);
			f.Show();
		}
		private void aceptarBtn_Click(object sender, System.EventArgs e)
		{
			consumidorFirma.setFirma(signer.GetFirma());
			if(Constantes.imagenATratar == 1)
				Constantes.caracteristicas1 = null;
			else if(Constantes.imagenATratar == 2)
				Constantes.caracteristicas2 = null;
	CambioImagen = true;
			this.Close();
		}
		private void cancelarBtn_Click(object sender, System.EventArgs e)
		{
			this.Close();
	CambioImagen = false;
		}
		private void importarImagenDesdeArchivo()
		{
			abrirOfd.Filter = "Tipo de archivo (*.bmp; *.gif; *.jpg;
*.png)|*.bmp;*.gif;*.jpg;*.png|" + "All files (*.*)|*.*";
			if( abrirOfd.ShowDialog()==DialogResult.OK )
			{	try
				{	Image imagen = Image.FromFile(abrirOfd.FileName);
if(imagen.Width > signer.getAnchoResolucion() || imagen.Height > signer.getAltoResolucion())
					{
MessageBox.Show(this, "Imágen no válida: Resolución máxima 300 x 300", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
					}
					else
					{
						signer.cargarImagen(imagen);
					}
				}
				catch (Exception)
				{
MessageBox.Show(this, "La imágen no se pudo cargar", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);
				}
			}
		}
		private void linkLabel1_LinkClicked(object sender,
System.Windows.Forms.LinkLabelLinkClickedEventArgs e)
		{
			signer.cargarUltimoTrazo();
		}
	}
}
```

FORMULARIO COMPARA FIRMAS

```
using System;
using System.Drawing;
using System.Drawing.Imaging;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Threading;
using System.IO;
using RecFirma.Clases;

namespace RecFirma.Formularios
{
	public class ComparaFirmasFrm : System.Windows.Forms.Form,
RecFirma.Clases.Firmable
	{	private System.ComponentModel.IContainer components = null;
		private System.Windows.Forms.PictureBox firma1Ptb;
		private System.Windows.Forms.PictureBox firma2Ptb;
		private System.Windows.Forms.OpenFileDialog abrirImagenOfd;
		private Joaqs.UI.XpButton abrirFirma1Btn;
		private Joaqs.UI.XpButton abrirFirma2Btn;
		private PrincipalFrm formularioPrincipal;
		private Joaqs.UI.XpButton compararBtn;
		private System.Windows.Forms.CheckBox euclideaChbx;
		private System.Windows.Forms.CheckBox estadisticaChbx;
		private System.Windows.Forms.CheckBox mayorPesoChbx;
		private System.Windows.Forms.GroupBox groupBox1;
		private System.Windows.Forms.ProgressBar progressBar1;
		private System.Windows.Forms.StatusBar statusBar1;
		private System.Windows.Forms.Label label1;
		private System.Windows.Forms.Label label2;
		private System.Windows.Forms.PictureBox pictureBox1;
		private System.Windows.Forms.PictureBox pictureBox2;
		private System.Windows.Forms.GroupBox groupBox2;
		private System.Windows.Forms.Label label3;
		private System.Windows.Forms.Label label4;
		private System.Windows.Forms.TextBox umbralEstadisticaTbx;
		private System.Windows.Forms.TextBox umbralEuclideoTbx;
		private Joaqs.UI.XpButton guardarImagen1Btn;
		private Joaqs.UI.XpButton guardarImagen2Btn;
		private System.Windows.Forms.SaveFileDialog guardarImagenSfd;
		private delegate void MetodoConcurrente (bool valor);
		private PictureBox ptbACargar;
		private bool actualizadaImagen1, actualizadaImagen2;
		private System.Windows.Forms.PictureBox okEuclideaPtb;
		private System.Windows.Forms.PictureBox falseEuclideaPtb;
		private System.Windows.Forms.PictureBox okEstadisticaPtb;
		private System.Windows.Forms.PictureBox okMayorPesoPtb;
		private System.Windows.Forms.PictureBox falseMayorPesoPtb;
		private System.Windows.Forms.PictureBox falseEstadisticaPtb;
		private System.Windows.Forms.GroupBox groupBox3;
		private System.Windows.Forms.PictureBox ocultoEuclideaPtb;
		private System.Windows.Forms.PictureBox ocultoMayorPesoPtb;
		private System.Windows.Forms.Label resultadoEuclidea1Lbl;
		private System.Windows.Forms.Label resultadoEuclidea2Lbl;
		private System.Windows.Forms.Label resultadoEuclidea3Lbl;
		private System.Windows.Forms.Label resultadoMayorPeso1Lbl;
		private System.Windows.Forms.Label resultadoMayorPeso2Lbl;
```

```
		private System.Windows.Forms.Label resultadoEstadistica1Lbl;
		private System.Windows.Forms.Label resultadoEstadistica2Lbl;
		private System.Windows.Forms.Label resultadoEstadistica3Lbl;
		private System.Windows.Forms.PictureBox informacionPtb;
		private System.Windows.Forms.Label informacionLbl;
		private System.Windows.Forms.Label porcentajeLbl;
		private System.Windows.Forms.PictureBox ocultoEstadisticaPtb;


		public ComparaFirmasFrm(PrincipalFrm f)
		{
			InitializeComponent();
			// Primero asignamos al formulario actual el formulario donde se abre.
			this.formularioPrincipal = f;
			this.MdiParent = this.formularioPrincipal;
			ptbACargar = null;
			// Se ocultan los elementos que no se desean mostrar
			this.progressBar1.Visible = false;
			ocultaResultados(); // Borramos los resultados de la pantalla
		}

		#region Codigo generado por .NET
		/// <summary>
		/// Limpiar los recursos que se estén utilizando.
		/// </summary>
		protected override void Dispose( bool disposing )
		{	if( disposing )
			{	if(components != null)
				{
					components.Dispose();
				}
			}
			base.Dispose( disposing );
		}
		#region Código generado por el Diseñador de Windows Forms
		/// <summary>
		/// Método necesario para admitir el Diseñador. No se puede modificar
		/// el contenido del método con el editor de código.
		/// </summary>
		private void InitializeComponent()
		{	System.Resources.ResourceManager resources = new
System.Resources.ResourceManager(typeof(ComparaFirmasFrm));
			this.firma1Ptb = new System.Windows.Forms.PictureBox();
			this.firma2Ptb = new System.Windows.Forms.PictureBox();
			this.abrirImagenOfd = new System.Windows.Forms.OpenFileDialog();
			this.abrirFirma1Btn = new Joaqs.UI.XpButton();
			this.abrirFirma2Btn = new Joaqs.UI.XpButton();
			this.compararBtn = new Joaqs.UI.XpButton();
			this.euclideaChbx = new System.Windows.Forms.CheckBox();
			this.estadisticaChbx = new System.Windows.Forms.CheckBox();
			this.mayorPesoChbx = new System.Windows.Forms.CheckBox();
			this.groupBox1 = new System.Windows.Forms.GroupBox();
			this.progressBar1 = new System.Windows.Forms.ProgressBar();
			this.statusBar1 = new System.Windows.Forms.StatusBar();
			this.label1 = new System.Windows.Forms.Label();
			this.label2 = new System.Windows.Forms.Label();
			this.pictureBox1 = new System.Windows.Forms.PictureBox();
			this.pictureBox2 = new System.Windows.Forms.PictureBox();
```

```
            this.groupBox2 = new System.Windows.Forms.GroupBox();
            this.label4 = new System.Windows.Forms.Label();
            this.label3 = new System.Windows.Forms.Label();
            this.umbralEstadisticaTbx = new System.Windows.Forms.TextBox();
            this.umbralEuclideoTbx = new System.Windows.Forms.TextBox();
            this.guardarImagen1Btn = new Joaqs.UI.XpButton();
            this.guardarImagen2Btn = new Joaqs.UI.XpButton();
            this.guardarImagenSfd = new
System.Windows.Forms.SaveFileDialog();
            this.groupBox3 = new System.Windows.Forms.GroupBox();
            this.informacionLbl = new System.Windows.Forms.Label();
            this.informacionPtb = new System.Windows.Forms.PictureBox();
            this.resultadoEstadistica3Lbl = new System.Windows.Forms.Label();
            this.resultadoMayorPeso2Lbl = new System.Windows.Forms.Label();
            this.resultadoEuclidea3Lbl = new System.Windows.Forms.Label();
            this.ocultoEstadisticaPtb = new System.Windows.Forms.PictureBox();
            this.ocultoMayorPesoPtb = new System.Windows.Forms.PictureBox();
            this.ocultoEuclideaPtb = new System.Windows.Forms.PictureBox();
            this.resultadoEstadistica1Lbl = new System.Windows.Forms.Label();
            this.resultadoMayorPeso1Lbl = new System.Windows.Forms.Label();
            this.resultadoEuclidea1Lbl = new System.Windows.Forms.Label();
            this.falseEstadisticaPtb = new System.Windows.Forms.PictureBox();
            this.falseMayorPesoPtb = new System.Windows.Forms.PictureBox();
            this.falseEuclideaPtb = new System.Windows.Forms.PictureBox();
            this.okEstadisticaPtb = new System.Windows.Forms.PictureBox();
            this.okMayorPesoPtb = new System.Windows.Forms.PictureBox();
            this.okEuclideaPtb = new System.Windows.Forms.PictureBox();
            this.resultadoEuclidea2Lbl = new System.Windows.Forms.Label();
            this.resultadoEstadistica2Lbl = new System.Windows.Forms.Label();
            this.porcentajeLbl = new System.Windows.Forms.Label();
            this.groupBox1.SuspendLayout();
            this.groupBox2.SuspendLayout();
            this.groupBox3.SuspendLayout();
            this.SuspendLayout();
            //
            // firma1Ptb
            //
            this.firma1Ptb.BackColor = System.Drawing.Color.White;
            this.firma1Ptb.BorderStyle =
System.Windows.Forms.BorderStyle.FixedSingle;
            this.firma1Ptb.Location = new System.Drawing.Point(24, 32);
            this.firma1Ptb.Name = "firma1Ptb";
            this.firma1Ptb.Size = new System.Drawing.Size(300, 300);
            this.firma1Ptb.TabIndex = 0;
            this.firma1Ptb.TabStop = false;
            //
            // firma2Ptb
            //
            this.firma2Ptb.BackColor = System.Drawing.Color.White;
            this.firma2Ptb.BorderStyle =
System.Windows.Forms.BorderStyle.FixedSingle;
            this.firma2Ptb.Location = new System.Drawing.Point(448, 32);
            this.firma2Ptb.Name = "firma2Ptb";
            this.firma2Ptb.Size = new System.Drawing.Size(300, 300);
            this.firma2Ptb.TabIndex = 1;
            this.firma2Ptb.TabStop = false;
            //
            // abrirFirma1Btn
```

```
                        //
                        this.abrirFirma1Btn.Location = new System.Drawing.Point(336, 96);
                        this.abrirFirma1Btn.Name = "abrirFirma1Btn";
                        this.abrirFirma1Btn.Size = new System.Drawing.Size(104, 32);
                        this.abrirFirma1Btn.TabIndex = 3;
                        this.abrirFirma1Btn.Text = "<< Obtener firma 1";
                        this.abrirFirma1Btn.Click += new
System.EventHandler(this.abrirFirma1Btn_Click);
                        //
                        // abrirFirma2Btn
                        //
                        this.abrirFirma2Btn.Location = new System.Drawing.Point(336, 152);
                        this.abrirFirma2Btn.Name = "abrirFirma2Btn";
                        this.abrirFirma2Btn.Size = new System.Drawing.Size(104, 32);
                        this.abrirFirma2Btn.TabIndex = 4;
                        this.abrirFirma2Btn.Text = "Obtener firma 2 >>";
                        this.abrirFirma2Btn.Click += new
System.EventHandler(this.abrirFirma2Btn_Click);
                        //
                        // compararBtn
                        //
                        this.compararBtn.Location = new System.Drawing.Point(336, 208);
                        this.compararBtn.Name = "compararBtn";
                        this.compararBtn.Size = new System.Drawing.Size(104, 32);
                        this.compararBtn.TabIndex = 5;
                        this.compararBtn.Text = "Comparar Firmas";
                        this.compararBtn.Click += new
System.EventHandler(this.compararBtn_Click);
                        //
                        // euclideaChbx
                        //
                        this.euclideaChbx.Location = new System.Drawing.Point(24, 16);
                        this.euclideaChbx.Name = "euclideaChbx";
                        this.euclideaChbx.Size = new System.Drawing.Size(120, 24);
                        this.euclideaChbx.TabIndex = 7;
                        this.euclideaChbx.Text = "Distancia Euclídea";
                        //
                        // estadisticaChbx
                        //
                        this.estadisticaChbx.Location = new System.Drawing.Point(24, 64);
                        this.estadisticaChbx.Name = "estadisticaChbx";
                        this.estadisticaChbx.Size = new System.Drawing.Size(136, 24);
                        this.estadisticaChbx.TabIndex = 8;
                        this.estadisticaChbx.Text = "Distancia Estadística";
                        //
                        // mayorPesoChbx
                        //
                        this.mayorPesoChbx.Location = new System.Drawing.Point(24, 40);
                        this.mayorPesoChbx.Name = "mayorPesoChbx";
                        this.mayorPesoChbx.Size = new System.Drawing.Size(160, 24);
                        this.mayorPesoChbx.TabIndex = 9;
                        this.mayorPesoChbx.Text = "Cualidad de Mayor Peso";
                        //
                        // groupBox1
                        //
                        this.groupBox1.Controls.Add(this.euclideaChbx);
                        this.groupBox1.Controls.Add(this.mayorPesoChbx);
                        this.groupBox1.Controls.Add(this.estadisticaChbx);
```

```
            this.groupBox1.Location = new System.Drawing.Point(24, 368);
            this.groupBox1.Name = "groupBox1";
            this.groupBox1.Size = new System.Drawing.Size(200, 96);
            this.groupBox1.TabIndex = 10;
            this.groupBox1.TabStop = false;
            this.groupBox1.Text = "Método de comparación";
            //
            // progressBar1
            //
            this.progressBar1.Location = new System.Drawing.Point(472, 472);
            this.progressBar1.Maximum = 22;
            this.progressBar1.Name = "progressBar1";
            this.progressBar1.Size = new System.Drawing.Size(240, 23);
            this.progressBar1.TabIndex = 11;
            //
            // statusBar1
            //
            this.statusBar1.Location = new System.Drawing.Point(0, 498);
            this.statusBar1.Name = "statusBar1";
            this.statusBar1.Size = new System.Drawing.Size(770, 22);
            this.statusBar1.TabIndex = 12;
            this.statusBar1.Text = "Esperando Firmas";
            //
            // label1
            //
            this.label1.Font = new System.Drawing.Font("Microsoft Sans Serif",
9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((System.Byte)(0)));
            this.label1.Location = new System.Drawing.Point(48, 8);
            this.label1.Name = "label1";
            this.label1.TabIndex = 15;
            this.label1.Text = "Firma número 1";
            //
            // label2
            //
            this.label2.Font = new System.Drawing.Font("Microsoft Sans Serif",
9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((System.Byte)(0)));
            this.label2.Location = new System.Drawing.Point(472, 8);
            this.label2.Name = "label2";
            this.label2.TabIndex = 16;
            this.label2.Text = "Firma número 2";
            //
            // pictureBox1
            //
            this.pictureBox1.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox1.Image")));
            this.pictureBox1.Location = new System.Drawing.Point(24, 8);
            this.pictureBox1.Name = "pictureBox1";
            this.pictureBox1.Size = new System.Drawing.Size(16, 16);
            this.pictureBox1.TabIndex = 17;
            this.pictureBox1.TabStop = false;
            //
            // pictureBox2
            //
            this.pictureBox2.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox2.Image")));
            this.pictureBox2.Location = new System.Drawing.Point(448, 8);
```

```
                        this.pictureBox2.Name = "pictureBox2";
                        this.pictureBox2.Size = new System.Drawing.Size(16, 16);
                        this.pictureBox2.TabIndex = 18;
                        this.pictureBox2.TabStop = false;
                        //
                        // groupBox2
                        //
                        this.groupBox2.Controls.Add(this.label4);
                        this.groupBox2.Controls.Add(this.label3);
                        this.groupBox2.Controls.Add(this.umbralEstadisticaTbx);
                        this.groupBox2.Controls.Add(this.umbralEuclideoTbx);
                        this.groupBox2.Location = new System.Drawing.Point(244, 368);
                        this.groupBox2.Name = "groupBox2";
                        this.groupBox2.Size = new System.Drawing.Size(184, 96);
                        this.groupBox2.TabIndex = 19;
                        this.groupBox2.TabStop = false;
                        this.groupBox2.Text = "Umbral";
                        //
                        // label4
                        //
                        this.label4.Font = new System.Drawing.Font("Microsoft Sans Serif",
8.25F, System.Drawing.FontStyle.Italic, System.Drawing.GraphicsUnit.Point,
((System.Byte)(0)));
                        this.label4.ForeColor = System.Drawing.SystemColors.ControlDark;
                        this.label4.Location = new System.Drawing.Point(80, 56);
                        this.label4.Name = "label4";
                        this.label4.Size = new System.Drawing.Size(96, 32);
                        this.label4.TabIndex = 17;
                        this.label4.Text = "empírico 0,82, Rango[-1, 1]";
                        //
                        // label3
                        //
                        this.label3.Font = new System.Drawing.Font("Microsoft Sans Serif",
8.25F, System.Drawing.FontStyle.Italic, System.Drawing.GraphicsUnit.Point,
((System.Byte)(0)));
                        this.label3.ForeColor = System.Drawing.SystemColors.ControlDark;
                        this.label3.Location = new System.Drawing.Point(80, 16);
                        this.label3.Name = "label3";
                        this.label3.Size = new System.Drawing.Size(96, 32);
                        this.label3.TabIndex = 16;
                        this.label3.Text = "empírico 1300, Rango [0, infinito)";
                        //
                        // umbralEstadisticaTbx
                        //
                        this.umbralEstadisticaTbx.Location = new System.Drawing.Point(8,
56);
                        this.umbralEstadisticaTbx.Name = "umbralEstadisticaTbx";
                        this.umbralEstadisticaTbx.Size = new System.Drawing.Size(64, 20);
                        this.umbralEstadisticaTbx.TabIndex = 15;
                        this.umbralEstadisticaTbx.Text = "0,82";
                        //
                        // umbralEuclideoTbx
                        //
                        this.umbralEuclideoTbx.BackColor =
System.Drawing.SystemColors.Window;
                        this.umbralEuclideoTbx.Location = new System.Drawing.Point(8, 24);
                        this.umbralEuclideoTbx.Name = "umbralEuclideoTbx";
                        this.umbralEuclideoTbx.Size = new System.Drawing.Size(64, 20);
```

```
                        this.umbralEuclideoTbx.TabIndex = 13;
                        this.umbralEuclideoTbx.Text = "1300";
                        //
                        // guardarImagen1Btn
                        //
                        this.guardarImagen1Btn.Location = new System.Drawing.Point(24,
336);
                        this.guardarImagen1Btn.Name = "guardarImagen1Btn";
                        this.guardarImagen1Btn.Size = new System.Drawing.Size(96, 24);
                        this.guardarImagen1Btn.TabIndex = 20;
                        this.guardarImagen1Btn.Text = "Guardar imagen";
                        this.guardarImagen1Btn.Visible = false;
                        this.guardarImagen1Btn.Click += new
System.EventHandler(this.guardarImagen1Btn_Click);
                        //
                        // guardarImagen2Btn
                        //
                        this.guardarImagen2Btn.Location = new System.Drawing.Point(448,
336);
                        this.guardarImagen2Btn.Name = "guardarImagen2Btn";
                        this.guardarImagen2Btn.Size = new System.Drawing.Size(96, 24);
                        this.guardarImagen2Btn.TabIndex = 21;
                        this.guardarImagen2Btn.Text = "Guardar imagen";
                        this.guardarImagen2Btn.Visible = false;
                        this.guardarImagen2Btn.Click += new
System.EventHandler(this.guardarImagen2Btn_Click);
                        //
                        // groupBox3
                        //
                        this.groupBox3.Controls.Add(this.informacionLbl);
                        this.groupBox3.Controls.Add(this.informacionPtb);
                        this.groupBox3.Controls.Add(this.resultadoEstadistica3Lbl);
                        this.groupBox3.Controls.Add(this.resultadoMayorPeso2Lbl);
                        this.groupBox3.Controls.Add(this.resultadoEuclidea3Lbl);
                        this.groupBox3.Controls.Add(this.ocultoEstadisticaPtb);
                        this.groupBox3.Controls.Add(this.ocultoMayorPesoPtb);
                        this.groupBox3.Controls.Add(this.ocultoEuclideaPtb);
                        this.groupBox3.Controls.Add(this.resultadoEstadistica1Lbl);
                        this.groupBox3.Controls.Add(this.resultadoMayorPeso1Lbl);
                        this.groupBox3.Controls.Add(this.resultadoEuclidea1Lbl);
                        this.groupBox3.Controls.Add(this.falseEstadisticaPtb);
                        this.groupBox3.Controls.Add(this.falseMayorPesoPtb);
                        this.groupBox3.Controls.Add(this.falseEuclideaPtb);
                        this.groupBox3.Controls.Add(this.okEstadisticaPtb);
                        this.groupBox3.Controls.Add(this.okMayorPesoPtb);
                        this.groupBox3.Controls.Add(this.okEuclideaPtb);
                        this.groupBox3.Controls.Add(this.resultadoEuclidea2Lbl);
                        this.groupBox3.Controls.Add(this.resultadoEstadistica2Lbl);
                        this.groupBox3.Location = new System.Drawing.Point(448, 368);
                        this.groupBox3.Name = "groupBox3";
                        this.groupBox3.Size = new System.Drawing.Size(301, 96);
                        this.groupBox3.TabIndex = 22;
                        this.groupBox3.TabStop = false;
                        this.groupBox3.Text = "Resultados";
                        //
                        // informacionLbl
                        //
                        this.informacionLbl.Location = new System.Drawing.Point(40, 24);
```

```
                this.informacionLbl.Name = "informacionLbl";
                this.informacionLbl.Size = new System.Drawing.Size(248, 56);
                this.informacionLbl.TabIndex = 18;
this.informacionLbl.Text = "Este recuadro contendrá los resultados obtenidos de la
comparación de las firmas " + "utilizando para ello los diferentes métodos de comparación
seleccionados";
                //
                // informacionPtb
                //
                this.informacionPtb.Image =
((System.Drawing.Image)(resources.GetObject("informacionPtb.Image")));
                this.informacionPtb.Location = new System.Drawing.Point(16, 24);
                this.informacionPtb.Name = "informacionPtb";
                this.informacionPtb.Size = new System.Drawing.Size(16, 16);
                this.informacionPtb.TabIndex = 17;
                this.informacionPtb.TabStop = false;
                //
                // resultadoEstadistica3Lbl
                //
            this.resultadoEstadistica3Lbl.Location = new System.Drawing.Point(224, 72);
                this.resultadoEstadistica3Lbl.Name = "resultadoEstadistica3Lbl";
                this.resultadoEstadistica3Lbl.Size = new System.Drawing.Size(48,
15);
                this.resultadoEstadistica3Lbl.TabIndex = 16;
                //
                // resultadoMayorPeso2Lbl
                //
            this.resultadoMayorPeso2Lbl.Location = new System.Drawing.Point(168, 48);
                this.resultadoMayorPeso2Lbl.Name = "resultadoMayorPeso2Lbl";
                this.resultadoMayorPeso2Lbl.Size = new System.Drawing.Size(88,
16);
                this.resultadoMayorPeso2Lbl.TabIndex = 14;
                //
                // resultadoEuclidea3Lbl
                //
                this.resultadoEuclidea3Lbl.Location = new System.Drawing.Point(224,
24);
                this.resultadoEuclidea3Lbl.Name = "resultadoEuclidea3Lbl";
                this.resultadoEuclidea3Lbl.Size = new System.Drawing.Size(48, 15);
                this.resultadoEuclidea3Lbl.TabIndex = 13;
                //
                // ocultoEstadisticaPtb
                //
                this.ocultoEstadisticaPtb.Image =
((System.Drawing.Image)(resources.GetObject("ocultoEstadisticaPtb.Image")));
                this.ocultoEstadisticaPtb.Location = new System.Drawing.Point(16,
72);
                this.ocultoEstadisticaPtb.Name = "ocultoEstadisticaPtb";
                this.ocultoEstadisticaPtb.Size = new System.Drawing.Size(16, 16);
                this.ocultoEstadisticaPtb.TabIndex = 11;
                this.ocultoEstadisticaPtb.TabStop = false;
                //
                // ocultoMayorPesoPtb
                //
                this.ocultoMayorPesoPtb.Image =
((System.Drawing.Image)(resources.GetObject("ocultoMayorPesoPtb.Image")));
                this.ocultoMayorPesoPtb.Location = new System.Drawing.Point(16,
48);
```

```
                this.ocultoMayorPesoPtb.Name = "ocultoMayorPesoPtb";
                this.ocultoMayorPesoPtb.Size = new System.Drawing.Size(16, 16);
                this.ocultoMayorPesoPtb.TabIndex = 10;
                this.ocultoMayorPesoPtb.TabStop = false;
                //
                // ocultoEuclideaPtb
                //
                this.ocultoEuclideaPtb.Image =
((System.Drawing.Image)(resources.GetObject("ocultoEuclideaPtb.Image")));
                this.ocultoEuclideaPtb.Location = new System.Drawing.Point(16, 24);
                this.ocultoEuclideaPtb.Name = "ocultoEuclideaPtb";
                this.ocultoEuclideaPtb.Size = new System.Drawing.Size(16, 16);
                this.ocultoEuclideaPtb.TabIndex = 9;
                this.ocultoEuclideaPtb.TabStop = false;
                //
                // resultadoEstadistica1Lbl
                //
                this.resultadoEstadistica1Lbl.Location = new
System.Drawing.Point(40, 72);
                this.resultadoEstadistica1Lbl.Name = "resultadoEstadistica1Lbl";
                this.resultadoEstadistica1Lbl.Size = new System.Drawing.Size(112,
16);
                this.resultadoEstadistica1Lbl.TabIndex = 8;
                //
                // resultadoMayorPeso1Lbl
                //
            this.resultadoMayorPeso1Lbl.Location = new System.Drawing.Point(40, 48);
                this.resultadoMayorPeso1Lbl.Name = "resultadoMayorPeso1Lbl";
                this.resultadoMayorPeso1Lbl.Size = new System.Drawing.Size(136,
16);
                this.resultadoMayorPeso1Lbl.TabIndex = 7;
                //
                // resultadoEuclidea1Lbl
                //
                this.resultadoEuclidea1Lbl.Location = new System.Drawing.Point(40,
24);
                this.resultadoEuclidea1Lbl.Name = "resultadoEuclidea1Lbl";
                this.resultadoEuclidea1Lbl.Size = new System.Drawing.Size(104, 16);
                this.resultadoEuclidea1Lbl.TabIndex = 6;
                //
                // falseEstadisticaPtb
                //
                this.falseEstadisticaPtb.Image =
((System.Drawing.Image)(resources.GetObject("falseEstadisticaPtb.Image")));
                this.falseEstadisticaPtb.Location = new System.Drawing.Point(16, 72);
                this.falseEstadisticaPtb.Name = "falseEstadisticaPtb";
                this.falseEstadisticaPtb.Size = new System.Drawing.Size(16, 16);
                this.falseEstadisticaPtb.TabIndex = 5;
                this.falseEstadisticaPtb.TabStop = false;
                //
                // falseMayorPesoPtb
                //
                this.falseMayorPesoPtb.Image =
((System.Drawing.Image)(resources.GetObject("falseMayorPesoPtb.Image")));
                this.falseMayorPesoPtb.Location = new System.Drawing.Point(16,
48);
                this.falseMayorPesoPtb.Name = "falseMayorPesoPtb";
                this.falseMayorPesoPtb.Size = new System.Drawing.Size(16, 16);
```

```
                        this.falseMayorPesoPtb.TabIndex = 4;
                        this.falseMayorPesoPtb.TabStop = false;
                        //
                        // falseEuclideaPtb
                        //
                        this.falseEuclideaPtb.Image =
((System.Drawing.Image)(resources.GetObject("falseEuclideaPtb.Image")));
                        this.falseEuclideaPtb.Location = new System.Drawing.Point(16, 24);
                        this.falseEuclideaPtb.Name = "falseEuclideaPtb";
                        this.falseEuclideaPtb.Size = new System.Drawing.Size(16, 16);
                        this.falseEuclideaPtb.TabIndex = 3;
                        this.falseEuclideaPtb.TabStop = false;
                        //
                        // okEstadisticaPtb
                        //
                        this.okEstadisticaPtb.Image =
((System.Drawing.Image)(resources.GetObject("okEstadisticaPtb.Image")));
                        this.okEstadisticaPtb.Location = new System.Drawing.Point(16, 72);
                        this.okEstadisticaPtb.Name = "okEstadisticaPtb";
                        this.okEstadisticaPtb.Size = new System.Drawing.Size(16, 16);
                        this.okEstadisticaPtb.TabIndex = 2;
                        this.okEstadisticaPtb.TabStop = false;
                        //
                        // okMayorPesoPtb
                        //
                        this.okMayorPesoPtb.Image =
((System.Drawing.Image)(resources.GetObject("okMayorPesoPtb.Image")));
                        this.okMayorPesoPtb.Location = new System.Drawing.Point(16, 48);
                        this.okMayorPesoPtb.Name = "okMayorPesoPtb";
                        this.okMayorPesoPtb.Size = new System.Drawing.Size(16, 16);
                        this.okMayorPesoPtb.TabIndex = 1;
                        this.okMayorPesoPtb.TabStop = false;
                        //
                        // okEuclideaPtb
                        //
                        this.okEuclideaPtb.Image =
((System.Drawing.Image)(resources.GetObject("okEuclideaPtb.Image")));
                        this.okEuclideaPtb.Location = new System.Drawing.Point(16, 24);
                        this.okEuclideaPtb.Name = "okEuclideaPtb";
                        this.okEuclideaPtb.Size = new System.Drawing.Size(16, 16);
                        this.okEuclideaPtb.TabIndex = 0;
                        this.okEuclideaPtb.TabStop = false;
                        //
                        // resultadoEuclidea2Lbl
                        //
                        this.resultadoEuclidea2Lbl.Location = new System.Drawing.Point(168,
24);
                        this.resultadoEuclidea2Lbl.Name = "resultadoEuclidea2Lbl";
                        this.resultadoEuclidea2Lbl.Size = new System.Drawing.Size(88, 16);
                        this.resultadoEuclidea2Lbl.TabIndex = 12;
                        //
                        // resultadoEstadistica2Lbl
                        //
                this.resultadoEstadistica2Lbl.Location = new System.Drawing.Point(168, 72);
                        this.resultadoEstadistica2Lbl.Name = "resultadoEstadistica2Lbl";
                        this.resultadoEstadistica2Lbl.Size = new System.Drawing.Size(88,
16);
                        this.resultadoEstadistica2Lbl.TabIndex = 15;
```

```
                        //
                        // porcentajeLbl
                        //
                        this.porcentajeLbl.Location = new System.Drawing.Point(720, 472);
                        this.porcentajeLbl.Name = "porcentajeLbl";
                        this.porcentajeLbl.Size = new System.Drawing.Size(40, 24);
                        this.porcentajeLbl.TabIndex = 23;
                        this.porcentajeLbl.TextAlign =
System.Drawing.ContentAlignment.MiddleCenter;
                        //
                        // ComparaFirmasFrm
                        //
                        this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);
                        this.BackColor = System.Drawing.SystemColors.ControlLight;
                        this.ClientSize = new System.Drawing.Size(770, 520);
                        this.Controls.Add(this.porcentajeLbl);
                        this.Controls.Add(this.groupBox3);
                        this.Controls.Add(this.guardarImagen2Btn);
                        this.Controls.Add(this.guardarImagen1Btn);
                        this.Controls.Add(this.groupBox2);
                        this.Controls.Add(this.pictureBox2);
                        this.Controls.Add(this.pictureBox1);
                        this.Controls.Add(this.label2);
                        this.Controls.Add(this.label1);
                        this.Controls.Add(this.statusBar1);
                        this.Controls.Add(this.progressBar1);
                        this.Controls.Add(this.groupBox1);
                        this.Controls.Add(this.compararBtn);
                        this.Controls.Add(this.abrirFirma2Btn);
                        this.Controls.Add(this.abrirFirma1Btn);
                        this.Controls.Add(this.firma2Ptb);
                        this.Controls.Add(this.firma1Ptb);
                        this.FormBorderStyle =
System.Windows.Forms.FormBorderStyle.FixedToolWindow;
                        this.MaximizeBox = false;
                        this.Name = "ComparaFirmasFrm";
                        this.StartPosition =
System.Windows.Forms.FormStartPosition.CenterScreen;
                        this.Text = "Comparar firmas";
                        this.groupBox1.ResumeLayout(false);
                        this.groupBox2.ResumeLayout(false);
                        this.groupBox3.ResumeLayout(false);
                        this.ResumeLayout(false);
                }
                #endregion
                #endregion
                public PictureBox getFirma1()
                {    return firma1Ptb;
                }
                public PictureBox getFirma2()
                {
                        return firma2Ptb;
                }
                #region Miembros de Firmable
                public void setFirma(System.Drawing.Bitmap firma)
                {
                        ptbACargar.Image = firma;
                }
```

```
		#endregion
		private void abrirFormularioFirma()
		{
			FirmarFrm f = new FirmarFrm(formularioPrincipal, this, this);
			f.Show();
			this.Enabled = false;
		}
		/* Esta función abre la ventana que permite al usuario cargar una firma
		 * existente o firmar sobre el cuadro de imagen original del formulario.
		 */
		private void abrirFirma1Btn_Click(object sender, System.EventArgs e)
		{
	Constantes.imagenATratar = 1; // Para que limpie las características de la primera
imagen
			abrirFormularioFirma();
			ptbACargar = firma1Ptb;
			guardarImagen1Btn.Visible = true;
			ocultaResultados();
		}
	/* Esta función abre la ventana que permite al usuario cargar una firma
		 * existente o firmar sobre el cuadro de imagen original del formulario.
		 */
		private void abrirFirma2Btn_Click(object sender, System.EventArgs e)
		{
	Constantes.imagenATratar = 2; // Para que limpie las características de la segunda
imagen
			abrirFormularioFirma();
			ptbACargar = firma2Ptb;
			guardarImagen2Btn.Visible = true;
			ocultaResultados();
		}
		private void guardarImagen1Btn_Click(object sender, System.EventArgs e)
		{
			guardarImagenSfd.Filter = "Archivo de mapa de bits (*.bmp)|*.bmp";
			if( guardarImagenSfd.ShowDialog() == DialogResult.OK )
			{	try
				{
			firma1Ptb.Image.Save(guardarImagenSfd.FileName,
ImageFormat.Bmp);
				}
				catch(Exception)
				{
MessageBox.Show(this, "La imágen no se ha podido guardar", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
				}
			}
		}
		private void guardarImagen2Btn_Click(object sender, System.EventArgs e)
		{	guardarImagenSfd.Filter = "Archivo de mapa de bits (*.bmp)|*.bmp";
			if( guardarImagenSfd.ShowDialog() == DialogResult.OK )
			{	try
				{
			firma2Ptb.Image.Save(guardarImagenSfd.FileName,
ImageFormat.Bmp);
				}
				catch(Exception)
				{
```

```
                        MessageBox.Show(this, "La imágen no se ha podido guardar",
"Error", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                                }
                        }
                }
                /* Esta función es la que encapsula toda la comparación que se hace
                 * entre las dos imagenes que contienen las firmas.
                 */
                private void compararBtn_Click(object sender, System.EventArgs e)
                {
                        if(firma1Ptb.Image != null && firma2Ptb.Image != null)
                        {
if(!this.euclideaChbx.Checked && !this.estadisticaChbx.Checked &&
!mayorPesoChbx.Checked)
MessageBox.Show(this, "Seleccione al menos un método de comparación", "Error en la
validacion de la firma", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        else // Si están cargadas las imágenes y seleccionado algún método de comparación
                                {
                                        //Thread t = new Thread(new ThreadStart(hilo));
                                        //t.Start();
                                        hilo();
                                        informacionLbl.Visible = false;
                                        informacionPtb.Visible = false;
                                }
                        }
                        else
MessageBox.Show(this, "ERROR: Debe abrir primero las imagenes a comparar", "Error en la
validacion de la firma", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                }
                private void hilo()
                {       this.Cursor = Cursors.WaitCursor;
                        muestraProgreso(true);
                        this.Refresh();
                        this.progressBar1.Value = 0;
                        this.actualizadaImagen1 = false;
                        this.actualizadaImagen2 = false;
                // Ahora hacemos el tratamiento de las matrices. Aplicando el operador Sobel
                        // y el gradiente a cada una de las imágenes
                        if(Constantes.caracteristicas1 == null)
                        {
                                Constantes.imagenATratar = 1;
                                extraeCaracteristicas(this.firma1Ptb.Image);
                                this.actualizadaImagen1 = true;
                        }
                        if(Constantes.caracteristicas2 == null)
                        {
                                Constantes.imagenATratar = 2;
                                extraeCaracteristicas(this.firma2Ptb.Image);
                                this.actualizadaImagen2 = true;
                        }
                        //progressBar1.Visible = false;
                        this.Cursor = Cursors.Default;
                        comparaFirmas();
                if(this.actualizadaImagen2) // Si se han actualizado las 2 firmas o solo la
segunda
                        {
                                if(this.actualizadaImagen1) // Si solo se actualiza la primera
                                        actualizaProgreso("", 100, 2);
```

```
                        else
                                actualizaProgreso("", 100, 12);
                        }
                else // Si se ha actulizado la primera firma o se ha cambiado la comparación
                        {
                                if(this.actualizadaImagen1)
                                        actualizaProgreso("", 140, 12);
                                else
                                        actualizaProgreso("", 100, 22);
                        }
                        actualizaProgreso("Listo", 0, 0);
                        this.statusBar1.Text = "Listo";
                        muestraProgreso(false);
                        //this.Refresh();
                        //this.progressBar1.Visible = false;
                        //concurrente.EndInvoke(operacion);
                }

        /* Esta funcion aplica el operador de Sobel tanto horizontalmente como verticalmente,
         * obteniendo para cada pasada una matriz en escala de grises de las imagen. Dichas
 * matrices serán Sx y Sy. A partir de dicho operador realiza la extracción de las
características
                 * de la imagen.
                 */
                private void extraeCaracteristicas(System.Drawing.Image imagen)
                {
        // Primero creo una matriz de pixeles del tamaño del lienzo donde esta la imagen
dibujada
                        bool[][] pixels = new bool[firma1Ptb.Image.Width][];
                        for(int i=0; i<firma1Ptb.Image.Width; i++)
                                pixels[i] = new bool[firma1Ptb.Image.Height];
                        // Copiamos el bitmap original
                        Bitmap copia = new Bitmap((Bitmap) imagen);
                        TransformaImagen sobel = new TransformaImagen(copia);
                        TransformaImagen gradiente = new TransformaImagen(copia);
Imagenes.inicializaMatricesImagen(); //Limpiamos las matrices de la imagen y gradiente
correspondientes
                        // Tratamiento de la imagen (cálculo de Sobel y Gradiente
                        //*****************************************************
actualizaProgreso("Cargando características imagen " + Constantes.imagenATratar.ToString()
+ " [Características de gradiente] ...", 0, 0);
                                sobel.AplicarSobel();
                                actualizaProgreso("", 45, 2);
                                // Se obtienen las características del gradiente
                                Constantes.inicializa();
                                gradiente.Gradiente();
                                actualizaProgreso("", 70, 2);
                        //*****************************************************
                        double[][]p = Imagenes.matrizGradiente1;
                        int [] grad = Constantes.Gradiente1f1;
                // Primero trato la imagen que está siendo comparada en el primer recuadro
                        if(Constantes.imagenATratar == 2)
                        {
                                Imagenes.imagenSobel1 = new Bitmap(sobel.Bitmap);
                                Imagenes.imagenGradiente1 = new
Bitmap(gradiente.Bitmap);
                                Imagenes.imagenBordesSobel1 = new Bitmap(sobel.Bitmap);
                        }
```

```
                        else
                        {
                                Imagenes.imagenSobel2 = new Bitmap(sobel.Bitmap);
                                Imagenes.imagenGradiente2 = new
Bitmap(gradiente.Bitmap);
                                Imagenes.imagenBordesSobel2 = new Bitmap(sobel.Bitmap);
                        }

                        // Se extraen las características estructurales y de concavidad
actualizaProgreso("Cargando características imagen " + Constantes.imagenATratar.ToString()
+ " [Características estructurales] ...", 0, 0);
                        gradiente.structuralFeatures();
                        actualizaProgreso("Cargando características imagen " +
Constantes.imagenATratar.ToString() + " [Características de concavidad] ...", 75, 2);
                        gradiente.concavityFeatures();
                        actualizaProgreso("", 0, 2);
                        // Por último se rellena el vector de características
                        gradiente.rellenaVectoresCaractetisticas();
                        actualizaProgreso("", 80, 2); // Se incrementa la barra de progreso
                } // Fin de la función que aplica Sobel, Gradiente y Threshold
                private void comparaFirmas()
                {       int []carac = Constantes.caracteristicas1;
                        ocultoEuclideaPtb.Visible = true;
                        ocultoMayorPesoPtb.Visible = true;
                        ocultoEstadisticaPtb.Visible = true;
                        if(this.euclideaChbx.Checked)
                        { actualizaProgreso("Comparando firmas [Distancia Euclídea] ...", 0,
0);
                                ocultoEuclideaPtb.Visible = false;
                                resultadoEuclidea3Lbl.BringToFront();
                                comparaDistanciaEuclidea();
                                //actualizaProgreso("", 0, 2);
                        }
                        else
                        {
                                resultadoEuclidea1Lbl.Text = "Distancia Euclídea:";
                        resultadoEuclidea2Lbl.ForeColor = System.Drawing.Color.DarkGray;
                                resultadoEuclidea2Lbl.Text = "No seleccionada";
                                resultadoEuclidea3Lbl.SendToBack();
                                resultadoEuclidea3Lbl.Text = "";
                                ocultoEuclideaPtb.Visible = true;
                        }
                        if(this.mayorPesoChbx.Checked)
                        {
                actualizaProgreso("Comparando firmas [Característica mayor peso] ...", 0, 0);
                                ocultoMayorPesoPtb.Visible = false;
                                comparaCaracteristicaMayorPeso();
                                //actualizaProgreso("", 0, 2);
                        }
                        else
                        {
                                resultadoMayorPeso1Lbl.Text = "Cualidad de Mayor Peso:";
                        resultadoMayorPeso2Lbl.ForeColor =
System.Drawing.Color.DarkGray;
                                resultadoMayorPeso2Lbl.Text = "No seleccionada";
                                ocultoMayorPesoPtb.Visible = true;
                        }
                        if(this.estadisticaChbx.Checked)
```

```
                        {
                        actualizaProgreso("Comparando firmas [Distancia Estadística] ...", 0,
0);
                                ocultoEstadisticaPtb.Visible = false;
                                resultadoEstadistica3Lbl.BringToFront();
                                comparaDistanciaEstadistica();
                                //actualizaProgreso("", 0, 2);
                        }
                        else
                        {
                                resultadoEstadistica1Lbl.Text = "Distancia Estadística:";
                        resultadoEstadistica2Lbl.ForeColor =
System.Drawing.Color.DarkGray;
                                resultadoEstadistica2Lbl.Text = "No seleccionada";
                                resultadoEstadistica3Lbl.SendToBack();
                                resultadoEstadistica3Lbl.Text = "";
                                ocultoEstadisticaPtb.Visible = true;
                        }
                }

                private void comparaDistanciaEuclidea()
                {
                        this.statusBar1.Text = "Comparando firmas usando distancia euclídea
...";
                int [] diferencias = new
int[(2*Constantes.K+9)*Constantes.N1*Constantes.N2];
                        double diferencia = 0, longitud = Constantes.caracteristicas1.Length;
                        double umbral = Double.Parse(this.umbralEuclideoTbx.Text);
                        double l = 0;
                        for(int i=0; i<longitud; i++)
                        {
                l = Math.Pow(Constantes.caracteristicas1[i] - Constantes.caracteristicas2[i],
2);
                                diferencia = diferencia + (double)Math.Sqrt(l);
                                diferencias[i] = (int)l;
                        }
                        // Escribimos los resultados por pantalla
                        //*********************************************
                        resultadoEuclidea1Lbl.Text ="Distancia Euclídea:";
                        resultadoEuclidea3Lbl.Text = "(" + diferencia.ToString() + ")";
                        if(diferencia <= umbral)
                        {
                                resultadoEuclidea2Lbl.ForeColor =
System.Drawing.Color.Green;
                                resultadoEuclidea2Lbl.Text = "Válido!";
                                okEuclideaPtb.Visible = true;
                                falseEuclideaPtb.Visible = false;
                        }
                        else
                        {
                                resultadoEuclidea2Lbl.ForeColor =
System.Drawing.Color.Red;
                                resultadoEuclidea2Lbl.Text = "No Válido!";
                                falseEuclideaPtb.Visible = true;
                                okEuclideaPtb.Visible = false;
                        }
                }
                private void comparaDistanciaEstadistica()
```

```
		{
		this.statusBar1.Text = "Comparando firmas usando distancia estadística ...";
			int longitud = Constantes.caracteristicas1.Length;
			int unos = 0, ceros = 0, unocero = 0, cerouno = 0;
			double diferencia = 0, umbral =
Double.Parse(this.umbralEstadisticaTbx.Text);
			for(int i=0; i<longitud; i++)
			{	if(Constantes.caracteristicas1[i] >= 50)
				{	if(Constantes.caracteristicas2[i] < 50)
					unocero++;
					else
					unos++;
				}
				else
				{	if(Constantes.caracteristicas2[i] < 50)
					ceros++;
					else
					cerouno++;
				}
			}
			double dividendo = 0, divisor = 0;
			dividendo = (unos * ceros) - (unocero * cerouno);
			divisor = (unocero + unos) * (cerouno + ceros);
			divisor = divisor * (unos + cerouno);
			divisor = divisor * (ceros + unocero);
			divisor = Math.Sqrt(divisor);
			diferencia = dividendo / divisor;
			// Escribimos los resultados por pantalla
			//*****************************************
			resultadoEstadistica1Lbl.Text ="Distancia Estadística:";
			String aux = diferencia.ToString();
			if(aux.Length > 4)
			aux = aux.Substring(0, 4);
			resultadoEstadistica3Lbl.Text = "(" + aux + ")";
			if(diferencia <= umbral)
			{
				resultadoEstadistica2Lbl.ForeColor =
System.Drawing.Color.Red;
				resultadoEstadistica2Lbl.Text = "No Válido!";
				falseEstadisticaPtb.Visible = true;
				okEstadisticaPtb.Visible = false;
			}
			else
			{
				resultadoEstadistica2Lbl.ForeColor =
System.Drawing.Color.Green;
				resultadoEstadistica2Lbl.Text = "Válido!";
				okEstadisticaPtb.Visible = true;
				falseEstadisticaPtb.Visible = false;
			}
		}
		private void comparaCaracteristicaMayorPeso()
		{
		this.statusBar1.Text = "Comparando firmas usando comparación por bloques
...";
			int i = 0, tam = Constantes.N1*Constantes.N2, k = 0;
		int diferenciaGradiente = 0, diferenciaEstructural = 0, diferenciaConcavidad =
0;
```

```
                        bool parecidos = true;
// Se recorre hasta el final o hasta que encontremos un conjunto de características muy
diferentes
                        while(i<Constantes.caracteristicas1.Length && parecidos)
                        {       if(i % Constantes.numCaracteristicasPorBloque == 0)
        {  if(diferenciaGradiente > 40 || diferenciaEstructural > 40 || diferenciaConcavidad >
40)
                                {       parecidos = false;
                                        break;
                                }
                                diferenciaGradiente = 0;
                                diferenciaEstructural = 0;
                                diferenciaConcavidad = 0;
                                k = 0;
                        }
                        if(k<Constantes.K)
diferenciaGradiente = diferenciaGradiente + Math.Abs(Constantes.caracteristicas1[i] -
Constantes.caracteristicas2[i]);
                        else if(i>=Constantes.K && i<(Constantes.K*2)
diferenciaEstructural = diferenciaEstructural + Math.Abs(Constantes.caracteristicas1[i] -
Constantes.caracteristicas2[i]);
                        else
                                diferenciaConcavidad = diferenciaConcavidad +
Math.Abs(Constantes.caracteristicas1[i] - Constantes.caracteristicas2[i]);
                        i++;
                        k++;
                }
                // Escribimos los resultados por pantalla
                //*****************************************
                resultadoMayorPeso1Lbl.Text ="Cualidad de Mayor Peso:";
                if(parecidos)
                {
                resultadoMayorPeso2Lbl.ForeColor = System.Drawing.Color.Green;
                        resultadoMayorPeso2Lbl.Text = "Válido!";
                        okMayorPesoPtb.Visible = true;
                        falseMayorPesoPtb.Visible = false;
                }
                else
                {       resultadoMayorPeso2Lbl.ForeColor =
System.Drawing.Color.Red;
                        resultadoMayorPeso2Lbl.Text = "No Válido!";
                        falseMayorPesoPtb.Visible = true;
                        okMayorPesoPtb.Visible = false;
                }
        }
        private void importarImagenDesdeArchivo(bool origen)
        { abrirImagenOfd.Filter = "Tipo de archivo (*.bmp; *.gif; *.jpg;
*.png)|*.bmp;*.gif;*.jpg;*.png|" + "All files (*.*)|*.*";
                if( abrirImagenOfd.ShowDialog()==DialogResult.OK )
                {       try
                {       Image imagen = Image.FromFile(abrirImagenOfd.FileName);
                        if(imagen.Width > 300 || imagen.Height > 300)
                                {
MessageBox.Show(this, "Imágen no válida: Resolución máxima 300 x 300", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                                }
                                else
                                {
```

```
                                        if (origen)
                                                this.firma1Ptb.Image = imagen;
                                        else
                                                this.firma2Ptb.Image = imagen;
                                }
                        }
                        catch (Exception)
                        {
MessageBox.Show(this, "La imágen no se pudo cargar", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);
                        }
                }
        }
        private void actualizaProgreso(String texto, int porcentaje, int incremento)
        {       int valor = this.progressBar1.Value;
                if(texto.CompareTo("") != 0)
                        this.statusBar1.Text = texto;
                if(porcentaje != 0)
                {       if(Constantes.imagenATratar == 1)
                        porcentaje = porcentaje - 40;
                        porcentajeLbl.Text = porcentaje.ToString() + " %";
                }
                if(incremento != 0)
                        this.progressBar1.Value = valor + incremento;
                        //this.progressBar1.Increment(incremento);
                this.Refresh();
        }
        private void muestraProgreso(bool valor)
        {
                this.progressBar1.Visible = valor;
                this.porcentajeLbl.Visible = valor;
                //this.imagenGifPtb.Visible = valor;
        }
        // Oculta todos los iconos del resultado
        private void ocultaResultados()
        {       okEuclideaPtb.Visible = false;
                falseEuclideaPtb.Visible = false;
                resultadoEuclidea1Lbl.Text = "";
                resultadoEuclidea2Lbl.Text = "";
                resultadoEuclidea3Lbl.Text = "";
                okMayorPesoPtb.Visible = false;
                falseMayorPesoPtb.Visible = false;
                resultadoMayorPeso1Lbl.Text = "";
                resultadoMayorPeso2Lbl.Text = "";
                okEstadisticaPtb.Visible = false;
                falseEstadisticaPtb.Visible = false;
                resultadoEstadistica1Lbl.Text = "";
                resultadoEstadistica2Lbl.Text = "";
                resultadoEstadistica3Lbl.Text = "";
                ocultoEuclideaPtb.Visible = false;
                ocultoMayorPesoPtb.Visible = false;
                ocultoEstadisticaPtb.Visible = false;
                // Tambien se pone a visible el cuadro de informacion
                informacionLbl.Visible = true;
                informacionPtb.Visible = true;
        }
        private void visible()
        {  //this.imagenGifPtb.Visible = true;
```

```
                }

        } // Fin de la clase
}
```

FORMULARIO BUSCAR

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Drawing;
using System.IO;
using System.Windows.Forms;
using RecFirma.Clases;
namespace RecFirma.Formularios
{
        /// <summary>
        /// Descripción breve de BuscarFrm.
        /// </summary>
        public class BuscarFrm : System.Windows.Forms.Form
        {       #region Atributos de .NET
                private System.Windows.Forms.TextBox textBox1;
                private System.Windows.Forms.Label etiquetaLbl;
                private Joaqs.UI.XpButton cancelarBtn;
                private System.ComponentModel.Container components = null;
                #endregion
                private PrincipalFrm formularioPrincipal;
                private Joaqs.UI.XpButton aceptarBtn;
                private System.Windows.Forms.PictureBox pictureBox1;
                private int tipo;
      private Firmable consumidorFirma;
                public BuscarFrm(PrincipalFrm f, int tipo)
                {       InitializeComponent();
                        this.formularioPrincipal = f;
                        this.MdiParent = this.formularioPrincipal;
                        this.tipo = tipo;
                }

                #region Codigo generado por .NET
                protected override void Dispose( bool disposing )
                {       if( disposing )
                        {       if(components != null)
                                {  components.Dispose();
                                }
                        }
                        base.Dispose( disposing );
                }
                #region Código generado por el Diseñador de Windows Forms
                /// <summary>
                /// Método necesario para admitir el Diseñador. No se puede modificar
                /// el contenido del método con el editor de código.
                /// </summary>
      private void InitializeComponent()
      {
         System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(BuscarFrm));
         this.textBox1 = new System.Windows.Forms.TextBox();
         this.etiquetaLbl = new System.Windows.Forms.Label();
```

```
            this.aceptarBtn = new Joaqs.UI.XpButton();
            this.cancelarBtn = new Joaqs.UI.XpButton();
            this.pictureBox1 = new System.Windows.Forms.PictureBox();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).BeginInit();
            this.SuspendLayout();
            //
            // textBox1
            //
            this.textBox1.Location = new System.Drawing.Point(91, 20);
            this.textBox1.MaxLength = 10;
            this.textBox1.Name = "textBox1";
            this.textBox1.Size = new System.Drawing.Size(112, 20);
            this.textBox1.TabIndex = 0;
            //
            // etiquetaLbl
            //
            this.etiquetaLbl.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.etiquetaLbl.Location = new System.Drawing.Point(30, 24);
            this.etiquetaLbl.Name = "etiquetaLbl";
            this.etiquetaLbl.Size = new System.Drawing.Size(55, 16);
            this.etiquetaLbl.TabIndex = 2;
            this.etiquetaLbl.Text = "Cedula";
            //
            // aceptarBtn
            //
            this.aceptarBtn.Location = new System.Drawing.Point(216, 16);
            this.aceptarBtn.Name = "aceptarBtn";
            this.aceptarBtn.Size = new System.Drawing.Size(88, 24);
            this.aceptarBtn.TabIndex = 3;
            this.aceptarBtn.Text = "Buscar";
            this.aceptarBtn.Click += new System.EventHandler(this.aceptarBtn_Click);
            //
            // cancelarBtn
            //
            this.cancelarBtn.Location = new System.Drawing.Point(320, 16);
            this.cancelarBtn.Name = "cancelarBtn";
            this.cancelarBtn.Size = new System.Drawing.Size(88, 24);
            this.cancelarBtn.TabIndex = 4;
            this.cancelarBtn.Text = "Salir";
            this.cancelarBtn.Click += new System.EventHandler(this.cancelarBtn_Click);
            //
            // pictureBox1
            //
            this.pictureBox1.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox1.Image")));
            this.pictureBox1.Location = new System.Drawing.Point(13, 24);
            this.pictureBox1.Name = "pictureBox1";
            this.pictureBox1.Size = new System.Drawing.Size(16, 16);
            this.pictureBox1.TabIndex = 5;
            this.pictureBox1.TabStop = false;
            //
            // BuscarFrm
            //
            this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);
            this.BackColor = System.Drawing.SystemColors.ControlLight;
            this.ClientSize = new System.Drawing.Size(426, 64);
            this.Controls.Add(this.pictureBox1);
```

```
        this.Controls.Add(this.cancelarBtn);
        this.Controls.Add(this.aceptarBtn);
        this.Controls.Add(this.etiquetaLbl);
        this.Controls.Add(this.textBox1);
        this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedToolWindow;
        this.Name = "BuscarFrm";
        this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
        this.Text = "Búsqueda de personas";
        ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).EndInit();
        this.ResumeLayout(false);
        this.PerformLayout();
                }
                #endregion
                #endregion
                private void cancelarBtn_Click(object sender, System.EventArgs e)
                {
                        this.Close();
                }

                // Realiza la búsqueda de la persona en la base de datos y si no la
encontramos
                // mostramos un mensaje de error
            private void aceptarBtn_Click(object sender, System.EventArgs e)
                {
                        // Creamos un objeto persona
                        Persona p1 = new Persona();
                        Persona p2 = null;
                        p1.Cdi = this.textBox1.Text;
                        p2 = formularioPrincipal.DAO.buscarPersona(p1);
                        if(p2 == null)
                        {
MessageBox.Show(this, "No se ha encontrado la persona", "Error en la búsqueda",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                        }
                        else
                        {
// Si viene desde el formulario de firma, tenemos que cargar la imagen en el signer
unicamente
                                if(Constantes.formularioCarga)
                                {
                                if(p2.Firma != null)
                                        {  MemoryStream ms = new MemoryStream();
                                        ms.Write(p2.Firma, 0 ,p2.Firma.GetUpperBound(0) +
1);

          FirmarFrm.signer.cargarImagen(System.Drawing.Image.FromStream(ms));
              }
                                        else
MessageBox.Show(this, "La persona encontrada no tiene una firma asociada", "Error en la
carga de firma", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                                        this.Close();
                                }
// En caso contrario (vamos hacia edicion de persona). Tenemos que abrir el formulario de
PersonaFrm
                                else
                                {
                        PersonaFrm f = new PersonaFrm(this.formularioPrincipal, this.tipo,
p2);
```

```
                                        this.Close();
                                        f.Show();
                                }
                        }
                }
        }
}
```

FORMULARIO ESPERA

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
namespace RecFirma.Formularios
{
        /// <summary>
        /// Descripción breve de EsperaFrm.
        /// </summary>
        public class EsperaFrm : System.Windows.Forms.Form
        {
                /// <summary>
                /// Variable del diseñador requerida.
                /// </summary>
                private System.ComponentModel.Container components = null;
                public EsperaFrm()
                {       //
                        // Necesario para admitir el Diseñador de Windows Forms
                        //
                        InitializeComponent();
                        //
                // TODO: agregar código de constructor después de llamar a
InitializeComponent
                        //
                }
                /// <summary>
                /// Limpiar los recursos que se estén utilizando.
                /// </summary>
                protected override void Dispose( bool disposing )
                {       if( disposing )
                        {       if(components != null)
                                {       components.Dispose();
                                }
                        }
                        base.Dispose( disposing );
                }
                #region Código generado por el Diseñador de Windows Forms
                /// <summary>
                /// Método necesario para admitir el Diseñador. No se puede modificar
                /// el contenido del método con el editor de código.
                /// </summary>
                private void InitializeComponent()
                {
        System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(EsperaFrm));
        this.SuspendLayout();
        //
```

```
        // EsperaFrm
        //
        this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);
        this.BackgroundImage =
((System.Drawing.Image)(resources.GetObject("$this.BackgroundImage")));
        this.ClientSize = new System.Drawing.Size(373, 215);
        this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.None;
        this.Name = "EsperaFrm";
        this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
        this.Text = "Iniciando RecFirma 1.0 ...";
        this.ResumeLayout(false);
                }
                #endregion
        }
}
```

FORMULARIO ACERCA DE

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
namespace RecFirma.Formularios
{       /// <summary>
        /// Descripción breve de AcercaDeFrm.
        /// </summary>
        public class AcercaDeFrm : System.Windows.Forms.Form
        {       private System.ComponentModel.Container components = null;
                private System.Windows.Forms.Label label1;
                private PrincipalFrm formularioPrincipal;
                public AcercaDeFrm(PrincipalFrm f)
                {       InitializeComponent();
                        // Primero asignamos al formulario actual el formulario donde se abre.
                        this.formularioPrincipal = f;
                        this.MdiParent = this.formularioPrincipal;
                }
                #region Codigo generado por .NET
                /// <summary>
                /// Limpiar los recursos que se estén utilizando.
                /// </summary>
                protected override void Dispose( bool disposing )
                {       if( disposing )
                        {       if(components != null)
                                {       components.Dispose();
                                }
                        }
                        base.Dispose( disposing );
                }
                #region Código generado por el Diseñador de Windows Forms
                /// <summary>
                /// Método necesario para admitir el Diseñador. No se puede modificar
                /// el contenido del método con el editor de código.
                /// </summary>
                private void InitializeComponent()
                {
        System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(AcercaDeFrm));
```

```
            this.label1 = new System.Windows.Forms.Label();
            this.SuspendLayout();
            //
            // label1
            //
            this.label1.BackColor = System.Drawing.Color.Transparent;
            this.label1.Font = new System.Drawing.Font("Microsoft Sans Serif", 6.75F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label1.Location = new System.Drawing.Point(216, 167);
            this.label1.Name = "label1";
            this.label1.RightToLeft = System.Windows.Forms.RightToLeft.No;
            this.label1.Size = new System.Drawing.Size(48, 16);
            this.label1.TabIndex = 0;
            this.label1.Text = "-    2009";
            //
            // AcercaDeFrm
            //
            this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);
            this.BackgroundImage =
((System.Drawing.Image)(resources.GetObject("$this.BackgroundImage")));
            this.ClientSize = new System.Drawing.Size(376, 216);
            this.Controls.Add(this.label1);
            this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedToolWindow;
            this.Name = "AcercaDeFrm";
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Acerca de...";
            this.ResumeLayout(false);
                        }
                        #endregion
                        #endregion
            }
}
```

BIBLIOGRAFÍA INTERNET

BIBLIOGRAFÍA GENERAL

DIGITALIZACION DE FIRMAS.

- http://www.invenia.es/oai.net 2009/04/15
- http://www.qualisoft.com.br/produtos.asp 2009/04/18

DISEÑO DE SOFTWARE

- http://www.arquitectuba.com.ar 2009/05/5
- http://www.abcdatos.com 2009/05/08
- http://es.wikipedia.org 2009/05/12

RECONOCIMIENTO DE FIRMAS

- http://www.royal-tec.com 2009/02/18
- http://www.monografias.com 2009/04/05
- http://inza.wordpress.com 2009/04/05

BIBLIOGRAFÍA RELACIONADA AL TEMA

DIGITALIZACION DE FIRMAS.

- http://www.invenia.es/oai:www.recercat.net:2072/2882 2009/04/15

- http://www.qualisoft.com.br/produtos/qualifp/qualifpesp.asp 2009/04/18

DISEÑO DE SOFTWARE

- http://www.arquitectuba.com.ar/manuales-espanol/manual-visual-basic-net-gratis-espanol 2009/05/5
- http://www.abcdatos.com/tutoriales/programacion/visualbasic.net.html 2009/05/08
- http://es.wikipedia.org/wiki/Visual_Basic 2009/05/12

RECONOCIMIENTO DE FIRMAS

- http://www.royal-tec.com/royalsignature.htm 2009/02/18
- http://www.monografias.com/trabajos10.shtml 2009/04/05
- http://inza.wordpress.com/2007/03/24/digitalizacion-de-firma-manuscrita/ 2009/04/05