Índice de contenido

Parte I

Introducción

Capítulo 1

PRESENTACIÓN DEL PROYECTO

1.1. Descripción del proyecto

El proyecto consiste en la creación de una herramienta software que facilite el aprendizaje de escritura de una persona con discapacidad intelectual.

La herramienta está desarrollada para la utilización de un lápiz óptico como pueden ser las tabletas gráficas o los Tablet PC. De forma conjunta se simula un Tablet PC, utilizando las posibilidades que proporciona la plataforma sobre tinta digital y reconocimiento de escritura manuscrita.

La finalidad del proyecto es ser una herramienta de apoyo para los discapacitados a la hora de aprender a escribir de una forma que les resulte divertida y entretenida.

1.2. Alcance

La aplicación está pensada para personas con discapacidad intelectual o como herramienta de apoyo para niños durante el aprendizaje de la escritura.

1.3. Acerca de esta documentación

Esta memoria pretende ser rigurosa con los contenidos teóricos y no extenderse en descripciones exhaustivas. No pretende hacer un análisis detallado del hardware o del software empleado, pero si proporcionar nociones para el uso de la aplicación y otras posibles vías de estudio. Se ciñe en dar los conocimientos necesarios para la comprensión del desarrollo de la aplicación bajo un entorno de las características que aquí se prestan.

La memoria se organiza en partes, y ésta a su vez en capítulos que también son divididos en puntos, de esta forma se establece un orden lógico de lectura por temas.

Las partes de la memoria son:

- **Parte I. Introducción:** Breve presentación del tema que aborda el proyecto, junto con los objetivos del mismo, a la vez que se hace una pequeña descripción del contenido y estructura de la memoria.

- **Parte II. Fundamentos Teóricos:** Desarrollo de los contenidos teóricos necesarios para la comprensión del proyecto. Son los siguientes:

  Escritura en alumnos con discapacidad: Contexto de desarrollo de la aplicación, breve descripción de discapacidades, cómo acercar las nuevas tecnologías de la información y comunicación a este tipo de alumnado, normas mundiales de accesibilidad a las plataformas informáticas y adaptaciones de los materiales didácticos para el uso del ordenador.

  Herramientas informáticas y discapacidades: Tabletas Gráficas: Definición y características más importantes, Tablet PC: Definición, características, campos de aplicación, reconocimiento de escritura en Tablet PC y Windows XP Tablet PC Edition..

  Estudio previo sobre tinta digital, reconocimiento de escritura manual, fundamentos, alternativas y posibles campos de estudio. Centrándonos en la librería usada para la implementación Microsoft.Ink.

- **Parte III. Desarrollo de la aplicación:** Análisis, diseño, implementación, requisitos software previos de la aplicación objeto del proyecto y pruebas.

- **Parte IV Manual de Usuario:** Guía pormenorizada de instalación y manejo de la aplicación.

- **Parte V. Conclusiones**

- **Apéndices:** Plataforma .NET, contenidos del CD-ROM y Glosario de Términos

Capítulo 2

OBJETIVOS PROPUESTOS

2.1. Objetivos Propuestos

A continuación mencionamos los objetivos que nos hemos propuesto:

- Elaboración de un estudio de frameworks específicos y emuladores proporcionados por los fabricantes enfocados a facilitar el aprendizaje de escritura manual y el reconocimiento de la misma mediante el uso de una tableta gráfica para personas con discapacidad intelectual o problemas motores.
- Realización de una aplicación que permita a las personas con discapacidad intelectual o problemas motores un aprendizaje más sencillo y divertido de la escritura.
- Como consecuencia del anterior objetivo, conseguir mayor independencia del usuario de nuestra aplicación y una mayor integración de éste en la sociedad actual.
- El desarrollo de una aplicación real, para usuarios finales de fácil manejo.
- Fomentar el trabajo en equipo, tanto entre nosotros como con algunos de los usuarios finales.
- Ayudar a mejorar la calidad de vida de estas personas con discapacidad como la de las personas de su entorno.
- Implicar al Centro Obregón y a centros con características similares, en el desarrollo y uso de nuevas tecnologías para ayudar a la integración de las personas discapacitadas en nuestra sociedad.
- Utilizar una metodología de desarrollo de software que permita la creación de esta aplicación.
- Aplicar correctamente los conocimientos y técnicas adquiridas a lo largo de estos años.
- Conocer un entorno de trabajo específico como es el desarrollo de aplicaciones en un nuevo lenguaje C #, y una nueva forma de trabajar la reutilización de código y librerías proporcionadas por los fabricantes dentro de la filosofía de trabajo del grupo GIRO.

Parte II

Fundamentos Teóricos

Capítulo 3

CONTEXTO DE LA APLICACIÓN: ESCRITURA EN ALUMNOS CON DISCAPACIDAD

3.1 Tipos de Discapacidades

A continuación haremos una breve descripción de las trece discapacidades más comunes, sólo unas breves pautas ya que no somos profesionales de la materia y en cada caso concreto se deben analizar las necesidades de aprendizaje de cada niño y adaptarse a ellas. En la mayor parte de ellas observaremos que el niño presenta trastornos de aprendizaje dentro de los cuales podemos enmarcar la dificultad para aprender a escribir.

3.1.1 Autismo/Trastorno Generalizado del Desarrollo (PDD)

Generalmente evidente antes de los tres años de edad, tanto el autismo como el PDD son trastornos neurológicos que afectan la habilidad del niño en cuanto a comunicación, comprensión del lenguaje, juego y su relación con los demás.

Incidencia: Ocurre de 5 a15 por cada 10.000 nacimientos, cuatro veces más común en niños que en niñas.

3.1.2 Desorden Deficitario de la Atención / Hiperactividad

Condición que hace difícil que una persona pueda sentarse tranquila, controlar su conducta y poner atención, generalmente comienzan antes de los 7 años.

Incidencia: 5 de cada 100 niños tienen AD/HD. Los niños son tres veces más propensos que las niñas

3.1.3. Epilepsia

Condición física que ocurre cuando hay un breve pero repentino cambio en el cerebro, las células cerebrales no están funcionando bien, la conciencia, movimientos, o acciones de una persona pueden alterarse por un breve periodo de tiempo.

Incidencia: 125.000 casos nuevos son descubiertos cada año.

3.1.4 Espina Bífida

Quiere decir partidura en la espina, o sea que la columna vertebral no se ha cerrado completamente. Hay tres tipos de espina bífida (varían de leve a severo) y son:

- *Espina Bífida Oculta*: Una apertura en una o más de las vértebras de la columna espinal,m sin ningún daño aparente a la médula espinal.
- *Meningocele:* Los meninges (cobertura protectora que rodea la médula espinal) se han salido a través de una apertura en las vértebras, en un saco llamado el "meningocele".
- *Myelomeningocele:*Esta es la forma más severa de espina bífida, en la cual una porción de la médula espinal sobresale a través de la espalda.

Pueden tener dificultades para poner atención en la clase, en la comprensión o expresión, en leer y en aritmética. En el caso de myelomeningocele puede incluir debilidad muscular o parálisis bajo el área donde ocurre la apertura, acumulación anormal de líquidos en el cuerpo y por tanto en el cerebro lo que se conoce como hidrocefália.

Incidencia: Aproximadamente el 40% de la población puede tener espina bífida oculta, aunque muy pocas personas llegan a saberlo ya que carece de síntomas. Los otros dos tipos ocurren 1 de cada 1000 nacimientos.

3.1.5 Impedimentos Visuales

Los términos vista parcial, baja visión, legalmente ciego y totalmente ciego son utilizados en el contexto educacional para describir los estudiantes con impedimentos visuales. Estos se definen de la siguiente manera:

- *Vista parcial* indica que algún tipo de problema visual ha resultado en la necesidad de servicios de educación especial.
- *Baja Visión* se refiere generalmente a algún impedimento visual severo, individuos con cierto grado de visión pero que no pueden leer a una distancia normal aun con la ayuda de gafas.
- *Legalmente ciego* indica que una persona tiene menos de 20/200 en el ojo mas fuerte o un campo de visión limitado (20 grados como máximo).
- Los estudiantes *totalmente ciegos* aprenden mediante el alfabeto braille u otros medios visuales.

Incidencia: Los impedimentos visuales ocurren en 12.2 de cada 1000 individuos menores de 18 años. Los impedimentos visuales severos ocurren en .06 de cada 1000 individuos.

3.1.6 Lesión Cerebral Traumática

Es una herida en el cerebro causada generalmente por un golpe en la cabeza, esta herida puede cambiar cómo la persona actúa, se mueve y piensa. También puede cambiar cómo el alumno aprende y actúa en la escuela.
Como consecuencia pueden tener una o mas discapacidades, físicas como problemas al hablar, oír y usar otros sentidos, dificultades en pensar, memoria, concentración, etc.

Incidencia: Más de un millón de niños reciben lesiones cerebrales cada año, de ellos 30.000 tienen discapacidades para toda la vida como resultado de la lesión cerebral.

3.1.7 Parálisis Cerebral

Condición causada por heridas a aquellas partes del cerebro que controlan la habilidad de mover los músculos y el cuerpo. La herida ocurre a menudo antes del nacimiento, durante el parto o en los primeros días de vida. La parálisis cerebral puede ser leve, el niño es torpe de movimientos, moderada puede significar que el niño camina cojeando y severa puede afectar todos los aspectos de las habilidades físicas del niño. A veces pueden presentar problemas del aprendizaje con problemas con el oído o visión (problemas sensoriales) o retraso mental. No empeora con el tiempo y la mayoría de los niños tienen una longevidad normal.
Existen tres tipos principales de parálisis cerebral:

- *Parálisis cerebral espástica*, condición en la cual hay demasiado tono muscular o músculos apretados, es la más común.
- *Parálisis cerebral atetoide*, puede afectar los movimientos del cuerpo entero con lentos movimientos incontrolados.
- *Parálisis cerebral mixta*, combinación de síntomas de las dos anteriores.

Otras palabras se usan para definirla:

- Diplegia, sólo las piernas son afectadas.
- Hemiplegía, la mitad del cuerpo es afectada (por ejemplo brazo y pierna derechos).
- Quadriplegía, ambos brazos y piernas son afectados, a veces incluyendo músculos faciales y torso.

Incidencia: Cada año aproximadamente 1.500 niños preescolares son diagnosticados con parálisis cerebral.

3.1.8 Problemas del Aprendizaje

Un problema del aprendizaje es un término general que describe problemas del aprendizaje específicos. Puede causar que una persona tenga dificultades aprendiendo y usando ciertas destrezas. Las destrezas que son afectadas con mayor frecuencia son:lectura, ortografía, escuchar, hablar, razonar y matemática.

Incidencia: Muy comunes 1 de cada 5 personas tiene un problema de aprendizaje.

3.1.9 Problemas Emocionales

Problemas emocionales, mentales o del comportamiento se definen como una condición que exhibe una o mas características de las siguientes a través de un largo periodo de tiempo y hasta cierto grado, lo cual afecta al rendimiento educacional del niño:

- Incapacidad de aprender que no puede explicarse mediante factores intelectuales, sensoriales, o de la salud.
- Incapacidad de formar o mantener relaciones interpersonales con los compañeros y profesores.
- Comportamiento o sentimientos inapropiados, bajo circunstancias normales.
- Un estado general de descontento o depresión.

Características: hiperactividad, agresividad, retraimiento, inmadurez y dificultades de aprendizaje.

Incidencia: Elevada

3.1.10 Retraso Mental

Es un término que se usa cuando una persona tiene ciertas limitaciones en su funcionamiento mental y en destrezas tales como comunicación, cuidado personal y destrezas sociales. Estas limitaciones causan que el niño aprenda y se desarrolle mas lentamente que un niño típico.
Se diagnostica observando la habilidad del cerebro de la persona para aprender, pensar y resolver problemas (funcionamiento intectual), y si la persona tiene las destrezas que necesita para vivir independientemente (conducta adaptativa).
El funcionamiento intelectual tambien conocido como coeficiente de inteligencia medida promedio es 100, las personas con un coeficiente entre 70 y 75 tienen un retraso mental.

Incidencia: 3 de cada 100 personas.

3.1.11 Síndrome de Down

Es la más común y fácil de reconocer de todas las condiciones asociadas con el retraso mental, es el resultado de una anormalidad de los cromosomas 47 en lugar de 46, el cromosoma adicional cambia totalmente el desarrollo ordenado de cuerpo y cerebro.
Existen más de 50 síntomas presentes o no en cada individuo, falta de tono muscular, ojos alargados, hiperflesibilidad, manos pequeñas y anchas con una sola arruga en la palma, cuello corto, cabeza pequeña etc.

Incidencia: Se podría decir que 1 de entre 800 y 1000 nacimientos tiene esta condición.

3.1.12 Sordera y la Pérdida de la Capacidad Auditiva

Sordera se define como un impedimento del oído que es tan severo que el niño resulta impedido en procesar información lingüística a través del oído, con o sin amplificación.
Impedimento auditivo se define como un impedimento permanente o fluctuante que perjudique el rendimiento escolar del niño.
Incidencia: Muy común ya que afecta a individuos de todas las edades, en mayor o menor grado, sin olvidar que muchos alumnos con otras discapacidades también están afectados por esta.

3.1.13 Trastornos del Habla y Lenguaje

Se refiere a los problemas de la comunicación u otras áreas relacionadas, tales como las funciones motoras orales. Estos atrasos y transtornos varían desde simples substituciones de sonido hasta la inhabilidad de comprender o utilizar el lenguaje o mecanismo motor-oral para el habla.

Incidencia: Elevada.

3.2 Informática y Educación Especial

La utilización de la informática en Educación Especial requiere adaptar nuestros ordenadores a toda una amplia gama de discapacidades. En primer lugar, y dentro de lo que podemos denominar como hardware (soporte físico del ordenador) conviene tener en cuenta las modificaciones que debemos de llevar a cabo en nuestro ordenador como son adaptaciones al teclado, conmutadores e interruptores, digitalizador de voz, emulador de teclado, teclado de conceptos, ratón, emulador de ratón, lápiz de tabletas gráficas etc.

Muchas personas con discapacidad utilizan dispositivos alternativos de entrada. El ordenador debe disponer de un teclado y un ratón independientes o la posibilidad de colocar un teclado y/o un ratón externos adicionales, pues de esta forma se podrán colocar un teclado o ratón especiales adaptados a las características del usuario. En ocasiones habrá que colocarlos sobre un atril o soporte especial para controlarlos con una ayuda técnica, como la varilla bucal.

Merece hacer una pequeña referencia a un grupo de adaptaciones de hardware, entre el que encontramos a "Intellikeys", un teclado alternativo con láminas intercambiables para distintos usos. La utilización de estos teclados especiales necesita también un tipo determinado de software como el "Overlay Maker", "Intellipics", "Click it"y "Mathpad" de matemática. Otro software que se puede utilizar con este tipo de teclado es el "Board Maker", que permite al profesor diseñar bases de datos para crear símbolos y material didáctico. Asimismo, se pueden hacer plantillas en lenguaje Braille para adecuarla a personas ciegas. Además, permite la instalación de interruptores y funciona con cualquier software educativo ya sea comercial o especial porque es un sustituto del teclado convencional. Posee un cable de conexión y cinco láminas. A este teclado se le puede aplicar el Switch de Ablenet.Inc, un interruptor sencillo que permite imitar los movimientos de un mouse con pulsaciones sencillas y dobles. La forma de maniobrar este dispositivo es totalmente intuitiva.

Las dificultades de las personas con **problemas físicos** suelen ser derivados de su falta de coordinación, su debilidad, la dificultad para alcanzar las cosas o la imposibilidad de mover alguna o algunas extremidades.

Este tipo de personas pueden o no utilizar dispositivos específicos de naturaleza variada . Algunos ejemplos son los dispositivos de seguimiento de ojos, los teclados en pantalla, los sistemas de reconocimiento de voz y los punteros alternativos (licornios, punteros de manos, etc.), quizá sea este campo en el el se enmarca la tableta gráfica utilizando el lápiz como un puntero.

Nuestro proyecto pretende ser una alternativa al teclado y ratón ofreciendo la posibilidad de escritura a mano que sea reconocida en la medida de lo posible por el ordenador.

3.3 Norma mundial de accesibilidad a las plataformas informáticas

La variedad de la problemática de acceso que se presenta en función de las diversas discapacidades, han hecho necesaria la recopilación de todos los problemas de accesibilidad en dos documentos, estructurados como dos normas de **AENOR** (Asociación Española de Normalización y Certificación), que contemplan todos los posibles problemas detectados para discapacidades visuales, auditivas, físicas y psíquicas, en lo referente al interfaz de usuario, tanto del soporte lógico (software), como del soporte físico (hardware),

además de a la documentación asociada a estos productos.

La norma que afecta al hardware se llama "*Informática para la salud. Aplicaciones informáticas para personas con discapacidad. Requisitos de accesibilidad de las plataformas informáticas. Soporte físico.*" y tiene como número de norma **139.801**. En ella se contemplan los aspectos de accesibilidad de la unidad central, la pantalla, el teclado, el ratón y los periféricos.

La norma que afecta al software se denomina "*Informática para la salud. Aplicaciones informáticas para personas con discapacidad. Requisitos de accesibilidad de las plataformas informáticas. Soporte lógico.*" y tiene como número de norma **139.802**. En ella se describen los problemas de accesibilidad separando los que afectan al sistema operativo, a las aplicaciones y a Internet.

3.4 Los profesionales ante la discapacidad motórica

La discapacidad física exige un abordaje interdisciplinar en el que confluyen profesionales procedentes de muy diversos ámbitos, se deben adoptar por tanto posturas abiertas y flexibles para incidir en el campo de la deficiencia motórica. Todos los profesionales deben trabajar en base a un denominador común , la atención a la persona con discapacidad física. Siendo el objetivo posibilitar una mejor calidad de vida. Es esencial mantener una postura abierta y recurrir a quienes mejor puedan orientarnos y ayudarnos a planificar nuestra intervención sea cual sea nuestro campo de acción.

3.5 Las adaptaciones de materiales didácticos

Para los profesionales de la informática el reto es facilitar el acercamiento a la informática en general. Será necesario en muchos casos adaptar los materiales y útiles escolares para su manipulación, en definitiva adaptar hardware y software teniendo presente como objetivo mejorar la calidad de vida de las personas con discapacidades de distintos tipos.

Este apartado pretende hacer un recorrido por los diferentes recursos didácticos más utilizados en las escuelas, que posibilitan el uso y manejo de algunos materiales en aquellas personas con afectación motórica en sus posibilidades de manipulación.

En los niños con **deficiencias motóricas** (debidas a lesión cerebral por ejemplo) se van a ver enlentecidas y/o dificultadas todas las posibilidades de motilidad voluntaria.

En general, nos vamos a encontrar con una serie de dificultades en la manipulación que pueden oscilar entre:

- Quien no utiliza nada sus miembros superiores (de forma transitoria o permanente).
- Quien dirige la mano hacia los objetos pero no puede asirlos.
- Quien coge los objetos pero no puede soltarlos.
- Quien puede asir los objetos.

Antes de pormenorizar en cada una de las adaptaciones a realizar sobre los útiles clásicos de escritura, nos detendremos un poco en los materiales didácticos necesarios para esos alumnos en los que no se ha establecido como objetivo el conseguir las técnicas instrumentales, sino habilidades previas mucho más básicas como pueden ser:

- El mantenimiento del contacto visual
- Una atención mínima
- La direccionalidad de la vista, o del movimiento de un brazo, pie,...
- El control del cuello o la postura erguida de la cabeza
- El inicio de la comunicación mediante la negación y la afirmación SI/NO
- Aprendizaje de órdenes que impliquen un vocabulario básico, nociones espacio-temporales, acciones...

Para todo este tipo de situaciones son muy válidos materiales como sonajeros que el alumno pueda manipular, móviles colgados del techo que le obligarán a trabajar la direccionalidad de la vista, muñequeras con cascabeles, juguetes adaptados, juegos didácticos como puzzles, dominó, juegos de serializaciones, clasificaciones, numeraciones etc.

Todo este material claro esta adaptado en su tamaño, color, forma, textura, peso,...por ejemplo asideros en las piezas de un puzzle.

3.5.1 Qué hacer cuando no pueden manejar los útiles de escritura

En cualquier caso será necesario como siempre elegir la opción más acorde con las necesidades educativas especiales del niño sólo mencionaremos algunas opciones.

- **Materiales manipulables** para los aprendizajes básicos, cubos y figuras geométricas de diferentes formas y tamaños que puedan adaptarse a las posibilidades de prensión de cada niño.
- **Materiales imantados** para utilizar sobre pizarras férricas
- **Sistemas de imprentillas** , ajustados a la presa de la mano pueden posibilitar una ejecución gráfica alternativa al lapicero.
- **Ayudas técnicas a la comunicación escrita**, como la máquina de escribir electrónica y el ordenador. Es en este apartado donde se enmarcaría nuestro proyecto.
  Se trata de posibilitar la iniciación en los primeros niveles del proceso de enseñanza-aprendizaje de la lectoescritura y el cálculo, ver si podemos realizar seriaciones, clasificaciones, agrupamientos, establecimiento de relaciones que implique nociones y conceptos espaciales, temporales, semejanzas, diferencias, unión de letras, palabras, operaciones aritméticas,... y todo sin utilizar papel y lápiz.
  Nuestra aplicación podría utilizarse como una **ayuda técnica a la escritura**.

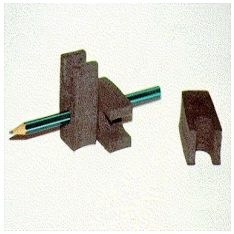
3.5.2 Qué hacer cuando pueden coger con alguna adaptación los útiles escolares

Si el niño esta posibilitado ante la actividad voluntaria de asir, es decir la prehensión voluntaria, deberemos conocer bien sus posibilidades es decir que tipo de presa es capaz de realizar (en puño, cilíndrica, esférica) o que tipo de pinza ha logrado (digitar, palmar, entre los dedos, lateral) y en función de ello facilitar la adaptación del lapicero mas acorde a sus necesidades.

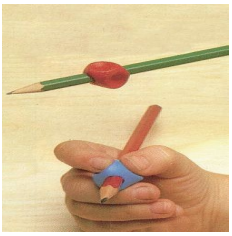
Este punto nos parece interesante ya que la tableta gráfica se maneja a través de un lápiz y con toda probabilidad será necesario utilizar adaptadores como los que se muestran a continuación.

Dependiendo de las posibilidades y características de la prensión podemos tener distintos tipos mostrados en las siguientes figuras.

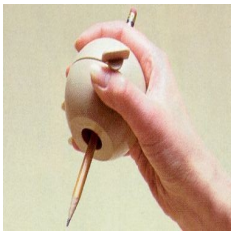
Adaptador de agarre



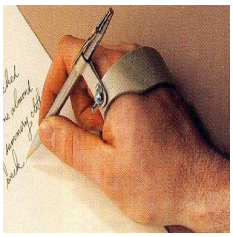
Engrosador de lapiceros



Asidero de lápiz

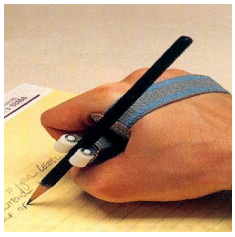


Adaptador esférico



Ayuda para la escritura 1

Ayuda para la escritura 2



Ayuda para la escritura 3

Figura 3.1. Diferentes adaptadores de lapicero según las características de la prensión

Alguna de estas posibilidades puede ser usada para manipular el lápiz de la tableta gráfica si es necesario, teniendo en cuenta las características de la propia tableta y del usuario de la misma.

También se podrá colocar la tableta en la posición más adecuada a cada caso utilizando atriles por ejemplo.

3.5.3 Qué hacer cuando no pueden controlar los movimientos anormales de sus miembros superiores

Los movimientos anormales en las personas con parálisis cerebral por ejemplo se deben a varias circunstancias:

- Algunas personas no hacen movimientos anormales en reposo. Al intentar moverse, la espasticidad en el grupo de los músculos antagonistas distorsiona el movimiento.
- Las que tienen afectación cerebelosa presentan un temblor y falta de coordinación cuando van a hacer un movimiento intencional.
- Las personas con afectación extrapiramidal presentan movimientos anormales incluso en reposo y pueden acentuarse en situaciones de tensión emocional.

Para estos casos y orientado siempre a poder manejar la tableta junto con el lápiz de la misma podremos usar *pulseras lastradas* (muñequeras con peso) por ejemplo para compensar este tipo de movimientos, o proporcionar una base de sustentación del antebrazo. Las siguientes figuras nos muestran algún ejemplo.

Figura 3.2. Muñequera con peso

3.5.4 Qué hacer cuando sus miembros superiores no son funcionales

En personas con deficiencias motóricas graves cabe intentar trabajar con un *puntero cabezal o licornio.* Exige de un buen control de los movimientos de la cabeza y de la musculatura del cuello. Incluye un apéndice o brazo frontal al que se pueden acoplar diversos accesorios para realizar actividades como escribir a máquina, dibujar, coger objetos etc, también en este caso podremos intentar adaptar la sujeción del lápiz de la tableta gráfica. Podemos ver distintos tipos en las siguientes figuras

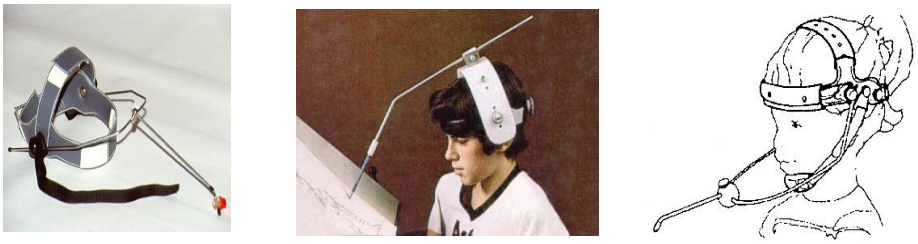


Figura 3.3. Diferentes licornios

También podemos probar algún tipo de adaptación bucal, e incluso recurrir al manejo y control de los pies si es posible.

Figura 3.4 Varilla para pie

3.6 Herramientas informáticas y discapacidades

3.6.1 Tabletas Gráficas

3.6.1.1 Características generales

Las tabletas gráficas o digitalizadoras son unos dispositivos de entrada que permiten digitalizar figuras y gráficos vectoriales. Están formadas generalmente por una superficie plana que contiene en su interior una *rejilla* formada por cientos de líneas de cobre por las que circulan corrientes perpendiculares independientes.

Figura 3.5 Tableta Gráfica

Para dibujar se utiliza un ***lápiz especial o trazador***, en cuyo interior se encuentra un pequeño electroimán que modifica el voltaje en cada una de las rejillas de la tableta proporcionalmente a su posición sobre ella, lo que permite a ésta establecer la ***posición*** del puntero del lápiz, su ***orientación e inclinación*** y la ***presión*** con que el usuario aprieta el lápiz.

La tableta está conectada al ordenador donde un programa gráfico va recogiendo los puntos y trazos marcados por el lápiz sobre la tableta, presentándolos en pantalla y asumiéndolos a todos los efectos como propios y pertenecientes al gráfico que en ese momento se encuentra activo en el mismo. Generalmente funcionan con posicionamiento absoluto y el área activa representa la pantalla completa del ordenador.

Las tabletas suelen presentar un área activa mínima de tamaño DIN A5 (101 x 76 mm ó 3 x 4 pulgadas) y una resolución que ronda las 1.500 lpi (líneas por pulgada), aunque también es normal encontrarlas de tamaño DIN-A4 con 3000 lpi y más. Se conectan al ordenador a través de cualquier puerto válido al efecto (generalmente un puerto USB) o por medios inalámbricos, que presentan la ventaja de eliminar las posibles molestias causadas por los cables.

El ***lápiz gráfico es sensible a la presión*** (son capaces de reconocer cientos de niveles de presión), pudiendo conseguirse con él multitud de efectos, como la simulación de dibujo natural sobre papel o de escritura caligráfica. Este suele tener forma de lápiz o cursor, y está unido al resto del sistema por un cable flexible o de forma inalámbrica. En el último caso el cursor tiene una ventana cerrada con una lupa, en cuyo interior se encuentra embebida una retícula en forma de cruz para señalar o apuntar con precisión el punto a digitalizar. El mando puede disponer de uno o varios pulsadores para controlar la modalidad de funcionamiento, forma de transmisión y selección de opciones del programa que gestiona la digitalización.

Este dispositivo resulta un complemento perfecto para dibujar y pintar directamente en el ordenador, siendo muy usadas en asociación a programas de retoque, de ilustración y de dibujo técnico y CAD, programas para los que el teclado y el ratón son dispositivos demasiado limitados para dibujar formas con precisión.

El lápiz suele pesar entorno a 12 gramos, lo que permite trabajar cómodamente durante un tiempo prolongado. El manejo del lápiz y su ligero peso suponen una ventaja decisiva frente a otros dispositivos de entrada también para personas con discapacidad.

Existen multitud de tabletas gráficas disponibles en el mercado, desde las más simples a las que ofrecen funcionalidades avanzadas. Como norma general, cuanto mayor es el área de trabajo, mayor es su resolución y mayor es su precio, aunque para la mayoría de los trabajos las más pequeñas son perfectamente adecuadas destacando entre los fabricantes de estos dispositivos Wacom y Genius.

Figura 3.6 Diferentes tabletas gráficas

Algunos modelos disponen de ***ExpressKeys*** son una serie de teclas agrupadas a la izquierda y a la derecha del área activade la tableta. Con estas teclas integradas, el usuario puede acceder directamente a funciones como cortar, copiar, pegar y deshacer sin tener que cambiar constantemente al teclado.

También pueden incorporar ***Touchpad,*** funciona en principio como el existente en muchos ordenadores portátiles, reaccionando al contacto con el dedo. Si su disposición es vertical, puede deslizar el dedo arriba y abajo, por ejemplo, para controlar el alcance del zoom en programas de edición de imágenes . También puede avanzar y retroceder en la barra de tiempo de programas de animación. Y, por supuesto, también resulta útil en otros programas que no son de diseño gráfico: permitiendo desplazarse fácilmente por documentos de texto o editores HTML con un simple movimiento del dedo. Generalmente se encuentra situado a ambos lados de la tableta, con lo que es accesible tanto para diestros como para zurdos.

La tecnología de lápiz sin cable y sin pilas patentada por Wacom se denomina ***"Penabled"*** y le ofrece un producto fiable y de alta calidad que casi no requiere mantenimiento. Todos los dispositivos de entrada lápiz o ratón si dispone de él, son sin cable y sin pilas, siendo la tableta la que se encarga de suministrarles la energía necesaria.

La mayoría de los profesionales de medios digitales emplean una tableta gráfica en combinación con un teclado convencional, puesto que resulta mucho más rápido y sencillo manejar muchos programas con accesos directos de teclas, como por ejemplo, "CTRL-C" para copiar elementos en el portapapeles.

Con el lápiz se puede retocar la fotografía con total precisión, una clara ventaja frente a la corrección automática de fotografías y la manera más sencilla de obtener fantásticas imágenes. Trabajar con el lápiz ergonómico sin cable ni pilas resulta especialmente cómodo e intuitivo.

En particular el manejo del teclado puede resultar una barrera difícil de salvar para las ***personas discapacitadas***, el uso de la tableta gráfica puede ayudar a manejar un ordenador a las personas que tienen este problema, ya que es posible utilizar el lápiz de modo similar al ratón en las aplicaciones e incluso mejor ya que sólo hay que "seleccionar" usando el lápiz.

El sistema se instala rápida y fácilmente y, además, es compatible con el software ya instalado en el ordenador practicamente en todos los casos.

3.6.1.2 Ejemplo de Tableta Gráfica

Para la realización de este proyecto hemos utilizado el modelo **Graphire3 de WACOM**, señalaremos a continuación algunas de sus principales características y su manejo.

El lápiz es una herramienta de mano alzada, sin bateria y sensible a la presión, dispone de un botón tipo DuoSwitch (dos botones en uno) que puede usarse para señalar, hacer clic, doble clic y arrastrar, cuya mina es reemplazable, en algunas aplicaciones el extremo contrario del lápiz actua como borrador.

Figura 3.7 Tableta Gráfica Graphire 3.

El ratón inalámbrico dispone de tres botones que se pueden personalizar para programar difentes funciones, funciona como un ratón convencional, aunque no todas las tabletas disponibles en el mercado incorporan ratón para usar la tableta.

Para **hacer clic** basta con dar un ligero golpe en la tableta con la punta del lápiz o presionar, puede seleccionarse en el panel de control Pen Tablet que se produzca o no sonido. Para **hacer doble clic** deberemos dar dos golpes ligeros en el mismo punto. Para ambos casos podemos configurar también el botón del lápiz, generalmente se corresponde con clic izquierdo y derecho del ratón convencional.

La **acción de arrastrar** se utiliza para seleccionar y mover objetos en la pantalla, debemos señalar el objeto que deseamos arrastrar, presionar la punta del lápiz sobre él y deslizar sobre la superficie del área activa de la tableta.

La sensibilidad de la punta y del borrador puede ajustarse suave o firme dependiendo del control que queramos realizar y de la aplicación donde estemos manejando el lápiz.

El **área activa de la tableta** es una representación proyectada de la pantalla del ordenador, en la que cada uno de los puntos del área activa se corresponde exactamente con un punto de la pantalla de visualización. No es necesario tocar la tableta con la punta del lápiz ya que en cuanto se aproxima el lápiz a la superficie del área activa se detecta por proximidad.

La tableta dispone de un LED que varia de color cuando el lápiz o el ratón estan dentro del área activa.

Es necesario habituarse al uso de la tableta, hay que coordinar los ojos y la mano no hay que olvidar que el área activa es una representación proyectada de la pantalla del ordenador y que cada vez que se sitúe el lápiz en el área activa el cursor de la pantalla saltará al punto correspondiente de la misma, esto es lo que se denomina "posicionamiento absoluto" para hacer posible el trazado, aunque también es posible que el área activa proyecte sólo una parte de la pantalla.

Figura 3.8 Tableta y Lápiz

El panel de control Pen Tablet permite personalizar las funcionalidad del lápiz en cuanto a nivel de presión, sonido de clic, cantidad de presión y velocidad. De igual modo podemos personalizar el ratón en cuanto a uso de los botones y velocidad.

3.6.2 Tablet PC

3.6.2.1 Características generales

Es un equipo similar a un ordenador portátil, tiene la particularidad de que ofrece una pantalla táctil en la que se puede escribir. De esta forma, el usuario puede prescindir del teclado y el ratón, aunque también los puede utilizar si lo desea. Además, es mucho más ligero que un portátil y ocupa menos espacio, propugnado por Microsoft y otros fabricantes.

Existen una gran variedad de diseños de Tablet PC para que pueda elegir el que mejor se ajuste a sus necesidades. Hay dos tipos de modelos ligeros—el modelo pizarra y el modelo convertible. Algunos modelos sólo pesan 1,3 kg. Todos los modelos disponen de la característica de acoplamiento sin previo aviso. El modelo pizarra es ultraplano e incluye un teclado externo fácil de transportar, lo que permitirá disfrutar de lo último en movilidad. El modelo convertible tiene un aspecto similar a un portátil pequeño con una pantalla y un teclado abatibles. La mayoría de los equipos Tablet PC convertibles tienen pantallas que giran y se apoyan en el teclado, lo que facilita la lectura y la escritura.

Tablet PC dispone del rendimiento y las características de los equipos portátiles actuales, pero con más opciones que permiten una mayor movilidad. Los modelos ultraligeros y características tales como la compatibilidad integrada con redes inalámbricas, una duración prolongada de la batería y la rotación instantánea de la pantalla permiten utilizar Tablet PC en muchas más situaciones que antes.

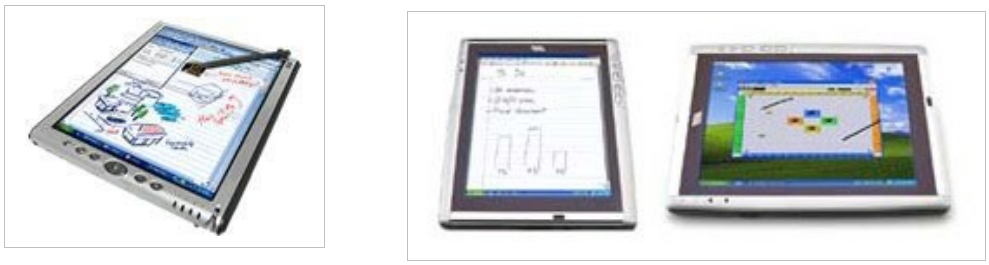


Figura 5.1 Tablet PC (Pizarra)

Figura 5.2 Tablet PC (Portátil convertible)

Estos dispositivos utilizan un sistema operativo que es una evolución del Windows XP Profesional, optimizado para trabajar con procesadores mobile que consumen menos energía (Windows XP para Tablet PC). El software especial que nos proporciona el sistema operativo, permite realizar escritura manual, y llevan una especie de lápiz para poder tomar notas a mano alzada y dibujar sobre la pantalla.

Como ya se dijo anteriormente, un Tablet PC es igual que un ordenador portátil al que se le ha añadido las siguientes funcionalidades:

- Componentes de escritura manual
- Selección táctil de opciones
- Reconocimiento de escritura

El reconocimiento de escritura lo trataremos más adelante ya que es la base de nuestro proyecto.

3.6.2.2 Campos de Aplicación

Al ser un dispositivo de gran capacidad de proceso y tener una pantalla grande, los campos de aplicación son muy amplios. A continuación se describen algunos en los que con un Tablet PC, se produce una mejora sustancial:

- Control/Gestión in situ: Es ideal para arquitectos e ingenieros que tienen que tomar notas y realizar procesos de cálculo y medida a pie de obra. Al tener integrado sistemas de conexión inalámbrica, permite volcar la información de una forma rápida en el ordenador central.

- Tablet PC es idóneo para los trabajadores móviles de las empresa que utilizan actualmente equipos portátiles. Puesto que cada vez son más los trabajadores de la información que emplean gran parte de su tiempo fuera de la oficina, frecuentemente fuera de su mesa de trabajo, tienen necesidades específicas que los equipos portátiles no satisfacen por ello la demanda de equipos portátiles continúa creciendo.

- Tablet PC ofrece la solución ideal para los trabajadores de la información. Gracias a las características de acoplamiento sin previo aviso y arranque instantáneo. La rotación de pantalla instantánea permite introducir datos discretamente en posición horizontal o vertical, puede utilizarse como si fuera un portapapeles y escribir con su lápiz digital.

- Los usuarios de Tablet PC podrán hacer funcionar el ordenador con el lápiz digital además de utilizar los métodos tradicionales de introducción de datos como el teclado y el ratón. Del mismo modo, el lápiz puede utilizarse para realizar las mismas tareas que se hacen con el ratón; moverse sobre la superficie, seleccionar las herramientas del menú, mover los objetos y cambiarlos de tamaño y activar programas. Cómo sucede con los lápices normales, el usuario puede elegir el color y el grosor del trazo de la tinta además de poder subrayar y escribir en negrita.

- **La ONCE y HP desarrollan una experiencia educativa basada en Tablet PCs adaptados**
  Las nuevas tecnologías mejoran la integración educativa de niños con ceguera o discapacidad visual. Esta iniciativa pionera en España permite a los alumnos realizar tareas que les ayuden a mejorar su aprendizaje. La ONCE y HP han desarrollado una experiencia educativa piloto dirigida a niños con ceguera o discapacidad visual. Esta iniciativa quiere analizar la viabilidad del uso del Tablet PC (o cuadernos digitales) por estos escolares incorporándolo como una herramienta de trabajo habitual que les permita realizar tareas que les ayuden a mejorar su aprendizaje. Al mismo tiempo, les proporciona motivación para el estudio.

  El lápiz óptico les permite trabajar simultáneamente con fichas en relieve o practicar la escritura, por ejemplo. La posibilidad de posicionar la pantalla en vertical y horizontal, escribir directamente, adecuar la iluminación, distancia de trabajo, postura, eliminación de reflejos, etc., sobre ella también supone un gran avance para estos escolares

3.6.2.3 Reconocimiento de Escritura en Tablet PC

El reconocimiento de escritura en los Tablet PC es uno de los puntos más interesantes. En los equipos convertibles, resulta sencillo introducir información gracias a que disponemos de un teclado que podremos utilizar siempre que deseemos. Sin embargo, en los modelos tipo pizarra, la única alternativa que queda es conectar un teclado y ratón externos para poder utilizarlos como un equipo convencional.

Esta es la razón por la cual el sistema de inserción de escritura es especialmente importante en esta plataforma. Para ello tenemos básicamente dos alternativas: que el equipo reconozca nuestra escritura natural o bien puntear letra a letra en un teclado virtual que se nos presenta en pantalla, algo que resulta lento y tedioso para insertar cierta cantidad de información.

En este campo en lugar de recurrir a las técnicas que se empleaban en el terreno de las PDA, Microsoft recurrió a un algoritmo capaz de reconocer, no los caracteres escritos por nosotros, sino los

movimientos de nuestra mano a lo largo de la pantalla. De esta forma, es posible identificar sin problemas la letra de dos personas con caligrafía muy diferente sin apenas entrenamiento y con un elevado índice de aciertos.

Por otro lado, este sistema de reconocimiento tiene un ligero inconveniente: los movimientos son distintos en función del idioma que se está representando. Por esta razón la versión de Windows XP Tablet PC Edition en castellano tardó más de un año en ver la luz, no sólo se trataba de traducir el sistema operativo sino de identificar y adaptar el algoritmo de reconocimiento a nuestro lenguaje.

Aún así, en la calidad del reconocimiento de escritura intervienen importantes factores como la velocidad de proceso de la máquina, la memoria disponible en el sistema así como la calidad del film sensible de la pantalla o de la tableta gráfica.

En la versión de Windows XP Tablet PC Edition se incluye como nueva característica la sensibilidad al contexto que mejora el reconocimiento de datos comunes, como direcciones de correo electrónico y URL, y permite la detección automática del ancho de caracteres y el orden de pulsaciones.

Respecto a la configuración, hemos de tener presente que el reconocimiento de escritura de manera fluida y mantenida es una tarea que precisa de unos recursos del sistema verdaderamente elevados.

3.6.2.4 Windows XP Tablet PC Edition

Puesto que la mayor parte de los Tablet PC utilizan este sistema operativo hemos creido oportuno hacer mención de algunas de sus características más importantes.

- **Una versión superior de Windows XP Professional.** Windows XP Tablet PC Edition, que se ha construido a partir del sistema operativo Windows XP Professional con mejoras específicas. Disfruta de las características, fiabilidad y potencial del último sistema operativo para PC de Microsoft así como el reconocimiento de voz y escritura.

- **Una plataforma para aplicaciones compatibles con Windows XP.** Puede ejecutar aplicaciones compatibles con Windows XP.

- **Un computador de escritorio.** Puede utilizar todos los periféricos que quiera incluido un monitor estándar, un teclado y ratón e impresoras entre otras cosas.

- **Herramientas de escritura y tinta que se pueden utilizar en Microsoft Office XP.** Utilizando el lápiz digital del Tablet PC para manejar aplicaciones del paquete Office.

- **Una plataforma sólida de desarrollo.** Microsoft le ofrece un kit de desarrollo de software (SDK) así como otros recursos para ayudar a los desarrolladores a que saquen el máximo beneficio a las herramientas de tinta y voz de la plataforma.

- **Introducción de datos con lápiz digital.** Tablet PC incluye lápices digitales para controlar la computadora e introducir textos de su propio puño y letra.

- **Panel de introducción de datos.** Es un teclado que aparece en la pantalla y cuadernos de notas donde puede escribir cuando no tiene el teclado real a mano o cuando prefiere no utilizarlo. El Panel de entrada funciona perfectamente en cualquier programa basado en

Windows, incluso en aquéllos que no son compatibles con la entrada manuscrita.

- **Microsoft Windows Journal.** Similar a un cuaderno digital para tomar notas. Pueden hacerse anotaciones durante una reunión o cualquier otra situación sin ningún tipo de problema. Ya no tendremos que escribir anotaciones en papel para mecanografiar posteriormente en el ordenador.

- **Conversión y reconocimiento de la escritura.** Las herramientas de reconocimiento de escritura ayudan en la conversión de tinta a texto ya que ayudan a localizar y corregir los errores de reconocimiento antes de introducir el texto en otra aplicación.
- **Reconocimiento de voz.** Puede utilizar la voz en lugar del ratón, el teclado o el lápiz para controlar las aplicaciones. El reconocimiento de voz es un componente esencial de cualquier Tablet PC que permite al usuario dictar contenidos o controlar las aplicaciones con su voz. En la mayoría de los casos se requiere disponer de un micrófono externo. El kit SDK incluye una interfaz del programador de aplicaciones de voz (SAPI, en sus siglas en inglés) con la que se puede ampliar la utilización de esta herramienta.

- **Añadir notas a los documentos.** Podremos añadir notas a documentos o subrayar.

- **Lectura.** Utiliza tecnología Microsoft ClearType y ofrece una presentación en alta resolución proporcionando una lectura natural y cómoda.

- **Girar la pantalla.** Puede utilizar la posición vertical para tomar notas, tal y como haría en un cuaderno de papel, o utilizar la posición horizontal sin reiniciar.

- **Gestos.** Puede hacer "gestos" con el lápiz digital. Los gestos son movimientos que se hacen con el lápiz para realizar tareas comunes. Dispone de otro medio para realizar las tareas que normalmente realizaría utilizando el ratón,, señalar, hacer clic, etc.

3.6.2.5 Otras alternativas a Microsoft

- La empresa española **Tuxum** ha creado el primer ordenador pizarra que funciona con GNU/Linux, concretamente con una metadistribución de Debian, adaptada para añadirle reconocimiento de escritura.

- El 3 de octubre de 2002 **Apple** presentó en España el innovador Mac OS X 10.2 (Jaguar), la nueva versión del sistema operativo de Apple, con más de 150 nuevas características y aplicaciones. Entre ellas la nueva tecnología de reconocimiento de escritura "**Inkwell**". La nuieva versión de Mac OS X (Tiger) también incorpora esta característica mejorada.

Inkwell

Una sofisticada tecnología de reconocimiento de escritura a mano alzada desarrollada por Apple para el famoso Newton (el PDA precursor de los actuales Palm y compañía). InkWell aporta su funcionalidad a todas las aplicaciones que corran sobre Mac OS X, o lo que es lo mismo, permitirá **escribir texto en aplicaciones** sin necesidad de utilizar el teclado , totalmente integrada en el sistema de manera que todas las aplicaciones compatibles Mac OS X pueden sacar pleno partido usando una tableta gráfica.

Inkwell permite escribir en cualquier lugar de la pantalla. Una vez reconocido, el texto aparece en el punto de inserción correspondiente, como si se hubiera escrito con el teclado. Así, cuando se trabaja con la tableta gráfica, no es necesario soltar el lápiz y coger el teclado para escribir un título, leyenda o nombre de archivo.

Permite escribir atajos de comandos clave con el lápiz como abrir y cerrar ventanas, y en general controlar aplicaciones, sin levantar el lápiz de la tableta. Aparte de palabras y números, puedes utilizar Inkwell para escribir comandos. Inkwell te ofrece un juego de Gestos fáciles de aprender y recordar que son muy cómodos para llevar a cabo tareas frecuentes, como cortar, copiar, pegar y seleccionar todo, con un solo trazo del lápiz.

En el modo de escritura, arrastra ventanas y utiliza las barras de desplazamiento y demás controles al instante. En esos lugares especiales, el lápiz se comporta como un ratón; en cualquier otro lugar, puedes pulsar y mantener presionado el lápiz brevemente para utilizarlo como ratón.
Inkwell también funciona con escritura en francés y alemán.

Capítulo 4

ESTUDIO PREVIO: TINTA DIGITAL

4.1 Consideraciones iniciales

Cuando hablamos de XP normalmente nos referimos a Windows XP Home Edition, que es el sistema operativo que viene preinstalado en la mayoría de los ordenadores actuales de sobremesa, pero está no es la única opción que propone Microsoft existen además otras versiones como son:

- **Windows XP Edición Profesional**

Mejora la edición Home en cuanto a fiabilidad, eficacia y rendimiento, capacidad multitarea, nuevo diseño visual , acceso remoto proporcionado, necesitaremos la edición profesional si queremos enlazar nuestra máquina a una red basada en dominios. Incorpora protección de datos mediante un sistema de encriptación de ficheros que impide el acceso a usuarios no autorizados, por ultimo proporciona un función completa de restauración del sistema en caso de fallo, creación de copias de seguridad y recuperación de datos mas robusta que la versión Home, aunque como desventaja hay que destacar su elevado precio.

- **Windows XP 64 bits**

Microsoft desarrolló esta edición para sacar el máximo partido a las características exclusivas del Itanium de Intel, un chip que no esta al alcance de todo el mundo, y que a menos que seamos profesionales de la industria científica no le sacaremos partido.

- **Windows XP Media Center Edition**

Esta versión del sistema operativo se encuentra sólo disponible para los consumidores americanos, entre sus prestaciones destacamos la inclusión de programas de grabación de vídeo y DVD.

- **Windows Vista**

Conocido como Longhorn será la versión de Windows más innovadora, robusta y depurada que gobernará los PCs del futuro.

- **Windows XP Tablet PC Edition**

Esta última versión Windows XP Tablet PC Edition fue diseñado a partir de Windows XP Edición Profesional para utilizar con la nueva gama de ordenadores portátiles, los Tablet PC. La principales características han sido ya mencionadas en el Capítulo precedente.

Microsoft Tablet PC funciona con Windows XP como sustrato extendiendo sus funcionalidades. Añadiendo posibilidad de reconocimiento de texto y tinta, proporciona la posibilidad de que una aplicación responda a "gestos" , eventos reconocidos por la aplicación y realizados por medio del lápiz, y herramientas de voz.

Es en esta última versión la que centra nuestro interés ya que usaremos alguna de sus características para simular el comportamiento de un ordenador Tablet PC en cuanto a reconocimiento de escritura manuscrita se refiere utilizando para ello una Tableta Gráfica conectada a un PC de sobremesa.

4.2 Tinta Digital

La superficie sobre la que se escribe en los Tablet PC es más o menos equivalente a la de un cuaderno de notas estándar. El usuario puede apoyar la mano en la pantalla mientras que escribe o mientras se está ejecutando las aplicaciones de software sin que esto afecte el funcionamiento. Se trata de una manera muy natural de utilizar Tablet PC pero que exige un diseño cuidadoso. De ahí que Tablet PC se haya diseñado para funcionar con un indicador digital en lugar de una pantalla táctil como se encuentra en una PDA (asistente personal digital) y otros dispositivos con pantalla pequeña. El indicador digital sólo acepta la información que se introduce con un lápiz especial equipado con una bobina electromagnética, sólo así tendremos un "cuaderno de notas digital".

El indicador electromagnético mejora aún más la experiencia de la tinta digital ya que evita que se produzca el contacto entre la mano del usuario y la pantalla y que mueva el cursor sin darse cuenta. Además la actividad de la pantalla es mayor ya que el usuario puede mover el cursor sin que haya contacto físico directo con la superficie de la pantalla. Gracias a esto se puede mover el cursor con rápidez y facilidad.

Desde un punto de vista técnico, la introducción de datos utilizando el lápiz y la tinta digitales se desarrolla de la siguiente manera: el sistema operativo Windows XP Tablet PC Edition capta los movimientos del lápiz sobre la pantalla, recoge y almacena estos movimientos como "tinta" y luego traslada estos golpecitos del lápiz a un "identificador" que los interpreta y convierte en escritura. Además Tablet PC ofrece soporte para *gestos*. Con tan solo hacer un gesto en la pantalla con el lápiz el usuario puede ejecutar varias tareas. Los gestos pueden utilizarse para una serie de acciones y comandos que se solicitan haciendo una marca con tinta con el lápiz en una o varias ubicaciones de la pantalla de Tablet PC.

4.2.1 Fundamento de la tinta digital

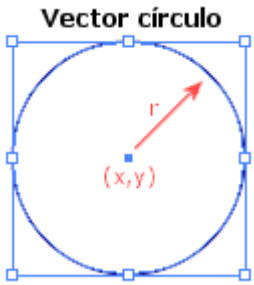
La tinta se almacena como una serie de ecuaciones complejas, llamadas **curvas de Bézier**. Gracias a esto manejaremos la tinta digital además ocupa muy poco espacio en los archivos lo que facilita su almacenamiento.

4.2.1.1. Gráficos vectoriales: Curvas de Bézier

Los gráficos vectoriales, también conocidos como **gráficos orientados a objetos**, son imágenes digitales. Son más simples que los gráficos de mapas de bits, ya que en ellos las imágenes se almacenan y representan por medio de trazos geométricos controlados por cálculos y fórmulas matemáticas, tomando algunos puntos de la imagen como referencia para construir el resto.

Por lo tanto, las imágenes en los gráficos vectoriales no se construyen píxel a píxel como los mapas de bits (bitmap), sino que se forman a partir de vectores, objetos formados por una serie de puntos y líneas rectas o curvas definidas matemáticamente.

Por ejemplo, una línea se define en un gráfico de mapa de bits mediante las propiedades de cada uno de los píxeles que la forman, mientras que en un gráfico vectorial se hace por la posición de sus *puntos inicial y final y por una función que describe el camino entre ellos*. Por ejemplo un círculo se define vectorialmente por la posición de su punto central (coordenadas x,y) y por su radio (r).



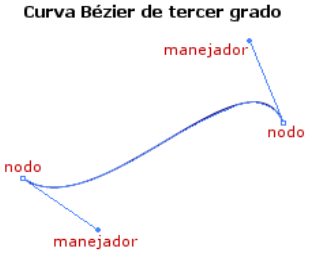
Cada vector en un gráfico vectorial tiene una línea de contorno, con un color y un grosor determinados, y está relleno de un color a elegir. Las características de contorno (o filete) y relleno se pueden cambiar en cualquier momento.

Las *imágenes vectoriales se almacenan como una lista* que describe cada uno de sus vectores componentes, su posición y sus propiedades.

Figura 4.1 Definición vectorial de un círculo

En cuanto a la resolución, los gráficos vectoriales son independientes de la resolución, ya que no dependen de una retícula de píxeles dada. Por lo tanto, tienen la máxima resolución que permite el formato en que se almacena.

Curvas de Bézier



Los principales elementos constituyentes de un vector son las denominadas curvas de Bézier, desarrolladas por Pierre Bézier por encargo de la empresa Renault, que buscaba una familia de curvas representables matemáticamente (son curvas de tercer grado) que permitieran representar las curvaturas suaves que deseaban dar a sus automóviles.

Figura 4.2 Curva de Bézier

Una curva Bézier queda totalmente definida por cuatro puntos característicos, los puntos inicial y final de la curva (**nodos** o puntos de anclaje) y dos puntos de control (puntos de control, **manejadores** o manecillas), invisibles en el gráfico final, que definen su forma. Para modificar su forma, basta cambiar de posición uno de sus nodos o uno de sus puntos de control.

Son curvas de manejo poco complejo y muy elegantes, con un desarrollo muy suave, capaces de adaptarse a casi cualquier forma imaginable, por lo que son muy usadas para diseñar logotipos e iconos y para copiar cualquier figura.

También son enormemente versátiles, pudiendo adoptar desde curvaturas muy suaves (casi líneas rectas) a curvaturas muy fuerte (curvas complejas), pasando por todos los valores intermedios. Pueden, incluso, cambiar de cóncavas a convexas alrededor de un punto.

Un curva de Bézier está determinada por cuatro puntos de control que son:
$P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$, $P_3 = (x_3, y_3)$, $P_4 = (x_4, y_4)$.

y se puede parametrizar por las ecuaciones siguientes:

donde $0 \le t \le 1$. Cuando $t = 0$ se tiene $(x, y) = (x_1, y_1)$ y que cuando $t = 1$, se tiene $(x, y) = (x_4, y_4)$, así que la curva de Bézier empieza en el punto (x_1, y_1) y termina en el punto (x_4, y_4).

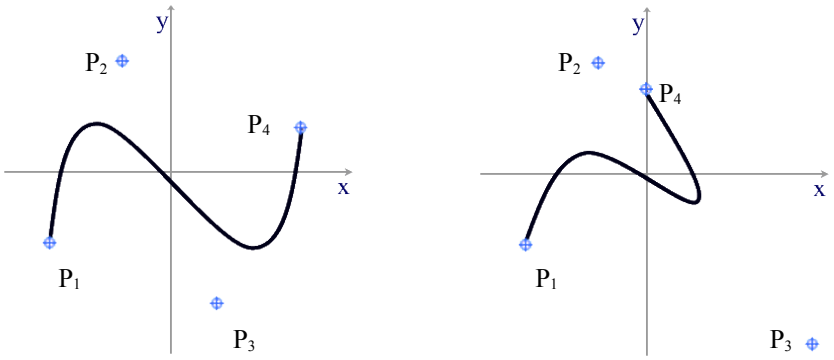


Figura 4.3 Variación de una curva de Bézier

Las **principales ventajas** que ofrecen los gráficos vectoriales, derivadas de su naturaleza matemática, son:

- Almacenan las imágenes en archivos muy compactos, ya que sólo se requiere la información (fórmulas matemáticas) necesaria para generar cada uno de los vectores. dado que no se ha de almacenar información para definir cada punto de la pantalla, sino una serie de fórmulas matemáticas.

- Permiten modificar el tamaño de las imágenes y de sus objetos componentes sin que se produzca pérdida de información, pues se actualizan de forma matemática todas las nuevas relaciones y posiciones de los elementos geométricos que las componen. Con ello, los cambios de tamaño de las imágenes vectoriales no afectan a la calidad de las mismas, apareciendo siempre con la misma nitidez.

- Son muy útiles a la hora de imprimir imágenes, ya que no es necesario pasar a la impresora la información de cada punto. Basta con ir pasándole la información de los vectores que forman la imagen.

- Cada objeto viene definido por sus propias fórmulas matemáticas y se maneja independientemente del resto, pudiendo escalarse, distorsionarse y cambiarse de forma o de posición sin afectar para nada los otros elementos del dibujo.

- Es posible un control independiente del color, tanto del contorno como del relleno, admitiendo la aplicación de texturas, degradados, transparencias, etc.

- Se puede controlar con gran precisión la forma, orientación y ordenación de los elementos.

- Cualquier efecto que se aplique a los objetos puede rectificarse en cualquier momento, ya que el dibujo es siempre editable. Esto no ocurre en las imágenes de mapas de bits, en las que una vez pintado un elemento ya no es posible modificarlo.

- Es fácil reutilizar un dibujo o una parte del mismo en otros proyectos. Basta copiarlo y pegarlo en un nuevo fichero o en uno ya existente.

- Se puede ordenar las formas de cualquier manera si está en superposición unas con otras.

- Permiten un **manejo de textos muy avanzado**, ya que admiten fuentes TrueType, que también son objetos vectoriales. Además, las letras se pueden convertir en contornos editables, descomponiendo un texto en los objetos vectoriales que lo constituyen. Una vez convertidas las letras en objetos, ya no hará falta tener instalada la fuente para seguir editando los contornos, porque ya no serán letras, sino objetos dentro del gráfico vectorial, pudiendo ser modificadas como tales.

Figura 4.3 Texto como objeto vectorial

- Se pueden incluir bitmaps en un dibujo vectorial, bien para rellenos de formas, bien como elementos separados. En el otro sentido, un vector puede exportarse a un formato de mapa de bits estándar, como GIF o JPG.

4.3 Plataforma Tablet PC

Trataremos en esta sección de abordar los principales puntos de interés en cuanto a "tinta" y reconocimiento de escritura, aunque para profundizar más en otros elementos de la plataforma podemos consultar la MSDN Library de Microsoft. Como ya mencionamos anteriormente Microsoft proporciona un kit de desarrollo de software (SDK) así como otros recursos para ayudar a los desarrolladores a que saquen el máximo beneficio a las herramientas de tinta y voz de la plataforma.

Cuando hablamos de "**tinta**" nos referiremos de ahora en adelante a un tipo de datos que representa las pinceladas del lápiz, incluye metadatos sobre las pinceladas y cómo usar tinta en una aplicación.

Microsoft proporciona conceptos e información de referencia para crear aplicaciones para Tablet PC sobre los siguientes temas:

1. **Entrada a través de Lápiz, Tinta y Reconocimiento**
2. **Biblioteca Administrada.**
3. **Biblioteca de Automatización**
4. **Controles de Tinta.**
5. **Referencia API para Tablet PC**

4.3.1 Entrada a través de Lápiz, Tinta y Reconocimiento

Una característica nueva e interesante de la plataforma Tablet PC es el sistema de entrada a través de lápiz. Todos los ordenadores Tablet PC tienen un *digitalizador bajo la pantalla,* es el dispositivo físico que proporciona la interfaz entre el lápiz y el tablet, es un hardware de alta resolución que reconoce los movimientos realizados con el lápiz y se los pasa al tablet, por tanto el tablet acepta entrada de datos a través de lápiz, nosotros lo emularemos mediante la tableta gráfica.

Este mecanismo nuevo de entrada requiere pensar en cómo construir aplicaciones específicamente para Tablet PC. En nuestro caso la utilización de la Tableta Gráfica trata de simular este comportamiento en un PC de sobremesa. La Interfaz de Programación de Aplicaciones (API) de la plataforma Tablet PC nos ayuda a hacer esto y esta disponible en la biblioteca administrada y la biblioteca de automatización.

- **Recolección, Manipulación de Datos, y Reconocimiento**

La plataforma Tablet PC esta dividida en tres áreas bien definidas:

- La Recolección de tinta **Ink Collection and Display** (los objetos que se usan para recoger tinta del digitalizador o tableta gráfica).

- La manipulación de datos relacionados con la tinta **Ink Data** (los objetos que están diseñados para operar con la tinta recogida del digitalizador o tableta gráfica).

- El reconocimiento de tinta **Recognition** (los objetos que están diseñados para convertir la tinta recogida del digitalizador o de la tableta gráfica en otros tipos de datos, como texto Unicode).

La siguiente imagen muestra , cómo trabajan la Recolección de tinta *Ink Collection API* (Pen API), la manipulación de datos relacionados con la tinta *Ink API*, y el reconocimiento de tinta *Ink Recognition API* conjuntamente en la plataforma Tablet PC.

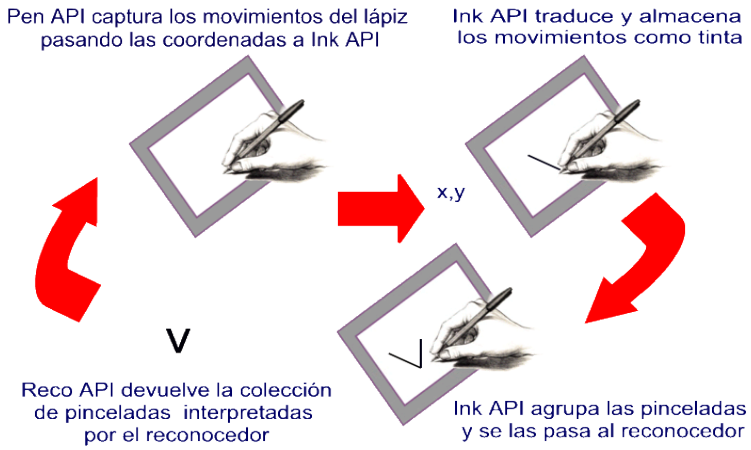


Figura 4.4 API para Tablet PC

El API está disponible en la Biblioteca Administrada y en la Biblioteca de Automatización Los siguientes temas describen los objetos en el API e ilustran cómo usan las aplicaciones estos objetos.

- ***Recolección de Tinta***
- ***Datos de Tinta***
- ***Reconocimiento de tinta***
- ***Análisis de Tinta***

A continuación trataremos de exponer las principales características de las clases más relevantes que forman el API dentro de los cuatro temas mencionados anteriormente, hay que destacar que cada clase tiene implementados métodos, campos, eventos y propiedades de las que no haremos un estudio exhaustivo ya que consideramos que para usar estos elementos es mejor consultar directamente la biblioteca Administrada en http://msdn.microsoft.com.

4.3.1.1 Recolección de Tinta

La recolección de tinta comienza con el digitalizador del dispositivo Tablet PC o la tableta gráfica en nuestro caso. Un usuario coloca el lápiz en el digitalizador o en la tableta gráfica y comienza a escribir. Se pueden usar las características de la recolección de tinta del API para manejar la colección de datos de tinta que proporcionan los movimientos del lápiz. Obtenemos acceso a la información acerca del hardware disponible en el Tablet PC a través de la colección *Tablets* y del objeto *Tablet*. Usaremos el objeto recolección de tinta (*Ink Collector*) para obtener los datos que provienen de la tableta o digitalizador.

- **Tablets y el objeto Tablet**

  Tablet representa el dispositivo digitalizador del Tablet PC, puede haber mas de un dispositivo digitalizador y recibe del dispositivo tablet mensajes o eventos. La clase *Tablets* contiene los objetos Tablet que representan las propiedades específicas del hardware de todos los tablet asociados al sistema. Usando el objeto *Tablet* podemos manipular los dispositivos disponibles y sus capacidades respectivas del hardware. Por ejemplo podemos determinar si el Tablet con el que trabajamos esta integrado o es un dispositivo externo.

  Los objetos Tablet trabajan estrechamente con las colecciones de tinta, la recolección de tinta (*InkCollector, InkOverlay o InkPicture*) puede ponerse en uno de los dos modos relacionados con el tablet:

  - Todos los modos tablet, es el modo predeterminado al que accedemos utilizando el método *SetAllTabletsMode* este método integra todos los dispositivos tablet en el caso de existir múltiples dispositivos asociados al sistema, ya que los cursores disponibles pueden usarse en cualquier dispositivo de la tableta y los mapas de la tableta en la pantalla entera, esto permite la actualización automática de ventanas.
  - Un modo simple integrado.

- **El objeto InkCollector**

  El objeto InkCollector captura tinta de los dispositivos disponibles. Es decir recoge cada movimiento del cursor dentro del área activa de la tableta o del tablet PC. Sólo recolecta tinta y/o formas de tinta que convierte directamente en un objeto Ink .

  El objeto InkCollector se asocia con una ventana de la aplicación, luego le permite al usuario utilizar cualquier dispositivo disponible lápiz o ratón del Tablet PC o de la tableta gráfica para colocar tinta en tiempo real en esa ventana.

  El único propósito de InkCollector es recoger tinta del hardware (por ejemplo a través de los objetos Cursor y Tablet) e introducirla en uno de los diferentes objetos Ink que pueden contenerla dentro de una aplicación.

  Mediante *InkCollector.Tablet* se representa el concepto lógico del dispositivo "digitizador" en el API de la plataforma. Lleva a cabo las características permanentes que describen un digitizador unido al sistema, tal como capacidades y métrica (valores por ejemplo sus mínimos y máximos, unidad de medida, y resolución) del hardware o dispositivo que el objeto de InkCollector está utilizando.

Para usar el objeto InkCollector debemos crearlo, asociarlo a una ventana donde recoger la tinta y habilitar el objeto, después podremos recolectar o recoger tinta en uno de los tres modos siguientes (especificados en la enumeración CollectionMode):

- Sólo tinta **InkOnly**, se recoge solamente tinta creando objetos *Stroke*, de forma que los eventos *InkCollector.Gesture*, *InkOverlay.Gesture* o *InkPicture.Gesture*s son puestos a false, esto significa que no se recogerán formas de tinta.
  Las pinceladas (strokes) de tinta son almacenadas en un objeto Ink. Estas pinceladas pueden ser manipuladas o enviadas a un reconocedor.

- Sólo Formas de Tinta **GestureOnly**, se recolectan solamente formas de tinta y no se crean objetos Stroke. Las formas de tinta pueden estar formadas por pinceladas simples o múltiples. Las pinceladas múltiples son aceptadas si están creadas sin el cronómetro de tiempo del reconocedor.
  Todos los eventos asociados a pinceladas y paquetes en InkCollector, InkOverlay e InkPicture no están activos. Como es lógico los eventos *InkCollector.Gesture*, *InkOverlay.Gesture o InkPicture.Gesture* son puestos a true, esto significa que se recogerán formas de tinta en los tres casos.

- Tinta y Formas de tinta **InkAndGesture**, acepta sólo formas y pinceladas simples. Los eventos InkCollector.Gesture, InkOverlay.Gesture o InkPicture.Gestures son los primeros en ser lanzados permitiendo aceptar, que es la opción por defecto, o cancelar el evento. Si la forma de tinta se acepta, la tinta se borra. Por el contrario si se cancela la tinta no se borra y se lanzan los eventos InkCollector.Stroke, InkOverlay.Stroke o InkPicture.Stroke según el caso.
  En este caso también los eventos InkCollector.Gesture, InkOverlay.Gesture o InkPicture.Gestures son puestos a true.

Esto significa que por cada movimiento del cursor que esta dentro del rango de la tableta, InkCollector siempre recolecta pincelada, forma de tinta o ambas. El soporte de formas de tinta ha sido construido usando el reconocedor de formas de Microsoft.

InkCollector maneja la entrada desde la tableta o el tablet a través de lápiz o ratón, cuando se encuentra un cursor nuevo los eventos *CursorInRange* con la propiedad *NewCursor* y el evento InkCollectorCursorInRangeEventArgs son verdaderos.

Más de un InkCollector puede ser asociado a un manipulador particular de una ventana, tendremos así diferentes áreas podemos hacerlo modificando el constructor o con el método *SetWindowInputRectangle.*

Para que el objeto de InkCollector fije exactamente el cursor del ratón o del lápiz dentro de una ventana con la tinta habilitada, esa ventana debe poder recibir el mensaje de WM_SETCURSOR. Esto es acertado para todas las ventanas regulares pero, para un control dentro de una caja de diálogo, el padre del diálogo filtra este mensaje del control. Para que el control reciba el mensaje, el estilo debe ser SS_NOTIFY.

Los eventos *MouseDown, MouseMove, MouseUp y MouseWhee*l devuelven las coordenadas x e y en pixels, y no en unidades Himetric que son las unidades asociadas al espacio de tinta, esto es por que esos eventos reemplazan los eventos del ratón en aplicaciones que no reconocen el lápiz y operan únicamente con pixels.

- **Objetos Cursor y CursorButton**

El cursor se corresponde con la punta del lápiz que se usa en el Tablet PC o en la tableta gráfica en nuestro caso, no hay que confundirlo con el cursor definido en System.WindowsForms.Cursor, normalmente el lápiz tiene dos usos un lado sirve para borrar y otro para escribir, cada uso se corresponde con un cursor. En un Tablet PC un cursor se define normalmente para ser usado para escribir o borrar. El cursor puede cambiar en la aplicación si se permite esta funcionalidad. Algunos dispositivos Tablet PC admiten lápices múltiples. Un cursor tiene un identificador único en el sistema. Cada cursor tiene cero o mas botones asociados. Estos botones se asocian a la aplicación como objetos CursorButton . La aplicación puede suministrar objetos DrawingAttributes específicos para cada cursor dado.

Al proporcionar un cursor de tinta, debemos considerar que: el cursor debe proporcionar la información sobre el color de la tinta y el grueso de la pluma. El cursor debe ser bastante pequeño para que no interrumpa la vista de la tinta a la vez que el usuario está escribiendo. Las reglas del diseño para los cursores normales también se aplican a los cursores de la tinta.

El cursor debe ser visible para los usuarios diestros y zurdos punto a tener en cuenta cuando se agrega la indicación visual es excéntrica del punto caliente del cursor o hot pixel.

Dentro de la aplicación debe dejarse claro a los usuarios donde pueden manejar tinta y donde no pueden, el cursor asignado a la tinta indica visualmente donde actúa el lápiz como dispositivo de entrada en comparación con otros cursores.

- **Objeto DrawingAttributtes**

Un objeto Drawing Attributtes describe como un conjunto de tinta será dibujado. Incluye propiedades básicas como Color, Anchura y puntas de lápiz. (Color, Width y Pen Tip).También puede incluir parámetros avanzados como transparencia y alisamiento de Bézier, los cuales proporcionan efectos interesantes que pueden mejor la legibilidad de la tinta.

La propiedad **DefaultDrawingAttributes** (InkCollector.DefaultDrawingAttributes, InkOverlay.DefaultDrawingAttributes, o InkPicture.DefaultDrawingAttributes) contiene las cualidades del dibujo por defecto. Estas cualidades por defecto se fijan con la propiedad DefaultDrawingAttributes en los objetos InkCollector.DefaultDrawingAttributes, InkOverlay.DefaultDrawingAttributes, o InkPicture.DefaultDrawingAttributes).

- **El objeto InkEdit**

El control InkEdit permite recoger tinta, reconocerla y mostrarla como texto en caracteres Unicode . Este control permite la entrada de formas naturales.

Este control es un superset del control Rich Edit es decir deriva de la clase base **RichTextBox** utilizado para editar, introducir, guardar, formatear, imprimir y guardar texto, Ink Edit amplia las características en cuanto a que permite además capturar, reconocer y mostrar tinta (Ink).

El control InkEdit puede reconocer texto escrito a mano de dos formas, manualmente llamando al método *InkEdit.Recognize* del que dispone, o automáticamente después de que el tiempo de reconocimiento expira variando el valor de la propiedad *RecoTimeout* cuyo valor por defecto es 2 segundos (2000 milisegundos) en ambos casos el texto reconocido se muestra dentro de los límites del control.

Otra característica interesante de este control es que es posible reconocer "tinta" dentro de un contexto en particular, la propiedad *Factoit* permite indicar distintos campos como PostalCode, Email, Web, teléfonos o nombres de fichero.

InkEdit es el control adecuado para trabajar con el objeto PenInputPanel como veremos al hablar de el ya que proporciona el motor de servicios de texto -Text Services Framework- (TSF).

También podemos elegir el reconocedor que usará InkEdit modificando el valor de la propiedad de lectura escritura *Recognizer*, de no hacerlo utilizará el reconocedor por defecto.

Si trabajamos con Visual Studio, al agregar como referencia el ensamblado Microsoft.Ink - como veremos mas adelante - el objeto InkEdit se agregará como un control, previamente debemos ampliar el Cuadro de Herramientas de Visual Studio, seleccionando **.NET Framework Components**. El control estará listo para usarlo en la pestaña Components, únicamente debemos seleccionar, arrastrar y colocar en la ventana como hacemos con otro tipo de controles.

Mediante este control podemos introducir tinta, aceptar gestos o reconocer escritura a mano entendida como "tinta" que representa caracteres incluyendo lenguaje escrito si previamente hemos instalado el SDK para Tablet PC y además para reconocer escritura a mano el módulo de reconocimiento en caso de no estar usando la edición profesional para Tablet PC del sistema operativo.

Si el sistema operativo no es la edición para Tablet PC, es posible asignar valores al control InkEdit y así poder copiar y pegar tinta en otras aplicaciones, pero el valor de su propiedad *InkMode* esta deshabilitado.

InkMode es un tipo enumerado que indica cuando el control InkEdit recolecta tinta, gestos o ambos. Sus posibles valores son:

- *Disabled*. La recolección de tinta esta deshabilitada y no se pueden crear pinceladas (strokes).
- *Ink*. Sólo es posible crear pinceladas de "tinta" (Ink).
- *InkAndGesture*. Es posible crear pinceladas de "tinta" y gestos (gestures).

El valor de esta propiedad permanece Disabled si estamos utilizando el SDK pero no hemos instalado el módulo de reconocimiento y podremos cambiar su valor si estamos utilizando el sistema operativo para Tablet PC.

La inserción inteligente del espacio está disponible solamente cuando la edición Tablet PC esta instalada.

Del mismo modo ficheros con objetos de tinta incrustados se pueden cargar y mostrar en cualquier edición de Windows.XP (incluyendo Tablet PC) y en los sistemas que pueden tener solamente el SDK de la tableta instalada. Sin embargo, los objetos tinta incrustados se pueden convertir al texto solamente cuando la edición Tablet PC de Windows.XP está instalada.

Este control es el idóneo para crear un "diccionario a medida", las palabras se almacenan en un objeto *WordList* que mencionaremos mas adelante, y podemos introducirlas en una aplicación escribiéndolas a mano.

El objeto PenInputPanel

El objeto PenInputPanel fue introducido para permitir integrar las herramientas de entrada de texto directamente en las aplicaciones. Esta disponible como un objeto incorporable para añadir al panel de entrada del Tablet PC la funcionalidad de controles existentes. La interfaz de usuario dependerá del lenguaje actual de entrada.

Existe la opción de escoger el método predeterminado de entrada para el panel de entrada del lápiz ya sea escritura a mano o teclado. El usuario final puede alternar entre los métodos de entrada usando botones en la interfaz de usuario.

Puesto que es posible utilizar un Tablet PC sin un teclado, la edición para Tablet PC y el kit del desarrollo del software Windows.XP (SDK) incluyen un panel llamado accesorio de la entrada de Tablet PC para la entrada de texto. El panel de entrada es a veces la única manera de que los usuarios introduzcan texto. El panel de la entrada permite a los usuarios introducir texto de entrada usando:

- Reconocimiento de escritura a mano de palabras.

Figura 4.5 Ejemplo de Panel de entrada para reconocimiento de escritura a mano

- El reconocimiento de caracteres y "atajos" (shorcuts) denominados gestos (gestures) para usar comandos especiales como por ejemplo ENTER.
- Reconocimiento de discurso o voz para los idiomas en los que esta implementado.
- Una representación de en pantalla de un teclado estándar con funciones de un teclado hardware.

Figura 4.6 Ejemplo de panel de entrada tipo teclado

El objeto InkOverlay

El objeto InkCollector es útil en aplicaciones que suministran su propio modelo de selección, borrado y otras interacciones con el usuario. El objeto InkOverlay es un superset del objeto InkCollector que suministra soporte de edición. Esto es muy usado en aplicaciones que integran dibujo y edición de tinta en el mismo documento usando para ello un conjunto de modelos estándar de selección de tinta que el objeto suministra.

Tanto InkCollector como InkOverlay así como el objeto InkPicture usan construcciones comunes, como por ejemplo el objeto Ink y la colección de atributos de dibujo DrawingAtributes, de forma que cambiar el color de la tinta se realice igual en los tres objetos. Esto permite rehusar código y tener un modo de acceso común, lo cual es importante si se utiliza como guión de soporte de la aplicación.

InkOverlay es un objeto COM muy usado en en aplicaciones en las que el usuario no esta tan interesado en realizar reconocimiento de tinta como en el tamaño, forma, color y posición de la tinta. Es conveniente para tomar notas sobre documentos. La interfaz de usuario predeterminada es un rectángulo transparente con tinta opaca. El uso primario de este objeto es exhibir la tinta como "tinta".

InkOverlay prolonga la clase InkCollector en tres sentidos:

- Formula eventos de empezar y terminar pinceladas, y cambios en los atributos de la tinta.
- Le permite a los usuarios seleccionar, borrar, y ajustar el tamaño de tinta.
- Soporta los comandos Cut, Copy y Paste (Cortar, Copiar y Pegar).

InkOverlay también se utiliza con frecuencia para poner marcas en diapositivas de presentación e imágenes. Permite la implementación fácil de la tinta y las capacidades de trazado que estos usos requieren.

Para trabajar con InkOverlay deberemos crear una instancia del objeto, asociar el objeto con un manipulador en la ventana mediante Handle, y poner la propiedad Enabled del objeto a True.

Incluye además soporte básico de impresión, pero debemos implementar vistas previas de impresión y otras capacidades de impresión avanzadas.

InkOverlay se guarda en formato serializado de tinta (ISF).

El objeto de InkOverlay es un objeto de COM que se puede unir a un control de la ventana y se utiliza permitir capacidad básica de la tinta. Se puede utilizar el objeto de InkOverlay para borrar la tinta fijando la propiedad EditingMode = Delete y fijar la propiedad EraserMode al movimiento o Stroke o al punto. Fijar la propiedad EraserMode a Stroke borra pinceladas o movimientos enteros. Fijar la propiedad EraserMode a point borra la tinta sobre la cual el cursor pasa.

Si EditingMode = Delete o EditingMode = Select otros eventos como InkAdded, InkDeleted y Stroke están disponibles. Estos eventos se usan mucho si queremos implementar nuestro propio modo de borrado o selección.

Seleccionando Tinta

El objeto InkOverlay facilita al usuario el uso de un lazo de selección que contendrá la región seleccionada, el usuario puede también seleccionar tinta golpeando suavemente con el lápiz sobre el objeto Ink en la pantalla que desee seleccionar.
Podemos usar la propiedad Selection para devolver una colección de pinceladas (Strokes) que se pueden usar para manipular la selección del usuario.
Cuando un objeto Ink o un conjunto de objetos Ink es seleccionado, un marco rodea la selección formado por cuadros de mayor tamaño en las cuatro esquinas y de menor tamaño en el medio unidos por lineas de puntos. Si el usuario arrastra donde quiera la región seleccionada, la tinta se convierte en móvil dentro del control al que ha sido asociada.
Valores por defecto
El objeto InkOverlay es una recolección de tinta por defecto. La Tinta tiene hasta 53 unidades de ancho (1 unidad de espacio de tinta es igual a 1 HIMETRIC). Por defecto la tinta es negra si el usuario no varia este valor, ni se utiliza el modo alto contraste COLOR_WINDOWTEXT, y la propiedad DrawingAttributes.FitToCurve es False.

El objeto InkPicture

El control de InkPicture permite colocar fácilmente una imagen (jpg, bmp, png, o formato del gif) en una aplicación donde los usuarios pueden agregar tinta. Esta pensado para escenarios en los cuales la tinta no necesita ser reconocida como texto, pero se almacena como tinta.

Los usuarios agregan la tinta a una capa transparente usando el lápiz. Es posible redimensionar el tamaño de la ventana de InkPicture sin perder ninguna información de la tinta.

El control de InkPicture incluye soporte de impresión básico; sin embargo, es posible implementar vistas previas y capacidades de impresión avanzadas.

InkPicture hereda de la clase de PictureBox. Por el defecto, la tinta es de color negro si no esta en modo high-contrast; si no, se fija al valor actual de Color.WindowText (COLOR_WINDOWTEXT). También por defecto FitToCurve es falso que el el valor que indica cuando el alisamiento de Bézier se esta utilizando para interpretar la tinta.

Dentro del control de InkPicture, se utiliza el objeto Ink para recuperar y guardar tinta, se guarda como tinta y no como texto.

Del mismo modo que ocurría con el objeto InkEdit Mediante este control podemos introducir tinta, aceptar gestos o reconocer escritura a mano entendida como "tinta" que representa caracteres incluyendo lenguaje escrito si previamente hemos instalado el SDK para Tablet PC y además para reconocer escritura a mano el módulo de reconocimiento en caso de no estar usando la edición profesional para Tablet PC del sistema operativo.

Si el sistema operativo no es la edición para Tablet PC, es posible asignar valores al control InkEdit y así poder copiar y pegar tinta en otras aplicaciones, pero el valor de su propiedad *InkEnabled* esta deshabilitado.

Puesto que InkPicture además de ser un objeto es un control podremos seleccionarlo del cuadro de herramientas de Visual Studio y arrastrarlo a la ventana para usarlo si previamente hemos agregado las referencias correspondientes que se detallan mas adelante.

- **El objeto Renderer**

Representa la manipulación del trazado de la tinta en una ventana de la pantalla. Se utiliza el objeto Renderer para mostrar, interpretar la tinta en una ventana. También puede utilizarse para recolocar y para volver a clasificar según el tamaño movimientos o pinceladas, por ejemplo para convertir coordenadas de espacio de tinta en pixels. Se usa en combinación con InkCollector, InkOverlay e InkPicture como propiedad.

4.3.1.2 Datos de Tinta

El equipo del desarrollo del Tablet PC decidió que la tinta tuviera su propio tipo de datos. La tinta creada por un programa tiene su propio formato gráfico portable llamado formato serializado de tinta, **Ink Serialized (ISF)**. Este formato se puede serializar y guardar en formato binario o XML, y puede ser puesto en el portatapapeles. La tinta puede puede habilitarse para Web si se convierte a un GIF a partir del formato ISF. La clase que maneja la tinta como datos a es la clase **Ink**.

Las clase InkCollector y la clase InkOverlay manejan un objeto Ink, que accede a todos los datos recogidos de la tinta. La clase **Ink** proporciona la ayuda para serializar la tinta y para apoyar el portatapapeles. La clase Ink guarda la tinta de forma persistente mediante el método *Save*. A pesar de su nombre, este método no escribe datos al disco, sino convierte datos de la tinta en un array de bytes. La clase también dispone de un método de recuperación *Load*, que copia el array de bytes que previamente fue creado al grabar, y lo coloca en un objeto Ink vacío.

Los datos de ISF están disponibles en cuatro diversos encodings o codificaciones:

- Como una corriente simple (stream) de ISF.
- Como imagen GIF con una stream de ISF encajado como metadata. Esta codificación pone una imagen de la tinta a disposición de aplicaciones que no permiten tinta visiblemente como por ejemplo en una página del Web, los programas que permiten el uso de tinta pueden también tener acceso a la corriente simple de ISF dentro del GIF.
- Como un stream de ISF almacenada como datos base-64 en XML-compatible.
- Como un GIF almacenado de forma semejante a datos base-64.

Los datos de la tinta se pueden transferir entre los programas vía portapapeles. Compartir los datos ha sido siempre una característica importante de Windows; la creación de un nuevo formato de datos plantea cómo se compartirá.

Cuando la tinta se copia al portapapeles, los formatos posibles se agrupan en dos tipos: programas que tienen tinta habilitada y programas que no la tienen. Un programa con tinta habilitada dispone de diversos tipos de datos en el portapapeles. Cuando un usuario pega objetos de tinta en un programa con tinta habilitada, el recipiente puede solicitar ISF del sujetapapeles. Cuando la operación de la goma se realiza a un programa no-tinta-permitido, una variedad de formatos gráficos alternativos, tales como BITMAP y metafiles, permita que el recipiente exhiba la tinta. El objeto de la tinta apoya la creación de objetos OLE embeddable. Esto es un acercamiento listo porque permite que la tinta sea incorporada en los usos que tinta-no se permiten de otra manera. En efecto, estos objetos embeddable permiten que cualquier programa OLE-compatible reciba objetos tinta-permitidos. Para el presente, hay dos tipos básicos de objetos OLE embeddable de la tinta: tinta del bosquejo y tinta del texto. Se espera que la tinta del bosquejo contenga un dibujo; se espera que la tinta del texto contenga los datos que se pueden convertir al texto vía un reconocedor tinta-compatible

Después de que la tinta sea recogida, las aplicaciones pueden manejar, manipular, y transferir esa información a otro soporte lógico informático. Las acciones de selección, copia, movimiento, guardar, visualizar, y alterar el lugar donde esta el objeto tinta(Ink) y los miembros que contiene se tratarán como una recolección de pinceladas (Strokes) y pincelada (Stroke).

Tinta, pinceladas y paquetes (Ink, Strokes and Packets)

- **Objeto Ink**

  Un objeto Ink es un contenedor de datos del pinceladas o movimientos (puntos). Los datos de pincelada, o los puntos que son recogidos por el lápiz, se colocan en un objeto Ink.

  La propiedad **Strokes** contiene los datos para todos los movimientos o pinceladas dentro del objeto Ink. El objeto InkCollector, el objeto InkOverlay, el control InkPicture, y el control InkEdit recogen puntos del dispositivo de entrada y los ponen en un objeto Ink.

Estos objetos esencialmente actúan como la fuente que distribuye la tinta en diversos objetos Ink, que actúan como contenedores que guardan la tinta distribuida. El espacio de la tinta es un espacio de coordinadas virtual en el cual se trazan las coordenadas del contexto de la tableta. Este espacio está fijado en un sistema de coordenadas denominado HIMETRIC, modo de trazado métrico que asigna a cada unidad lógica 0,01 mm.. En este espacio de coordenadas, un movimiento de 0 a 1 equivale 1 unidad de HIMETRIC. Esto proporciona facilidad para relacionar objetos múltiples de tinta.

El objeto **Renderer** es el encargado de manejar el trazado entre el objeto Ink y cómo se muestra en la ventana de la pantalla.

Un objeto tinta (Ink) es fundamentalmente un tipo de dato que maneja, manipula y almacena la recolección de datos de entrada que proviene del objeto InkCollector. Un objeto tinta (Ink) contiene una o mas pinceladas (objetos Stroke) y métodos y propiedades comunes para manejar esas pinceladas. Una pincelada (Stroke) es definida como el conjunto de datos que se capturó en un solo movimiento del lápiz, que consiste en la secuencia colocar el lápiz en la zona sensible de la tableta en nuestro caso, mover y levantar el lápiz. Los datos de la pincelada contienen una colección de paquetes. Un paquete (packet) es un conjunto de datos que el dispositivo Tablet ya sea Tablet PC o la tableta gráfica envía sobre cada punto simple. Estos datos contienen información como coordenadas, presión del lápiz, ángulo del lápiz con respecto a la zona sensible y cualquier otro dato que el hardware puede transmitir. La propiedad PacketDescription describe los paquetes que genera Tablet como clase .

Objeto Strokes

Podemos obtener las pinceladas o movimientos de un objeto tinta (Ink) accediendo a la propiedad Strokes del objeto, esta propiedad es una colección de referencias para las pinceladas en el objeto tinta (Ink). Si se suprimen o añaden pinceladas del objeto tinta la recolección previamente obtenida no será actualizada.

Para mantener esta propiedad sincronizada con el objeto tinta (Ink), deben ser tratadas mediante los eventos manipuladores StrokesAdded y StrokesRemoved. Estos eventos deberían obtener una copia nueva de Strokes actualizada.

Un objeto Stroke puede contener un solo punto, el método GetPoints del objeto Stroke puede devolver un array con un solo elemento.

La propiedad PacketDescription

Define el conjunto de información que la aplicación obtiene del dispositivo Tablet PC. La información incluye coordenadas, pero puede incluir mucha mas información detallada dependiendo de lo que el digitalizador pueda producir como el ángulo del lápiz o la presión del lápiz, este tipo de información puede ser accesible para el usuario ya que dispone de control información sobre las propiedades del dispositivo Tablet PC que quiere usar.

Objeto ExtendedProperties

Proveen un mecanismo para adjuntar datos definidos en la aplicación a Ink y otros objetos.

No se puede almacenar un objeto vacío de ExtendedProperties. El objeto debe contener datos antes de que pueda ser almacenado. Por ejemplo, si se intenta agregar propiedades extendidas a un objeto Stroke para un uso posterior, se lanza una excepción si la colección de ExtendedProperties no contiene ningún dato. Las colecciones de ExtendedProperties se pueden agregar a Stroke, DrawingAttributes, y Ink.

Traducción de Tinta (Ink Rendering)

El objeto Renderer es el responsable de la traducción de tinta. Proporcionado un contexto apropiado de dibujo, Renderer puede trazar un mapa de coordenadas del espacio de tinta en pixeles, aplicando transformaciones visuales y mostrar tinta en pantalla e impresora. Draw y DrawStroke son los métodos primarios para traducción de tinta.

Cúspides (Cups)

Una pincelada normalmente comienza cuando el lápiz se apoya sobre la zona sensible y acaba cuando se levanta. Dentro de una pincelada los picos, los ángulos y cambios radicales de direcciones son cúspides designadas. Los puntos finales de un golpe son también considerados cúspides, por ejemplo la letra "L" tiene tres cúspides. Cuando se considera una pincelada, normalmente es alisado usando una curva Bézier (o polílinea). Las curvas Bézier pueden convertir cúspides en lazos pequeños.

Las cúspides también pueden usarse para subdividir pinceladas en unidades borrables, borrando la tinta existente entre dos cúspides.

4.3.1.3 Reconocimiento de Tinta

No todas las aplicaciones requieren reconocimiento de tinta, la mayoría de las aplicaciones son diseñadas con texto introducido por teclado como información primaria, la capacidad de convertir tinta escrita a mano en texto es por tanto muy valorada. Podemos usar las características de reconocimiento de la plataforma Tablet PC. consultando la interfaz de programación de aplicaciones (API) para obtener información acerca de los motores de reconocimiento que están disponibles, como qué los idiomas reconocen. Un vez disponemos de reconocedores podemos enviar una recolección de pinceladas (Strokes) de un objeto tinta (Ink) al motor de reconocimiento y nos devolverá un objeto RecognitionResult como resultado.

- **Objeto RecognizerContext**

  Un objeto RecognizerContext es una instancia de un reconocedor instalado, nos permite reconocer una recolección síncrona o asíncrona de pinceladas. Cuando el reconocimiento es asíncrono el objeto RecognizerContext devuelve un objeto RecognitionResult cuando tiene lugar el lanzamiento del evento reconocer en la aplicación.

- **Objeto Recognizers y Recognizer**

  En el Tablet PC puede haber uno o mas reconocedores disponibles, determinar que reconocedor usar de la colección de reconocedores dependerá de la necesidad del usuario, un reconocedor proporciona información especifica sobre sus capacidades y el idioma que puede reconocer.

  Para poder instanciar InkRecognizerContext fuera del entorno de un sistema Tablet PC deberemos instalar un módulo de reconocimiento como es nuestro caso en el que utilizamos una tableta gráfica para simular el comportamiento del Tablet.

- **Objetos RecognitionResult y RecognitionAlternate**

  El resultado del reconocimiento se devuelve en un objeto RecognitionResult. Este resultado es la mejor aproximación como "cadena" en propiedad TopString, obtendremos también una colección de resultados como alternativa al mejor en RecognitionAlternates. El objeto RecognitionResult puede ser puede ser obtenido de la recolección original de pinceladas (Strokes) de la cual fue generado.

  Debido a que hay muchas variaciones entre la escritura a mano de cada individuo, los reconocedores de escritura a mano deben convertir escritura a mano en texto y puede que el resultado sea diferente al que el usuario quería expresar. Cuando un reconocedor realiza el reconocimiento en una recolección de pinceladas o movimientos, el reconocedor encuentra el conjunto más probable de palabras que la escritura a mano representa. Además, el reconocedor también encuentra un conjunto de alternativas del reconocimiento, que se almacenan en una colección de RecognitionAlternates. Para permitir aprovechar este reconocimiento suplente, se debe crear una interfaz de usuario que permita que se seleccione el RecognitionAlternate correcto.

  Se puede almacenar RecognitionAlternates de modo que el usuario pueda seleccionar o reelegir a suplentes en cualquier momento.

- **Objeto WordList**

  Para crear un diccionario a medida , es necesario fijar la propiedad de WordList del objeto RecognizerContext. El control de InkEdit accede al objeto RecognizerContext, así que no es posible fijar directamente la propiedad de WordList del objeto RecognizerContext del control de InkEdit. Para utilizar un diccionario necesitamos un control de InkEdit como ya mencionamos anteriormente, debemos tomar las pinceladas del control InkEdit, y pasarselas a un nuevo objeto de RecognizerContext con la propiedad WordList fijada con una lista de palabras que hemos previamente introducido a mano y que forma el diccionario, estos resultados los proporciona el RecognizerContext, y después pasa los resultados de nuevo al control de InkEdit.

Objeto RecognizerGuide

Representa el área que el reconocedor utiliza es decir donde la tinta puede ser dibujada. El área se conoce como la guía del reconocedor.

La guía del reconocedor puede consistir en filas y columnas, y proporciona al reconocedor un contexto mejor en el cual realizar el reconocimiento. Por ejemplo, usted puede dibujar las lineas horizontales en una pantalla de los usuarios, como se hace en papel, que muestran donde se debe escribir la tinta (este tipo de guía consistiría solamente en filas, y de ninguna columnas). Si un usuario escribe en las líneas, en vez de en un cierto espacio arbitrario, la exactitud del reconocimiento mejora.

Además de líneas de dibujo, es también posible dibujar células en la pantalla en la cual se escriben los caracteres o las palabras. Esto se llama entrada encajonada y es útil con algunas idiomas asiáticos que utilizan reconocimiento de caracteres como por ejemplo el chino. Por defecto, no hay guía, debe alterarse el valor de la propiedad para fijar la guía.

Objeto Gesture

Representa la capacidad de sondear propiedades particulares de un gesto (gestures) devuelto como resultado de un reconocimiento es decir un gesto es un movimiento que se hace o una forma que se dibuja con el lápiz para enviar un comando al Tablet PC. Cuando se domina el uso de los gestos, pueden utilizarse para realizar tareas comunes con mayor rapidez igual que puede utilizar un botón dedicado del teclado para abrir un programa

La plataforma Tablet PC proporciona el reconocedor de “gestos” de Microsoft como soporte de los gestos que pueden usarse en las aplicaciones. Podemos consultar la lista de los gestos apoyados por el reconocedor del gesto de Microsoft, en la enumeración de **ApplicationGesture**, por ejemplo cuadrado (Square), triángulo (Triangle), y flechas en diferentes posiciones. Los gestos definidos por el usuario normalmente utilizan una letra incluida en un círculo.

En el ejemplo mostrado en la figura dibujamos un cuadrado y el reconocedor de gestures responde “Square”

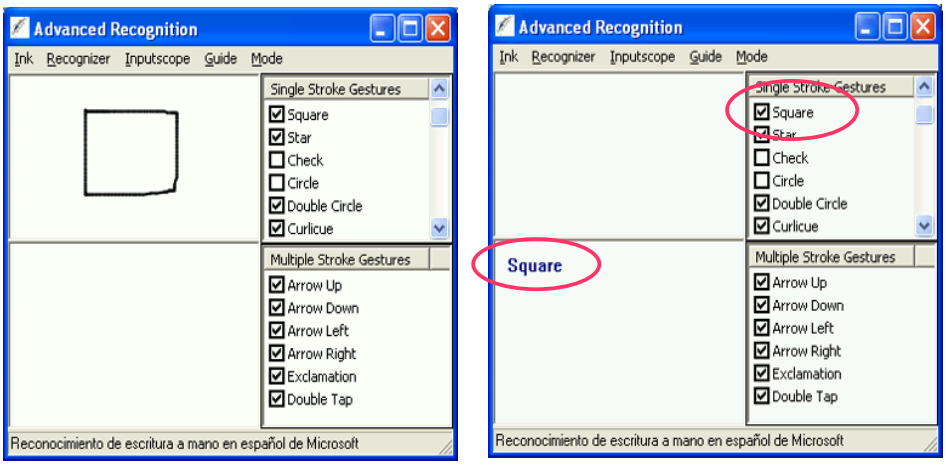


Figura 4.7 Reconocimiento de “gestures”

También proporciona gestos para el sistema, podemos consultar los gestos del sistema disponibles en la enumeración **SystemGesture**. Estos gestos son similares a las teclas de método abreviado y se pueden utilizar para cortar, copiar, pegar y para funciones definidas por el usuario.

Es posible utilizar el reconocedor de gestos utilizando la propiedad e CollectionMode del objeto InkCollector, del objeto InkOverlay, o del control InkPicture. Fijar la característica de CollectionMode a InkAndGesture (modo mezclado) o a GestureOnly (modo del gesto-solamente) pasa la tinta al reconocedor del gesto de Microsoft por defecto. Se puede emplear también el reconocedor con el control InkEdit fijando la característica de InkMode a InkAndGesture.

Por último podemos utilizar el reconocimiento de gestures con el objeto de RealTimeStylus agregando un objeto de GestureRecognizer a una de sus colecciones plug-in. Sin embargo, si los gestos que usted desea utilizar no son apoyados por la versión actual del reconocedor de gestures de Microsoft, podemos construir su propio reconocedor y utilizarlo conjuntamente con el reconocedor de gestures de Microsoft. Esto permite el apoyo total para los gestos en las aplicaciones.

4.3.1.4 Análisis de la Tinta

Cada vez más aplicaciones integran la tinta como forma primaria o alterna de entrada, hay una necesidad creciente de herramientas más de gran alcance que ayuden a manipular esa tinta. En práctica común, la tinta permite escribir notas, escribir entre líneas, utilizar distintos tamaños, dibujos, subrayados, números etc. tal y como lo hacemos en papel es perfectamente natural y absolutamente legible, los comentarios del margen son líneas distintas que no se mezclarán adentro con el cuerpo principal de una nota. Para el ordenador, sin embargo, tinta no es más que una recolección de pinceladas o movimientos en pantalla. Usando el objeto **Divider** se pueden agregar una estructura útil para capturar tinta en los programas. Esta estructura está disponible en dos tipos de clasificaciones:

La **clasificación de escritura a mano** se divide más a fondo en párrafos, líneas, y segmentos. En un reconocedor basado en palabras, un segmento se asocia a una palabra, mientras que en un reconocedor basado en carácter, un segmento se asocia a un carácter. Por ejemplo, el reconocedor de escritura a mano del inglés de Microsoft es un reconocedor basado palabra, y el reconocedor de escritura a mano de chino tradicional de Microsoft es un reconocedor basado carácter.

El objeto Divider analiza los movimientos de la tinta en una recolección dada de movimientos o pinceladas y computa una clasificación para los mismos.

La **clasificación de dibujo** contiene cualquier movimiento que el objeto Divider haya determinado que son dibujos. Todos los movimientos que no se agregan a la recolección de dibujo se dividen en segmentos en colecciones de párrafos, líneas y segmentos.

La clasificación útil porque facilita el reconocimiento, un reconocedor funciona mejor cuando una sola linea horizontal se le pasa para la conversión. Algunos aplicaciones compensan esto proporcionando una guía rectangular en el área de la entrada para indicar donde debe ser escrita la tinta . Sin embargo, las aplicaciones donde la tinta puede escribirse de forma libre no tienen guías de esta clase dentro del área de tinta. En el mejor de los casos, las líneas inmóviles pueden estar presentes, pero esto contradice permitir que los usuarios escriban libremente en la página.

- **Objeto Divider**

  El objeto Divider analiza la tinta y clasifica los movimientos o pinceladas en líneas completas. Las colecciones de movimientos que componen una línea se pueden entonces pasar por separado al reconocedor para la conversión. No hay necesidad de definir los rectángulos horizontales para la entrada.

  Para una aplicación que acepta escritura a mano y dibujo como entrada, la habilidad de distinguir en esta información de entrada segmentos de reconocimiento, líneas, párrafos y dibujo es muy útil. El objeto Divider proporciona características de análisis de trazado que clasifican y agrupan las pinceladas en diferentes elementos estructurales.

  El objeto Divider tiene una recolección de pinceladas (Strokes) y analiza tanto las pinceladas añadidas como las borradas dinámicamente. Podemos recuperar la clasificación actual y agrupar las pinceladas analizadas en un objeto DivisionResult llamando al método Divide del objeto Divider.

Las pinceladas que el objeto Divider analiza son almacenadas en la propiedad Strokes del objeto Divider, ya que la recolección de pinceladas (Strokes) es una referencia de los datos de tinta y no la tinta (Ink) en si misma, los cambios en el objeto Ink pueden invalidar la recolección de Strokes. Usaremos los eventos InkAdded y InkDeleted del objeto Ink para mantener la propiedad Strokes del objeto Divider sincronizada con el objeto Ink y cubrir así los casos en que se añade ,borra o recorta tinta del objeto Ink.

El objeto Divider utiliza un reconocedor de contexto para perfeccionar su análisis de los segmentos de reconocimiento y generar texto reconocido a partir de los elementos escritos a mano. Si no se asigna un reconocedor de contexto en la propiedad RecognizerContext del objeto Divider o no disponemos de reconocedores instalados entonces la característica de análisis de trazado realiza la segmentación pero ningún texto se asocia con el objeto DivisionResult. La propiedad RecognizerContext no puede ser cambiada después de que las pinceladas han sido asignadas al objeto Divider.

Divider no es serializable y no podemos guardar el estado del análisis de trazado pero podemos obtener una visión estática del mismo en el valor devuelto por el objeto DivisionResult.

El objeto Divider solo trabaja con escritura a mano de izquierda a derecha. Para que analice lenguajes verticales como por ejemplo el chino correctamente, los caracteres deben ser redibujados de izquierda a derecha y no de arriba a abajo.

El objeto Divider esta diseñado para separar dibujos y bloques de escritura a mano pero no para reconocer niveles más altos de estructura como tablas o columnas.

El objeto del divisor mejora el reconocimiento para las líneas que se escriben en ángulo. No obliga a los usuarios a un rectángulo particular para cada línea, las líneas levemente angulosas, verticales, o aún al revés se pueden escribir y reconocer con el mismo grado de exactitud que lineas horizontales. Es común para que los usuarios escriban comentarios a los ángulos en los márgenes al tomar notas en un documento. Al convertir estos comentarios a texto, se utiliza la propiedad Transform asociada a cada línea o segmento. Esta propiedad permite rotar los movimientos a un ángulo 0º, que se puede entonces pasar al reconocedor para mejorar resultados.

Además, una aplicación puede también clasificar los movimientos o pinceladas como dibujos, para evitar que los caracteres adicionales sean incluidos en el resultado del reconocimiento.

El objeto Divider esta pensado para ayudar a normalizar la tinta como paso del proceso previo antes de reconocer el texto Si se asigna un reconocedor de contexto al objeto Divider, internamente llama al reconocedor para determinar los segmentos de la palabra o del carácter basados en el contexto de la lengua, pero solamente el resultado superior del reconocimiento para cada palabra o carácter se devuelve. Las aplicaciones que requieren resultados completos del reconocimiento necesitan pasar las líneas al reconocedor directamente usando la propiedad RecognizerContext.

Objeto DivisionResult

Cada objeto DivisionResult registra el análisis del trazado de las pinceladas contenidas en el objeto Divider cuando se llama al método Divide . DivisionResult también almacena una copia de las pinceladas que fueron usadas en el análisis y puede mantener su información después de que el objeto Divider sea destruido.

El objeto DivisionResult agrupa los resultados del análisis por tipos de elementos estructurales. El método ResultByType del objeto DivisionResult devuelve en una colección DivisionUnits la colección de todos los elementos estructurales del mismo tipo. La enumeración InkDivisionType define los tipos del elemento que el análisis de trazado reconoce. Estos tipos son:

Segmento de reconocimiento

Línea de escritura a mano que contiene uno o más segmentos de reconocimiento

Párrafo como bloque de pinceladas que contiene una o mas lineas de escritura

Dibujo Tinta que no es texto.

La siguiente figura muestra un ejemplo con los diferentes tipos que el objeto DivisionResult reconoce:

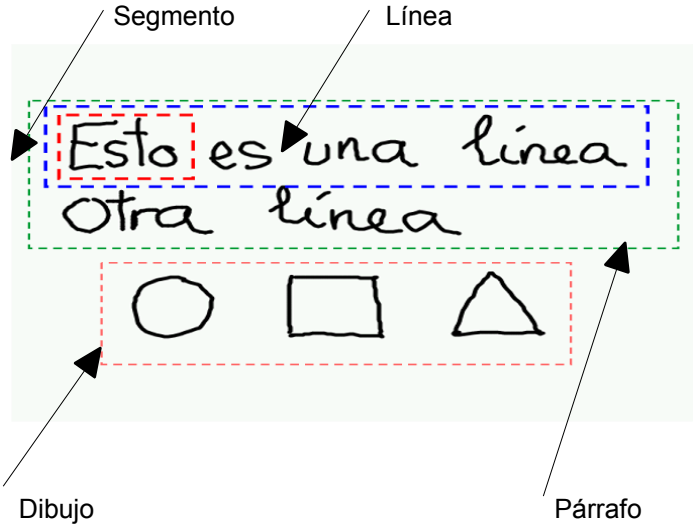


Figura 4.8 Bloques de reconocimiento

- **Colección DivisionUnits y el objeto DivisionUnit**

  Cada colección DivisionUnits es una copia del resultado del análisis de trazado para un solo tipo de elemento estructural. El objeto DivisionUnit representa un elemento individual en la colección DivisionUnits. Cada elemento estructural tiene una referencia a las pinceladas que originan el elemento. Si los reconocedores están instalados, los elementos escritos a mano tienen reconocimiento de texto disponible. Los elementos línea y segmento de reconocimiento también incluyen una matriz de rotación que puede cambiar los elementos de las pinceladas de vertical a horizontal mediante la propiedad Transform, para párrafos y elementos de dibujo la Transform devuelve una matriz identidad. El reconocimiento de texto se realiza mucho mejor si la escritura a mano esta alineada horizontalmente que si no lo esta.

  La propiedad RecognitionString del objeto DivisionUnit devuelve el texto reconocido para los elementos escritos a mano o null para los elementos de dibujo.

  Si en una aplicación se necesita analizar grandes cantidades de tinta es posible almacenar una copia del objeto Ink y del objeto DivisionResult para poder ser revisados. Esto elimina la necesidad de reanalizar el objeto Ink si el usuario regresa al elemento y reduce la cantidad de memoria utilizada.

4.3.2. Biblioteca Administrada Tablet PC

El Common Language Runtime (CLR) maneja código en tiempo de ejecución, suministra servicios como administración de memoria, administración de hilos y acceso remoto al implementar también seguridad estricta de código de hecho el concepto de código administrado es un principio fundamental del CLR. El código que sigue el modelo del CLR se denomina **código administrado** y el código que no sigue el modelo del CLR se denomina **código no administrado**.

La biblioteca de clases Framework es una colección comprimida, orientada a objetos de clases rehusables que podemos utilizar par desarrollar aplicaciones con la forma tradicional de linea de comandos o con interfaz de usuario gráfico (GUI) basándose en las últimas innovaciones proporcionadas por los servicios ASP.NET y Web.

La Biblioteca Administrada Tablet PC contiene un conjunto de objetos administrados que amplían el Framework para proporcionar soporte de entrada y salida para escribir a mano en un Tablet PC en nuestro caso a través de la Tableta Gráfica, así como facilitar el intercambio de datos con otros ordenadores.

Modelo de Objetos de la Biblioteca Administrada de Microsoft Tablet PC

Los objetos mas relevantes que forman parte de la biblioteca han sido mencionados en el punto anterior nos limitamos ahora a mostrar un diagrama que muestra las relaciones entre los miembros del API.

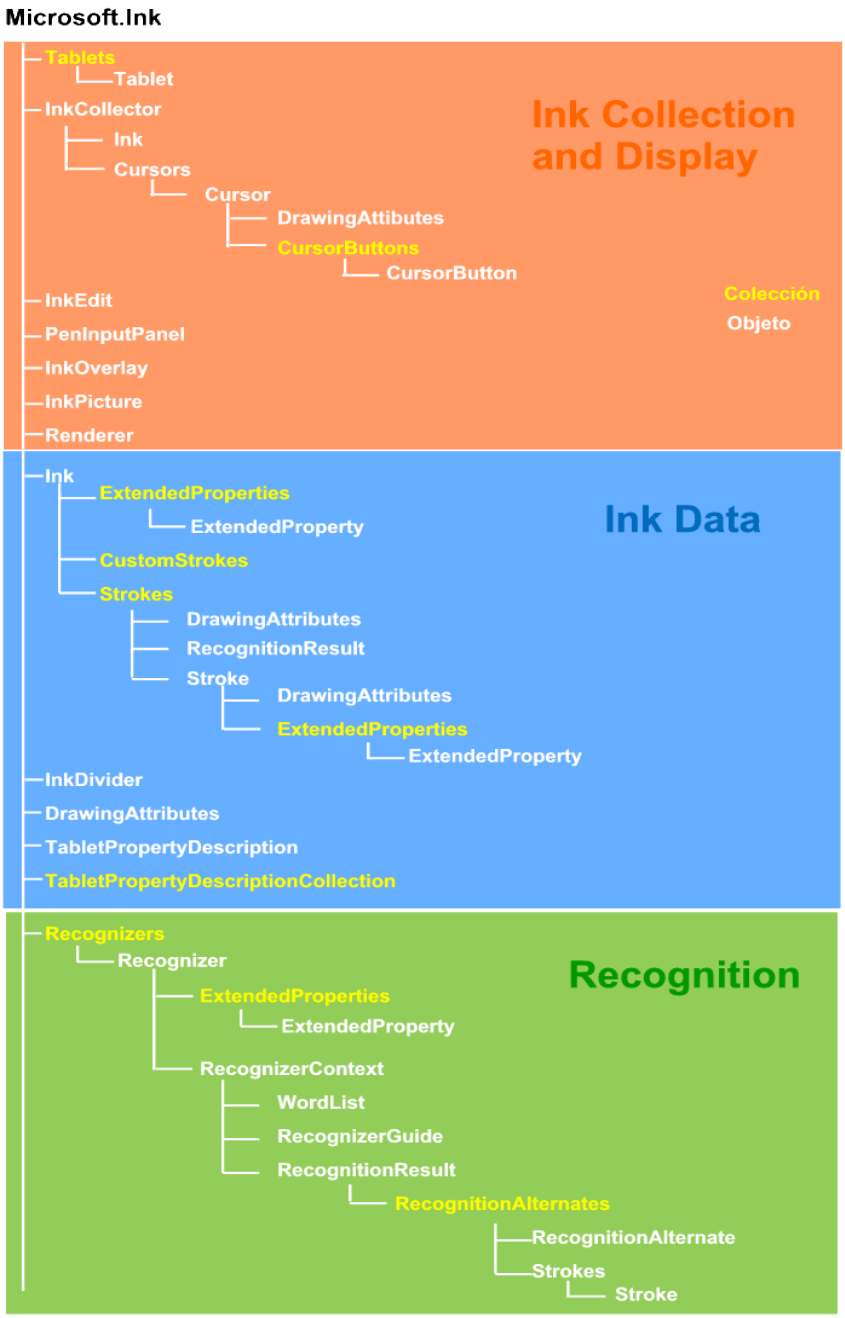


Figura 4.9 Diagrama de la Biblioteca Administrada

4.3.2.1 Microsoft.Ink

La unidad básica de desarrollo en Windows es el assembly o ensamblado. Contiene no solamente el código ejecutable sino también pueden contener archivos de soporte, tales como BITMAP y otros recursos que se puedan requerir para la aplicación. Los ensamblados dinámicos pueden ser creados en tiempo de ejecución usando **System.Reflection.Emit**, permitiendo que el código sea compilado e integrado en tiempo de ejecución. Aunque los ensamblados del NET se diferencian de componentes de COM en varios puntos importantes, existe interoperabilidad entre los dos tipos.

Los ensamblados privados de una aplicación no se colocan en el registro de Windows se almacenan en una carpeta local, privada. Los ensamblados compartidos están disponibles para múltiples aplicaciones, su nombre los identifica y se almacenan en el GAC. Más de una versión de un ensamblado se puede almacenar en el GAC, y se comprueban la versión utilizada en el tiempo de ejecución.

Microsoft.Ink.dll es un ensamblado compartido, esto quiere decir que se almacena en un **caché de ensamblado global** (GAC), para poder usarlo en una aplicación debe haber sido compilado reverenciándolo. Es una DLL (dynamic-link library) una librería de enlace dinámico, un módulo que contiene funciones y datos que pueden ser usados por otro módulo ya sea aplicación o DLL.

Una DLL proporciona una forma de crear aplicaciones modulares de forma que sus funcionalidades podrán actualizarse más fácilmente.

Contiene el espacio de nombres *namespace* dentro del cual se engloban todos los miembros del Tablet PC API anteriormente mencionado, no podemos disponer de este espacio de nombres al arrancar Visual Studio inicialmente, debemos añadirlo como referencia .NET, una vez hemos agregado **Microsoft Tablet PC API** y **Microsoft.Ink.resources** podremos disponer de los miembros de la biblioteca administrada para usarlos en nuestra aplicación.

Los controles InkEdit y InkPicture estarán entonces disponibles en el cuadro de herramientas de Visual Studio.

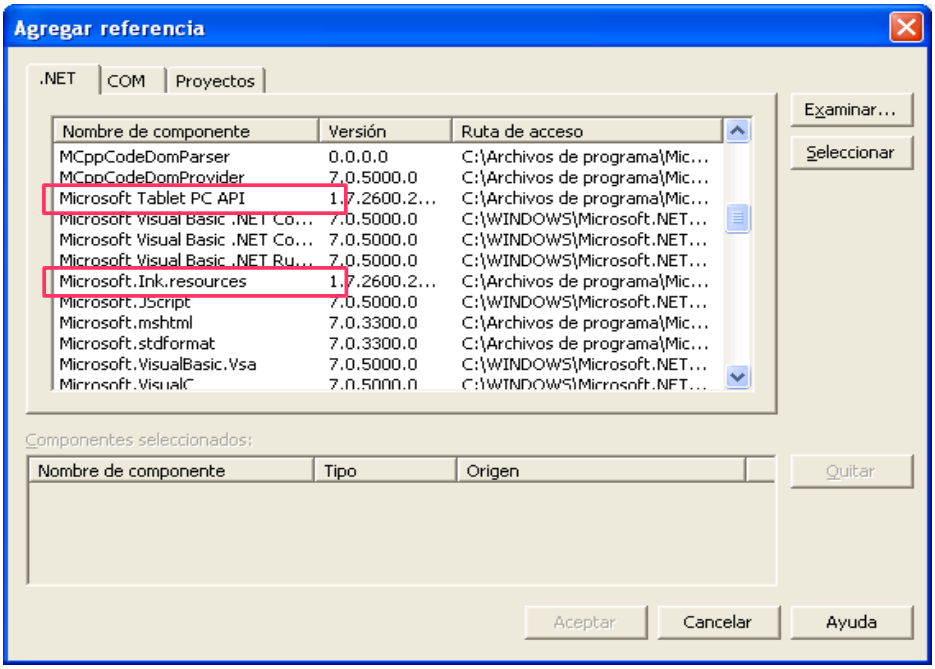


Figura 4.10 Agregar referencias a Microsoft.Ink

Las ultimas versiones de Microsoft.In..dll son compatibles con la versión 1.0 y con la versión 1.5 en la mayoría de los casos no se necesita realizar modificaciones en aplicaciones compiladas con ensamblados antiguos aunque si será necesario darle instrucciones al cargador del CLR para usar las nuevas DLLs si antes se compiló con anteriores versiones, redireccionando las versiones del ensamblado a nivel de aplicación incluyendo esta información en el archivo de configuración incluido dentro del directorio donde se encuentre el ejecutable. Las aplicaciones creadas utilizando el Kit de desarrollo para Tablet PC 1.7 utilizan la última versión del ensamblado, si se utiliza la clase Divider o PenInputPanel deben continuar utilizándose versiones anteriores o recompilar la aplicación con la nueva versión.

4.3.2.2 Microsoft.StylusInput y Microsoft.StylusInput.PluginData

Tablet PC incluye la tecnología para obrar recíprocamente con datos del lápiz la tableta mientras la tinta está siendo recogida. La clase RealTimeStylus es parte de los interfaces de programación de uso de StylusInput (APIs), que proporcionan el acceso a la secuencia de datos de la pluma de la tableta. Este APIs permite capturar, interrumpir, y modificar el stream de datos de tinta independientemente de interpretar y de recoger la tinta.

El StylusInput API se diseñó para:

- Proporcionar acceso en tiempo real a la secuencia de datos de la pluma de la tableta.
- Guardar los hilos (thread) del interfaz de usuario (UI) para bloquear la tinta dinámica que interpreta colocando en una cola los paquetes de datos sobre el hilo (thread) de UI y haciendo un solo hilo de la recolección de la tinta.
- Incrementar la funcionalidad del objeto de InkCollector, el objeto de InkOverlay, el control de InkPicture, o el control de InkEdit para recoger la tinta pero no se diseña específicamente para trabajar con dichos objetos sino para aumentar la realización de los mismos en cuanto a entrada en tiempo real e interpretación.

Cuando se necesita interactuar directamente con la secuencia de datos de la pluma o lápiz de la tableta o cuando su uso puede bloquear la entrada de tinta en tiempo real, debe utilizarse el objeto RealTimeStylus. Se añade a la aplicación el objeto RealTimeStylus y los plug-ins del mismo que pueden modificar los datos asociados a los paquetes y los métodos de notificación.

Los plug-ins pueden cancelar los métodos de los paquetes que están siendo recogidos y notificación de los métodos. Los plug-ins pueden también agregar datos de la aplicación al stream mediante objetos de CustomStylusData. La lista siguiente ofrece ideas para las categorías comunes en las que los plug-ins se pueden utilizar o crear:

- Plug-in filtro: Un objeto que quite o cancele selectivamente datos en la secuencia de datos de la pluma de la tableta.
- Plug-in modificador: Un objeto que modifica selectivamente datos en la secuencia de datos de la pluma de la tableta.
- Plug-in InterpretaciónDinámico: Un objeto que exhibe los datos de la pluma de la tableta en tiempo real mientras está siendo manejado por el objeto de RealTimeStylus. Después a través de eventos el plug-in DinamicRenderer o un plug-in recolección de la tinta pueden redibujar la tinta.
- Plug-in GestureRecognizer: Un objeto que explora el movimiento de la pluma de la tableta para los gestos (gestures) y la escritura a mano.
- Plug-in Ink-Collector: Un objeto que de la secuencia de datos de la pluma de la tableta cree y almacene la tinta.
- Plug-in de la envoltura: Actúa como interfaz entre el objeto de RealTimeStylus y otro plug-in u objeto como manera de modificar el comportamiento del objeto envuelto.

Los plug-ins DynamicRenderer y InkCollection se pueden crear para interpretar varios contextos como un archivo, un stream, o a un dispositivo que muestra tinta en pantalla. La tinta se puede también almacenar en varios formatos, tales como un objeto Ink, un archivo de formato de intercambio de gráficos (GIF), un archivo serializado tinta del formato (ISF), u otros formatos.

Dos plug-ins se proporcionan el StylusInput APIs: la clase de DynamicRenderer y la clase de GestureRecognizer. La clase de DynamicRenderer proporciona la representación básica de los datos de la tinta en tiempo real y se dinamiza para tener un impacto mínimo. La clase de GestureRecognizer proporciona el reconocimiento de gestos (gestures) para la clase de RealTimeStylus.

Por último decir que Microsoft.StylusInput y Microsoft.StylusInput.PluginData no forman parte de la biblioteca de automatización.

4.3.3. Biblioteca de Automatización

Este interfaz de programación de aplicaciones (API) proporciona un modelo de objetos para Microsoft Basic visual 6.0 y Microsoft C++ visual. La mayoría de los objetos de la biblioteca de automatización es idéntico a los encontrados en el API de la Biblioteca Administrada . Sin embargo, la automatización API contiene a algunos miembros más debido a diferencias entre el ambiente estándar de Microsoft Win32 y el kit del desarrollo del software del .NET Framework de Microsoft (SDK). Por ejemplo, los objetos de InkRectangle y de InkTransform se utilizan en la automatización, pero el marco SDK proporciona la puesta en práctica nativa para InkRectangle e InkTransform que elimine la necesidad de estos objetos en la API Administrada. Los objetos en la API de automatización y los controles de la tinta no se diseñan para el uso en las páginas activas del servidor (ASP).

4.3.4. Controles de tinta

La plataforma Tablet PC proporciona dos controles, InkEdit e InkPicture, que permiten agregar fácilmente reconocimiento de escritura a mano y tinta en las aplicaciones de usuario.

InkEdit pertenece a la biblioteca administrada como una clase más y soporta además los interfaces ActiveX y Win32, mientras que InkPicture solamente pertenece a la biblioteca administrada y soporta ActiveX.

La diferencia entre los controles está en cómo se guardan los datos. El control de InkEdit guarda la tinta como texto por defecto, mientras que InkPicture guarda tinta como Ink. El control de InkEdit esta orientado para la entrada de texto a través de reconocimiento de escritura a mano.

InkPicture esta orientado para la anotación (por ejemplo, marcando encima de una diapositiva de una presentación). En código manejado o administrado, los controles de la tinta se crean en el mismo hilo (thread) principal que el form o ventana que los contiene. Si un control InkEdit o InkPicture se crean en un otro thread o hilo diferente del principal pueden funcionar de forma incorrecta.

Como hemos destacado al hablar de los objetos InkEdit e InkPicture se agregan como controles en el cuadro de herramientas de Visual Studio al añadir las referencias adecuadas a Microsoft.Ink momento en el que están disponibles para agregarlos a un form como si de otro control se tratara.

4.3.5. Referencia API para Tablet PC

La plataforma proporciona información de referencia sobre las APIs y controles de tinta siguientes:

- Biblioteca administrada y Biblioteca de Automatización ya mencionadas.
- SetInputScope API sólo disponible en la Edición Tablet PC 2005, define la funcionalidad de alcance de la entrada proporcionada por Windows Text Services Framework.
- Recognizer APIs, API de reconocedores, describe la funciones que manipula varios reconocedores entre las que cabe destacar, qué reconocedores están instalados en el ordenador, identificar l reconocedor que debe usarse, creación de un reconocedor de contexto y recuperación de resultados del reconocedor y resultados alternativos.

Parte III

Desarrollo de la Aplicación

Capítulo 5

ANÁLISIS DEL SISTEMA. DEFINICIÓN DEL PROBLEMA

El análisis se dedica a la comprensión y modelado de la aplicación y del dominio en el cual funciona. La entrada inicial de la fase de análisis es una descripción del problema que hay que resolver y proporciona una visión general conceptual del sistema propuesto. Otras entradas adiciones del análisis son un diálogo con el cliente y un conocimiento a fondo del mundo real. La salida del análisis es un modelo formal que captura los tres aspectos esenciales del sistema: los objetos y sus relaciones, el flujo dinámico de control y la transformación funcional de datos que están sometidos a restricciones.

5.1. Consideraciones iniciales

El objetivo de este proyecto es la realización de una aplicación que sirva como ayuda para el aprendizaje de escritura a mano a personas discapacitadas de una manera asequible y divertida.

Pretende ser una herramienta fácil y práctica que podrán usar todas aquellas personas que puedan iniciarse en la capacidad de escribir y tengan dificultades para ello, con la ayuda de un monitor que supervise el manejo y aprendizaje al menos inicialmente.

5.2. Metodología empleada

El método a seguir deberá ser apropiado para la orientación a objetos, se partirá de la realidad y se irá construyendo una serie de modelos cada vez más detallados hasta llegar a un modelo implementable o solución. Para ello se usa el **Lenguaje de modelado unificado (UML)**.

Se intentará que el método de trabajo sea continuo e **incremental**, procurando eliminar fronteras entre las diversas fases del desarrollo, realizando modificaciones progresivas si es preciso hasta llegar a la implementación final.

El método además será un evolución natural en la que determinadas ideas pueden ser reconsideradas lo que implica modificar o eliminar elementos a medida que se avanza en el desarrollo, siguiendo un **proceso iterativo**.

Se buscará realizar un esfuerzo en la calidad de los componentes software de la solución final (clases).

De igual forma se intentará que las clases tengan independencia, apostando así por la reutilización y facilidad de mantenimiento.

Este proceso así descrito corresponde con **"El proceso Unificado"**, metodología de desarrollo software dirigida por **casos de uso**. Se centra en la funcionalidad del sistema orientada a satisfacer las necesidades del usuario final que interactúa con la aplicación.

Para la implementación se usará el **lenguaje C#**, utilizando Visual C# .NET apropiado para programación orientada a objetos.

5.3. Resultados de la entrevista

La captura de requisitos no es fácil en este caso debido a que las personas a las que va dirigido el proyecto no disponen de facilidad para comunicarse, por esto nos hemos basado en otras experiencias como son las de personas cercanas, como profesores y cuidadores del Centro Obregón, y las directrices sobre las necesidades que había que tener en cuenta que nos han requerido los tutores del proyecto.

También hay que destacar que la aplicación no va dirigida a una persona en concreto ni a una discapacidad determinada, pretende ser una aplicación genérica por tanto las valoraciones en cuento a motricidad y capacidad de comunicación a pesar de haber sido tenidas en cuenta para casos concretos son importantes, ya que las características y necesidades de cada persona con discapacidad son distintas en cada caso.

Debido a que las habilidades de movilidad pueden estar mermadas en muchos casos hemos procurado que el acceso a los controles de la herramienta se realice mediante botones lo suficientemente grandes y visuales aunque claro esta que el manejo de la misma deberá ser supervisado por un tutor o monitor al menos inicialmente, seleccionar los botones es fácil usando el lápiz de la tableta gráfica sólo hay que colocar el lápiz sobre ellos, no lo es tanto escribir en las zonas habilitadas dentro de la aplicación lo cual requerirá mayor destreza y entrenamiento.

5.4. Definición de actores

Los actores son personas o entidades externas a la aplicación que interactúan con ella, realizando un intercambio de información. Éste podrá ser tanto de entrada como de salida.

En sistemas más complejos, un actor puede representar varios papeles. En estos casos, se definen distintos actores para representarlos. Sin embargo, este no es el caso de nuestro sistema, ya que sólo hay un posible actor, que se corresponde con el usuario de la aplicación.

ACT-0001	***Alumno***
Descripción	Este actor representa *al usuario de la aplicación*

Figura 5.1: Descripción actor de los casos de uso.

5.5. Diagramas de casos de uso

Se emplean para visualizar el comportamiento del sistema, una parte de él o de una sola clase. De forma que se pueda conocer como responde esa parte del sistema. El diagrama de casos de uso es muy útil para definir como debería ser el comportamiento de una parte del sistema, ya que solo especifica como deben comportarse y no como deben estar implementadas las partes que define. Un caso de uso especifica un requerimiento funcional, es decir esta parte debe hacer esto cuando pase esto.

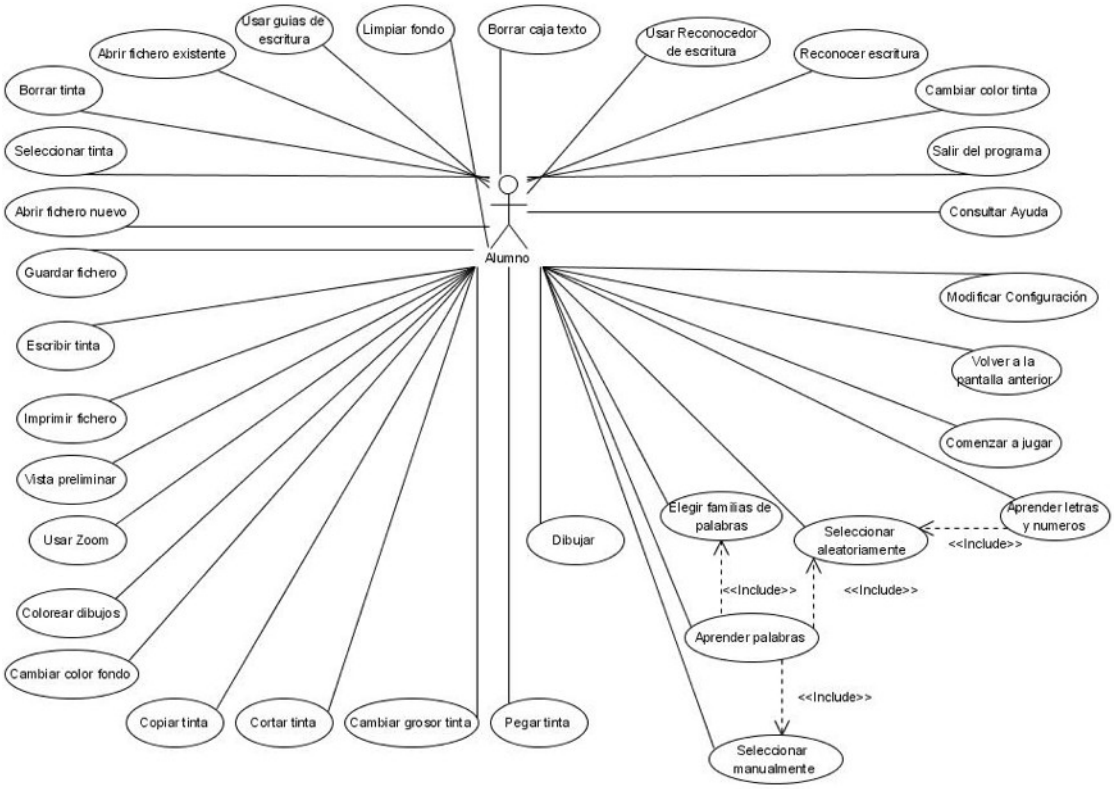


Figura 5.2: Diagrama de los casos de uso.

5.6. Descripción de los casos de uso

El comportamiento de un caso de uso se puede especificar describiendo un flujo de eventos de forma textual, lo suficientemente claro para que un persona ajena al sistema lo entienda fácilmente. A la hora de escribir este flujo de eventos, se debe incluir cómo y cuando empieza y termina el caso de uso en cuestión, cuando interactúa con los actores y que objetos intercambian, el flujo básico y los alternativos de comportamiento.

A medida que se obtiene una mejora en la compresión de los requisitos del sistema, estos flujos de eventos se especifican gráficamente mediante los diagramas de interacción. Normalmente se utiliza un diagrama de secuencia para especificar el flujo principal de un caso de uso.

5.6.1. Consultar Ayuda

UC-0001	***Consultar ayuda***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario quiera acceder a información sobre el uso de ciertas funciones del programa*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *solicita ayuda al sistema*
	2	El sistema *le ofrece información sobre la pantalla desde la que se ha pedido la ayuda*

Figura 5.3: Descripción caso de uso Consultar ayuda.

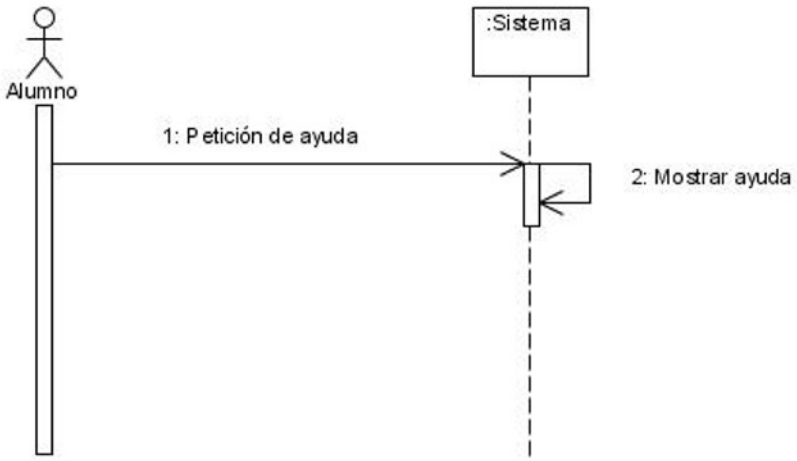


Figura 5.4: Diagrama del caso de uso Consultar ayuda.

5.6.2. Modificar Configuración

UC-0002	***Modificar Configuración***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario quiera modificar alguna de las opciones de configuración*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *solicita la opción de configuración*
	2	El sistema *le muestra un número determinado de opciones que puede modificar*
	3	El actor Alumno (ACT-0001) *elige los cambios que desea realizar*
	4	El sistema *actualiza los cambios*

Figura 5.5: Descripción caso de uso Modificar configuración.

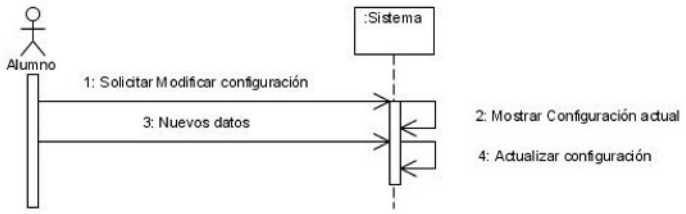


Figura 5.6: Diagrama del caso de uso Modificar configuración.

5.6.3. Usar Reconocedor de Escritura

UC-0003	***Usar Reconocedor de Escritura***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario quiera utilizar el Reconocedor de escritura*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción del reconocedor de escritura*
	2	El sistema *muestra el reconocedor de escritura*

Figura 5.7: Descripción caso de uso Usar reconocedor de escritura.

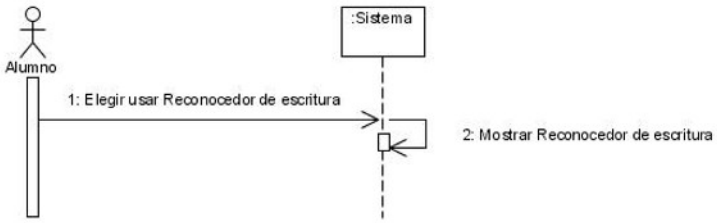


Figura 5.8: Diagrama del caso de uso Usar reconocedor de escritura.

5.6.4. Dibujar

UC-0004	***Dibujar***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario quiera utilizar la opción de Dibujar*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elije la opción de dibujar*
	2	El sistema *muestra la pantalla donde se puede dibujar*

Figura 5.9: Descripción caso de uso Dibujar.

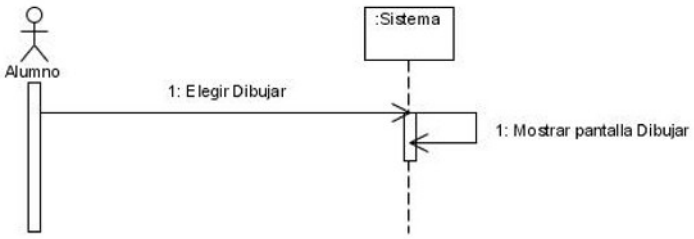


Figura 5.10: Diagrama del caso de uso Dibujar

5.6.5. Aprender letras y números

UC-0005	***Aprender letras y números***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario quiera aprender a escribir las letras del abecedario y algunos números*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de aprender las letras del abecedario y algunos números*
	2	El sistema *muestra el teclado con el que se pueden aprender las letras y los números*
	3	Si *el usuario elige la opción Dados,* se realiza el caso de uso Seleccionar automáticamente (UC-0008)
	4	Si *el usuario elige directamente la letra o el número que quiere aprender,* se realiza el caso de uso Seleccionar manualmente (UC-0009)
	5	El actor Alumno (ACT-0001) *escribe la selección en la caja de texto*
	6	Si *el usuario quiere saber si ha escrito bien la selección,* se realiza el caso de uso Reconocer escritura (UC-0017)

Figura 5.11: Descripción caso de uso Aprender letras y números.

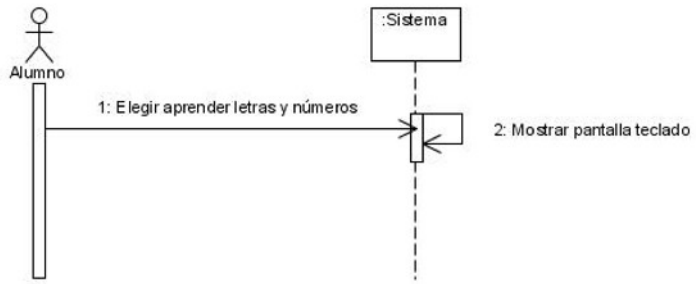


Figura 5.12: Diagrama del caso de uso Aprender letras y números

5.6.6. Elegir familias de palabras

UC-0006	***Elegir familias de palabras***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario quiere elegir entre las distintas familias de palabras que dispone el sistema para aprenderlas* o durante la realización de los siguientes casos de uso: [UC-0007] Aprender palabras	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la familia de palabras que quiere aprender*
	2	El sistema *muestra la pantalla donde se van a aprender palabras*

Figura 5.13: Descripción caso de uso Elegir familias de palabras.

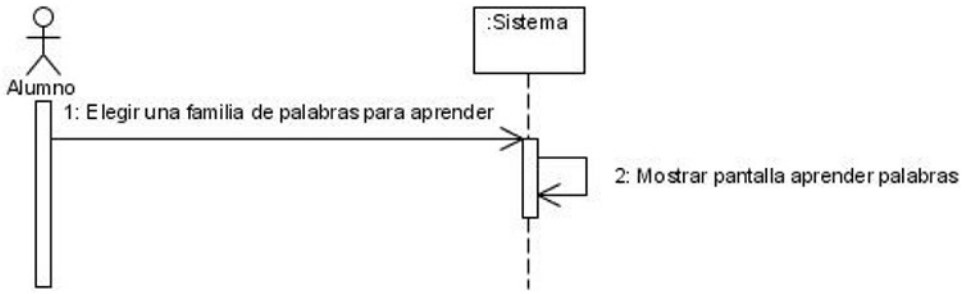


Figura 5.14: Diagrama del caso de uso Elegir familias de palabras

5.6.7. Aprender palabras

UC-0007	***Aprender palabras***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario quiere aprender a escribir algunas palabras*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de aprender palabras*
	2	Se realiza el caso de uso Elegir familias de palabras (UC-0006)
	3	El sistema *muestra la pantalla en la que se podrán aprender las palabras que pertenecen a la familia de palabras escogida anteriormente*
	4	Si *el usuario elige la opción Dados,* se realiza el caso de uso Seleccionar automáticamente (UC-0008)
	5	Si *el usuario elige el mismo la palabra que quiere aprender,* se realiza el caso de uso Seleccionar manualmente (UC-0009)
	6	El actor Alumno (ACT-0001) *escribe la selección en la caja de texto*
	7	Si *el usuario quiere saber si ha escrito bien la selección,* se realiza el caso de uso Reconocer escritura (UC-0017)

Figura 5.15: Descripción caso de uso Aprender palabras.

Figura 5.16: Diagrama del caso de uso Aprender palabras

5.6.8. Seleccionar automáticamente

UC-0008	***Seleccionar automáticamente***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario decida que quiere que el sistema le escoja de forma aleatoria una palabra o una letra o un número para aprender su escritura* o durante la realización de los siguientes casos de uso: [UC-0005] Aprender letras y números, [UC-0007] Aprender palabras	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción dados*
	2	El sistema *elige aleatoriamente la palabra o la letra o el número que tenemos que escribir*

Figura 5.17: Descripción caso de uso Seleccionar automáticamente.

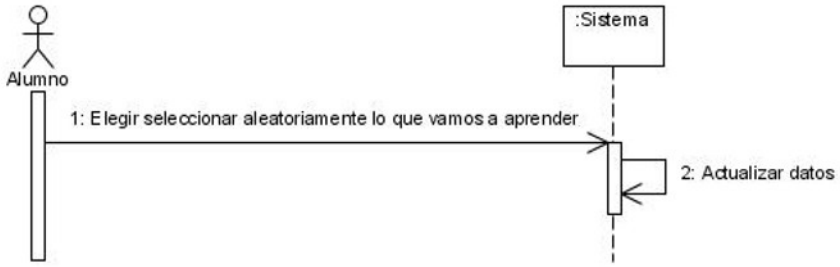


Figura 5.18: Diagrama del caso de uso Seleccionar automáticamente.

5.6.9. Seleccionar manualmente

UC-0009	***Seleccionar manualmente***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario quiera elegir por el mismo la palabra o la letra o el número que quiere aprender a escribir* o durante la realización de los siguientes casos de uso: [UC-0005] Aprender letras y números, [UC-0007] Aprender palabras	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *escoge la palabra o la letra o el número que quiere aprender a escribir*
	2	El sistema *se actualiza marcando la selección del usuario*

Figura 5.19: Descripción caso de uso Seleccionar manualmente.

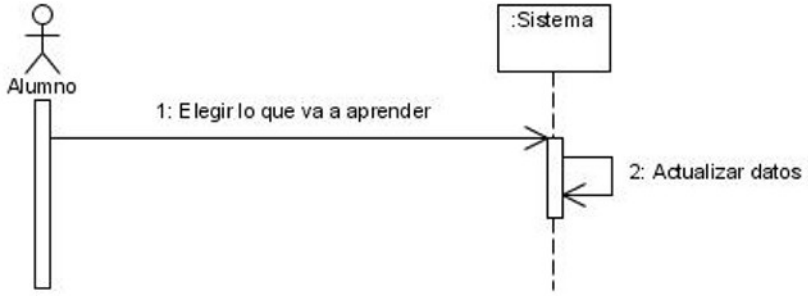


Figura 5.20: Diagrama del caso de uso Seleccionar manualmente.

5.6.10. Abrir fichero existente

UC-0012	***Abrir fichero existente***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de abrir un fichero que ya existe*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de abrir un fichero existente*
	2	El sistema *muestra los ficheros existentes*
	3	El actor Alumno (ACT-0001) *elige un fichero*
	4	El sistema *muestra el fichero que ha elegido el usuario*

Figura 5.21: Descripción caso de uso Abrir fichero existente.

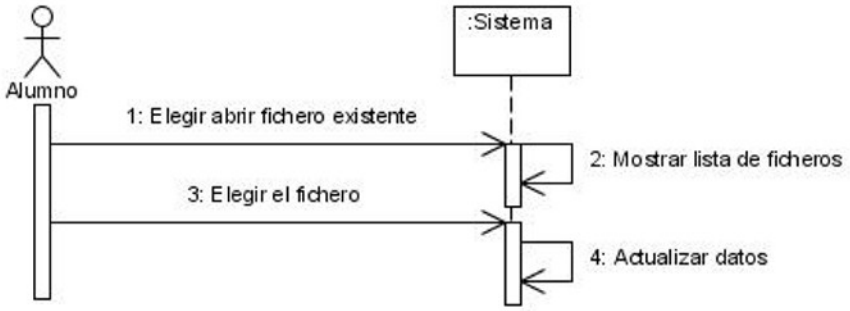


Figura 5.22: Diagrama del caso de uso Abrir fichero existente.

5.6.11. Abrir fichero nuevo

UC-0013	***Abrir fichero nuevo***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de abrir un fichero nuevo*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de abrir un fichero nuevo*
	2	El sistema *muestra un fichero nuevo*

Figura 5.23: Descripción caso de uso Abrir fichero nuevo

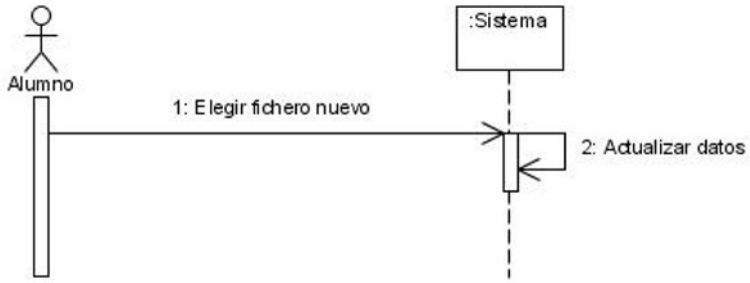


Figura 5.24: Diagrama del caso de uso Abrir fichero nuevo

5.6.12. Guardar fichero

UC-0014	*Guardar fichero*	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de guardar el fichero actual*	
Secuencia normal	*Paso*	*Acción*
	1	El actor Alumno (ACT-0001) *elige la opción de guaradr el fichero actual*
	2	El sistema *muestra un dialogo en el que el usuario eligirá la ubicación y el nombre del fichero a guardar*
	3	El actor Alumno (ACT-0001) *elige el nombre del fichero y donde lo quiere guardar*
	4	El sistema *realiza los cambios que mando el usuario*

Figura 5.25: Descripción caso de uso Guardar fichero.

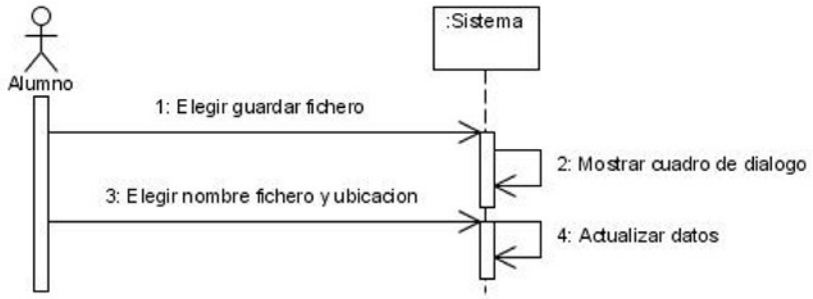


Figura 5.26: Diagrama del caso de uso Guardar fichero

5.6.13. Imprimir fichero

UC-0015	*Imprimir documento*	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *cuando el usuario elige la opción de imprimir el documento actual*	
Secuencia normal	*Paso*	*Acción*
	1	El actor Alumno (ACT-0001) *elige la opción de Imprimir*
	2	El sistema *muestra un mensaje donde te muestra el proceso de impresión del documento y donde te da la opción de cancelar la impresión del documento*
	3	El sistema *realiza la impresión del documento*

Figura 5.27: Descripción caso de uso Imprimir fichero.

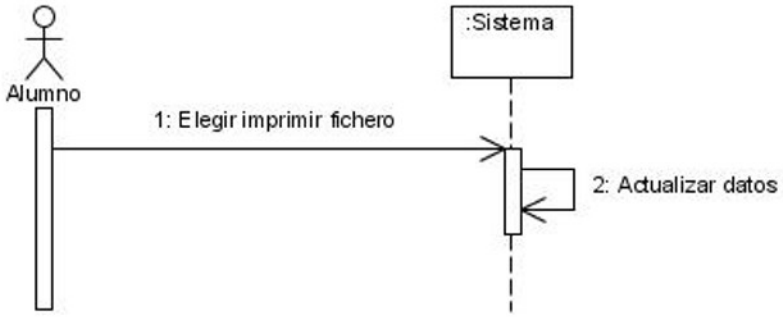


Figura 5.28: Diagrama del caso de uso Imprimir fichero

5.6.14. Vista preliminar

UC-0016	***Vista preliminar***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de vista preliminar*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de Vista preliminar del documento*
	2	El sistema *muestra una ventana en la que se ve el documento antes de imprimirlo*

Figura 5.29: Descripción caso de uso Vista Preliminar.

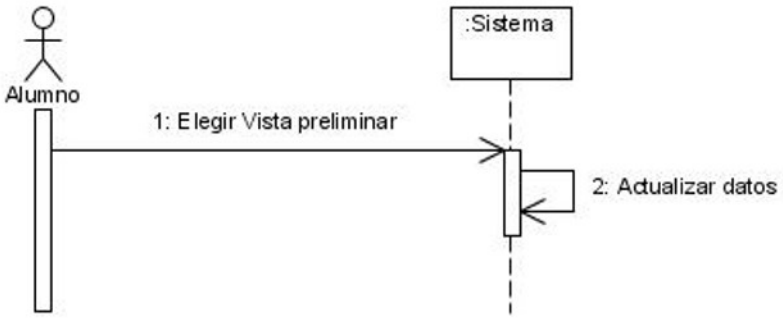


Figura 5.30: Diagrama del caso de uso Vista Preliminar

5.6.15. Reconocer escritura

UC-0017	***Reconocer escritura***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de reconocer lo que se ha escrito*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de Reconocer*
	2	El sistema *muestra lo que ha reconocido en la caja de texto*

Figura 5.31: Descripción caso de uso Reconocer escritura.

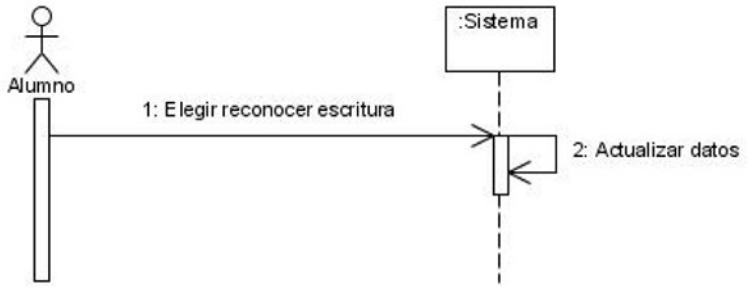


Figura 5.32: Diagrama del caso de uso Reconocer escritura

5.6.16. Escribir tinta

UC-0018	***Escribir tinta***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *cuando el usuario elige la opción de escribir*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de Escribir tinta digital*
	2	El sistema *se actualiza y muestra en el área de escritura un cursor lápiz que indica que puede comenzar a escribir*

Figura 5.33: Descripción caso de uso Escribir tinta.

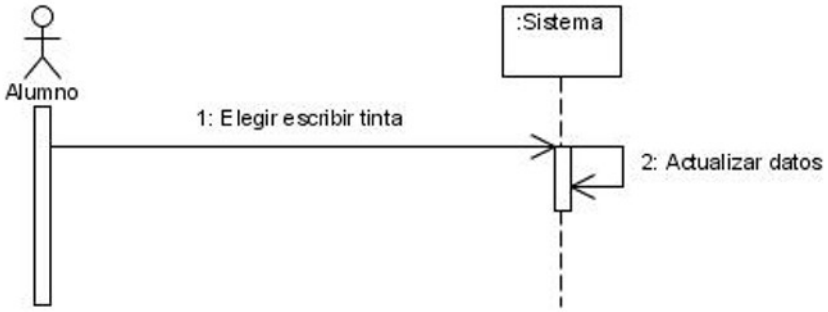


Figura 5.34: Diagrama del caso de uso Escribir tinta

5.6.17. Borrar tinta

UC-0019	***Borrar tinta***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción Borrar de borrar tinta digital*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de Borrar*
	2	El sistema *se actualiza y muestra en el área de escritura un cursor goma que indica que puede comenzar a borrar pinceladas*

Figura 5.35: Descripción caso de uso Borrar tinta.

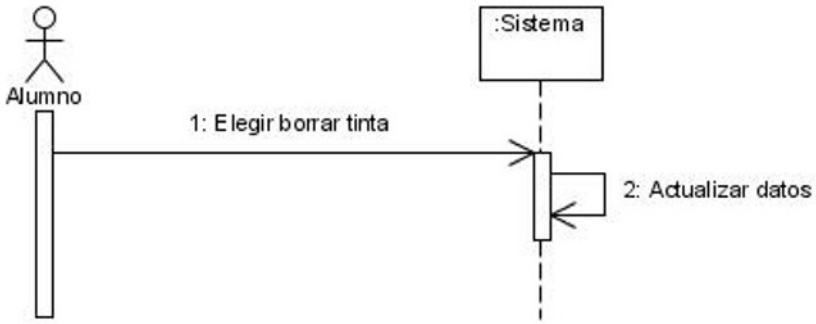


Figura 5.36: Diagrama del caso de uso Borrar tinta

5.6.18. Cortar tinta

UC-0020	***Cortar tinta***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de Cortar tinta digital*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de Cortar*
	2	Si *hay tinta seleccionada,* el sistema *corta las pinceladas seleccionadas*

Figura 5.37: Descripción caso de uso Cortar tinta.

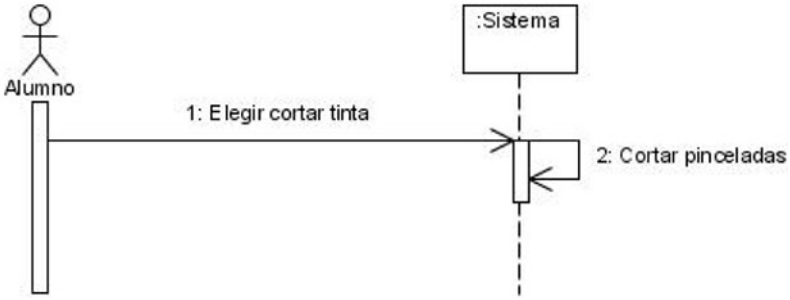


Figura 5.38: Diagrama del caso de uso Cortar tinta

5.6.19. Pegar tinta

UC-0021	***Pegar tinta***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de pegar tinta digital que haya sido cortada o copiada*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de Pegar*
	2	Si *hay tinta que haya sido copiada o cortada,* el sistema *pega esa tinta en la esquina superior izquierda*

Figura 5.39: Descripción caso de uso Pegar tinta.

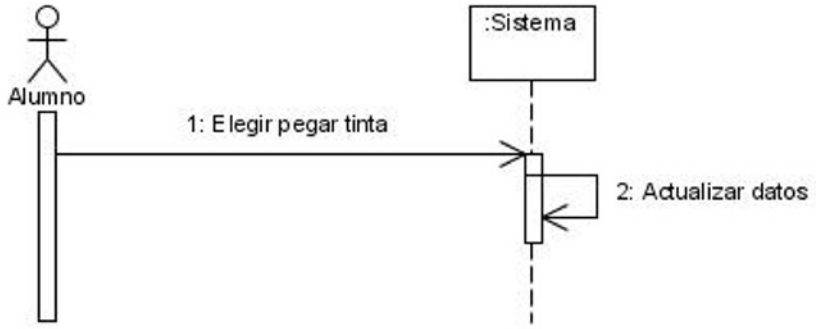


Figura 5.40: Diagrama del caso de uso Pegar tinta

5.6.20. Copiar tinta

UC-0022	***Copiar tinta***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de copiar tinta seleccionada*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de Copiar*
	2	Si *hay tinta seleccionada,* el sistema *copia las pinceladas seleccionadas*

Figura 5.41: Descripción caso de uso Copiar tinta.

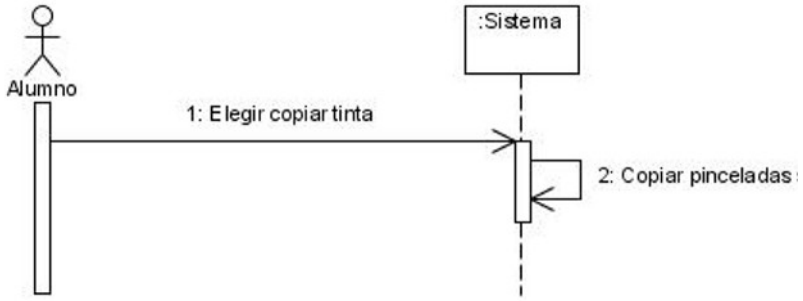


Figura 5.42: Diagrama del caso de uso Copiar tinta

5.6.21. Seleccionar tinta

UC-0023	*Seleccionar tinta*	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de seleccionar tinta*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de Seleccionar tinta digital*
	2	El sistema *se actualiza y muestra en el área de escritura un cursor flecha que indica que puede comenzar a seleccionar*

Figura 5.43: Descripción caso de uso Seleccionar tinta.

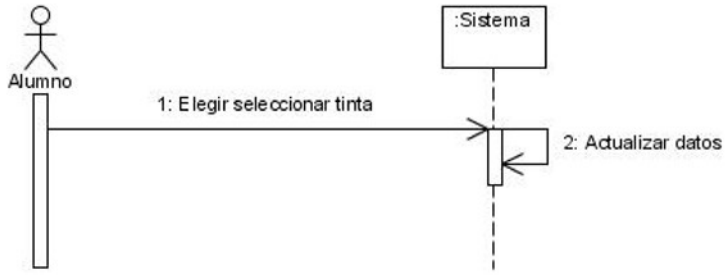


Figura 5.44: Diagrama del caso de uso Seleccionar tinta

5.6.22. Usar Zoom

UC-0024	*Usar Zoom*	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de utilizar el Zoom*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de hacer Zoom*
	2	El sistema *se actualiza mostrando los cambios*

Figura 5.45: Descripción caso de uso Usar zoom.

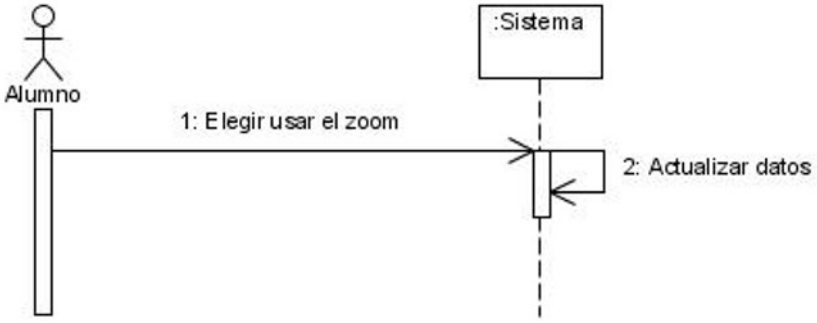


Figura 5.46: Diagrama del caso de uso Usar zoom

5.6.23. Usar guías de escritura

UC-0025	***Usar guías de escritura***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de utilizar guias de escritura*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige el tipo de guía de escritura que quiere*
	2	El sistema *se actualiza mostrando las guías de escritura elegidas*

Figura 5.47: Descripción caso de uso Usar guias de escritura.

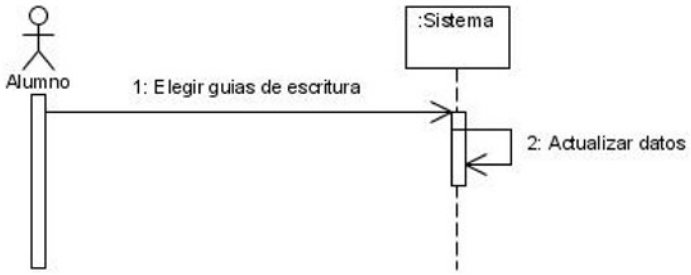


Figura 5.48: Diagrama del caso de uso Usar guias de escritura

5.6.24. Borrar caja de texto

UC-0026	***Borrar caja de texto***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de borrar el contenido del cuadro de texto donde el sistema escribe el resultado de reconocer la escritura*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de Borrar texto*
	2	El sistema *borra el contenido de la caja donde escribe el resultado del reconocimiento de escritura*

Figura 5.49: Descripción caso de uso Borrar caja de texto.

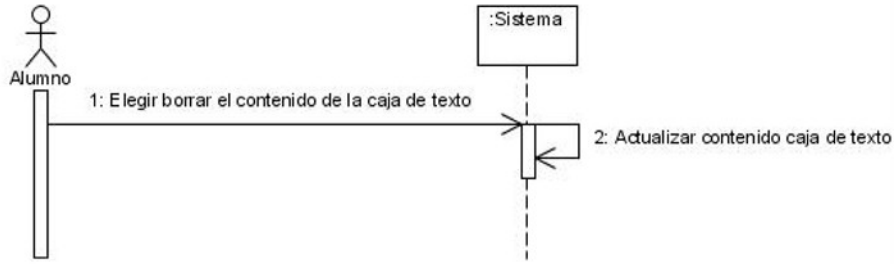


Figura 5.50: Diagrama del caso de uso Borrar caja de texto

5.6.27. Cambiar color de fondo

UC-0027	***Elegir color de fondo***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de ponerle color al fondo del área de escritura*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de cambiar el Color de fondo*
	2	El sistema *muestra un cuadro de diálogo donde el usuario puede elegir entre distintos colores*
	3	El actor Alumno (ACT-0001) *elige un color*
	4	El sistema *se actualiza mostrando los cambios seleccionados*

Figura 5.51: Descripción caso de uso Elegir color de fondo.

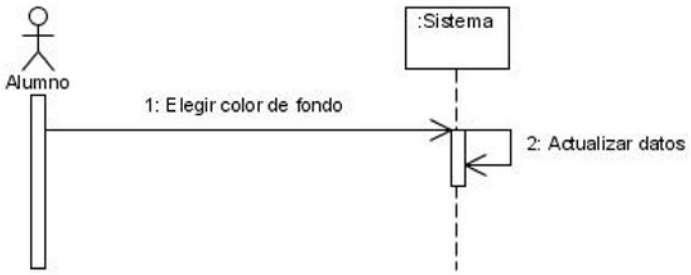


Figura 5.52: Diagrama del caso de uso Elegir color de fondo

5.6.26. Limpiar fondo

UC-0028	***Limpiar fondo***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de Limpiar el fondo*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de limpiar el fondo*
	2	El sistema *limpia el fondo poniendo el color de fondo a blanco y quitando si hubiese la guías de escritura*

Figura 5.53: Descripción caso de uso Limpiar fondo.

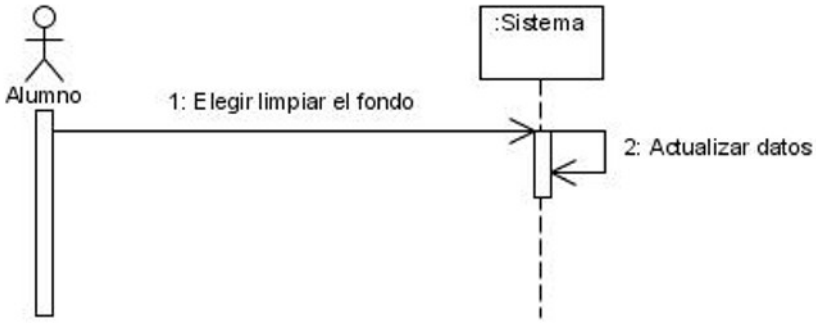


Figura 5.54: Diagrama del caso de uso Limpiar fondo

5.6.27. Cambiar color de la tinta

UC-0029	***Cambiar color de la tinta***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de cambiar el color de la tinta*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige el color que quiere para la tinta*
	2	El sistema *se actualiza y cambia el color de la tinta al seleccionado por el usuario*

Figura 5.55: Descripción caso de uso Cambiar color de la tinta.

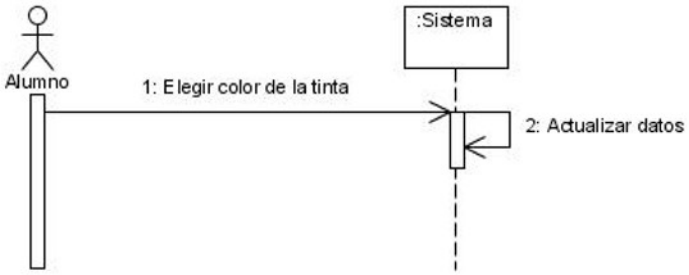


Figura 5.56: Diagrama del caso de uso Cambiar color de la tinta

5.6.28. Cambiar el grosor de la pincelada

UC-0030	***Cambiar el grosor de la pincelada***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige el nuevo grosor que quiere para la tinta*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige el grosor que quiere para la tinta*
	2	El sistema *se actualiza y muestra el nuevo grosor elegido por el usuario*

Figura 5.57: Descripción caso de uso Cambiar el grosor de la tinta

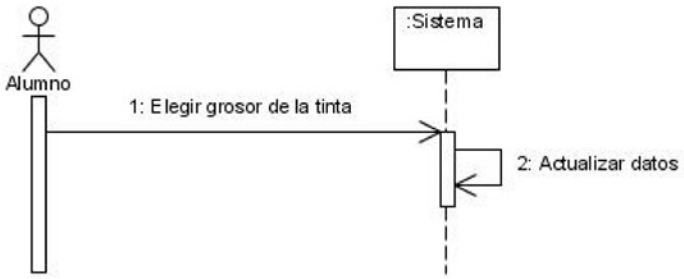


Figura 5.58: Diagrama del caso de uso Cambiar el grosor de la tinta

5.6.29. Colorear dibujos

UC-0031	***Colorear dibujos***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de colorear dibujos que dispone el programa*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de Dibujos para colorear*
	2	El sistema *muestra una lista con los dibujos que se pueden elegir*
	3	El actor Alumno (ACT-0001) *escoge un dibujo para colorear*
	4	El sistema *se actualiza y muestra el dibujo para colorear*

Figura 5.59: Descripción caso de uso Colorear dibujos.

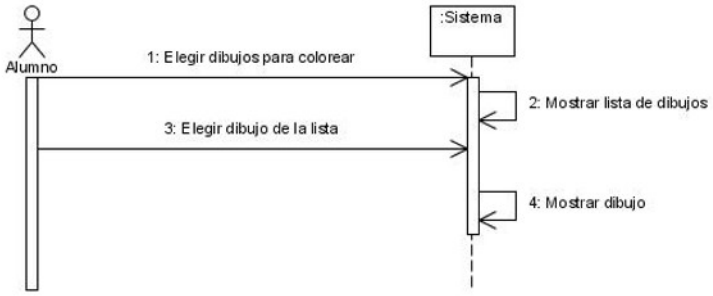


Figura 5.60: Diagrama del caso de uso Colorear dibujos

5.6.30. Volver a la pantalla anterior

UC-0032	***Volver a la pantalla anterior***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de volver a la pantalla anterior*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de volver a la pantalla anterior*
	2	El sistema *muestra la pantalla anterior*

Figura 5.61: Descripción caso de uso Volver a la pantalla anterior.

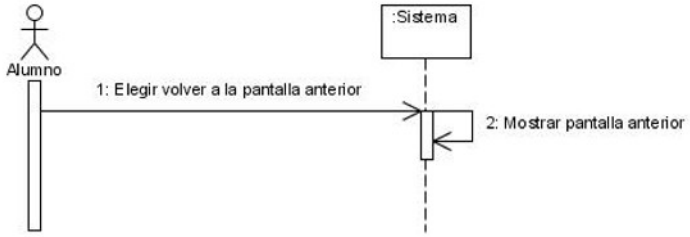


Figura 5.62: Diagrama del caso de uso Volver a la pantalla anterior

5.6.31. Comenzar a jugar

UC-0033	***Comenzar a jugar***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de comenzar a jugar en el programa*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige la opción de entrar en el programa*
	2	El sistema *muestra las opciones de entretenimiento que tiene el programa*

Figura 5.63: Descripción caso de uso Comenzar a jugar.

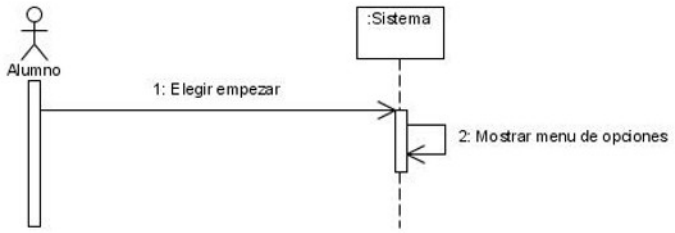


Figura 5.64: Diagrama del caso de uso Comenzar a jugar

5.6.32. Salir del programa

UC-0034	***Salir del programa***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario elige la opción de salir del programa y volver a Windows*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige salir de la aplicación*
	2	El sistema *cierra el programa*

Figura 5.65: Descripción caso de uso Salir del programa.

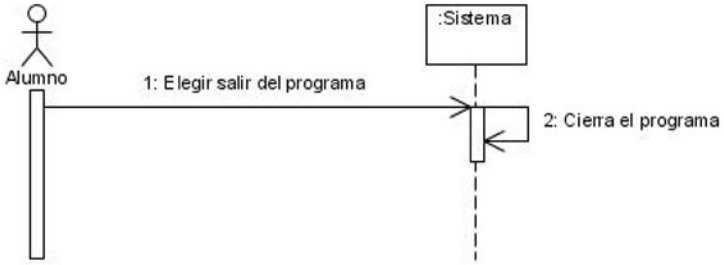


Figura 5.66: Diagrama del caso de uso Salir del programa

5.7. Modelo de clases

El modelo de objetos muestra la estructura estática de datos correspondientes al sistema del mundo real, y la organiza en segmentos manejables describiendo clases de objetos del mundo real, y sus relaciones entre sí. Lo más importante es la organización de más alto nivel del sistema, en clases conectadas mediante asociaciones.

5.7.1. Diagrama inicial de clases

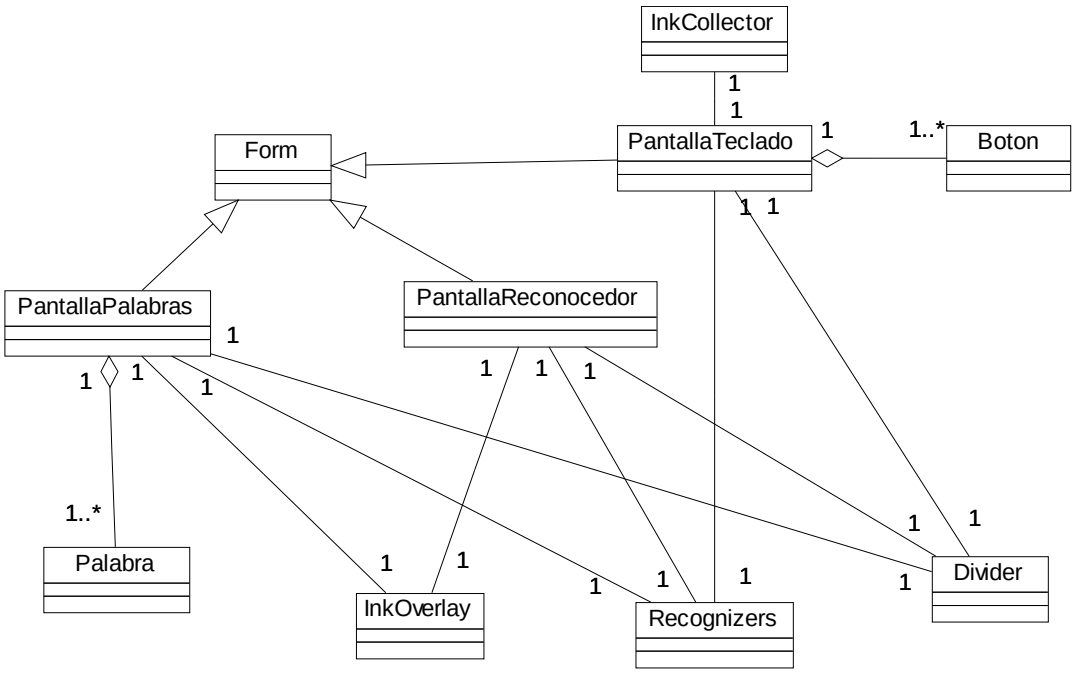


Figura 5.67: Diagrama inicial de clases

Capítulo 6

DISEÑO

Una vez completada la fase del análisis, y antes de pasar a la implementación y programación, es necesaria una actividad de nivel superior.

La fase de diseño es una fase intermedia entre la vista abstracta de un sistema y la implementación real de éste. Los productos de esta fase pueden ser más cercanos a una visión abstracta o una visión implementable.

6.1. Casos de uso

6.1.1. Consultar Ayuda

UC-0001	***Consultar ayuda***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario pulse el botón Ayuda de alguna de las pantallas*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *pulsa el botón de Ayuda*
	2	El sistema *muestra información sobre la página desde la que se pulsó el botón ayuda*
Excepciones	***Paso***	***Acción***
	2	Si *el usuario desea consultar información sobre otra pantalla que no sea desde la que ha accedido,* el actor Alumno (ACT-0001) *debe pulsar sobre la pestaña que lleve el nombre de la pantalla sobre la que se quiere consultar información*, a continuación este caso de uso *continúa*
Importancia	importante	
Comentarios	El actor puede hacer las consultas a la ayuda en cualquier momento mediante los botones Hardware de Ayuda	

Figura 6.1: Descripción caso de uso Consultar ayuda.

6.1.2. Aprender letras y números

UC-0005	***Aprender letras y números***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario quiera aprender a escribir las letras del abecedario y algunos números*	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige pulsa el dibujo de aprender letras y números*
	2	El sistema *muestra el teclado con el que se pueden aprender las letras y los números*
	3	Si *el usuario pulsa el botón Dados,* se realiza el caso de uso Seleccionar automáticamente (UC-0008)
	4	Si *el usuario pulsa el botón de la letra o el número que quiere aprender,* se realiza el caso de uso Seleccionar manualmente (UC-0009)
	5	El actor Alumno (ACT-0001) *escribe la letra o el número seleccionado*
	6	El actor Alumno (ACT-0001) *pulsa el botón Leer*
	7	El sistema *muestra un mensaje indicando si el texto escrito por el usuario es igual a la letra o el número seleccionado*
Importancia	vital	
Comentarios	El sistema desea que el usuario pueda aprender a escribir letras y números, para ello las letras del abecedario y los números del 1 al 9 se encuentran agrupados en un teclado.	

Figura 6.2: Descripción caso de uso Aprender letras y números.

6.1.3. Modificar Configuración

UC-0002	***Modificar Configuración***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario quiera modificar alguna de las opciones de configuración*	
Precondición	estado actual sin modificar	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *pulsa el botón de configuración situado en la pantalla inicial*
	2	El sistema *le muestra la pantalla de configuración donde hay opciones que puede modificar*
	3	El actor Alumno (ACT-0001) *elige los cambios que desea realizar*
	4	El sistema *actualiza los cambios*
Postcondición	estado actual modificado	
Importancia	importante	
Comentarios	Este caso de uso puede realizarse desde la pantalla inicial del programa	

Figura 6.3: Descripción caso de uso Modificar Configuración.

6.1.4. Guardar Fichero

UC-0014	***Guardar fichero***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario pulsa el botón de guardar fichero* o durante la realización de los siguientes casos de uso: [UC-0012] Abrir fichero existente, [UC-0013] Abrir fichero nuevo, [UC-0031] Colorear dibujos, [UC-0032] Volver a la pantalla anterior	
Precondición	Debe haber tinta para poder guardar el fichero Hay n ficheros guardados	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *elige guardar el fichero actual*
	2	Si *hay tinta que se puede guardar,* el sistema *muestra un cuadro de dialogo donde el usuario podrá decidir como llamar al fichero y donde guardarlo o decidir si desea cancelar la operación*
	3	El actor Alumno (ACT-0001) *elige el nombre del fichero que quiere guardar y su ubicación*
	4	El sistema *actualiza los cambios y continua*
Postcondición	Hay n+1 ficheros guardados	
Excepciones	***Paso***	***Acción***
	3	Si *el usuario pulsa cancelar la operación,* el sistema *no guarda el fichero*, a continuación este caso de uso *queda sin efecto*
Importancia	importante	
Comentarios	El sistema desea que el usuario tenga guardados los ficheros que el quiera	

Figura 6.4: Descripción caso de uso Guardar Fichero.

6.1.5. Seleccionar automáticamente

UC-0008	***Seleccionar automáticamente***	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario decide que quiere que el sistema le escoja de forma aleatoria una palabra o una letra o un número para aprender su escritura* o durante la realización de los siguientes casos de uso: [UC-0005] Aprender letras y números, [UC-0007] Aprender palabras	
Secuencia normal	***Paso***	***Acción***
	1	El actor Alumno (ACT-0001) *pulsa el botón dados*
	2	El sistema *elige aleatoriamente la palabra a aprender*
Importancia	importante	
Comentarios	Este caso de uso se puede realizar desde la PantallaPalabras o desde la PantallaTeclado	

Figura 6.5: Descripción caso de uso Seleccionar Automáticamente.

6.2. Diagramas de Secuencia

En el diagrama de secuencia se muestra el orden de las llamadas en el sistema. Se utiliza un diagrama para cada caso de uso a representar. Es imposible representar en un solo diagrama todas las secuencias posibles del sistema, por ello se escoge un punto de partida. El diagrama se forma con los objetos que forman parte de la secuencia, estos se sitúan en la parte superior. De estos objetos sale un línea que indica su vida en el sistema. Esta línea simple se convierte en gruesa cuando representa que el objeto tiene el foco del sistema, es decir, cuando él está activo.

6.2.1. Diagrama de secuencia Consultar Ayuda

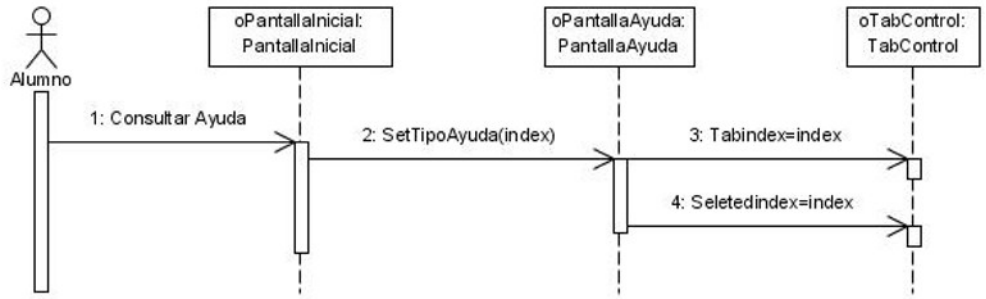


Figura 6.6: Diagrama de secuencia Consultar ayuda.

6.2.2. Diagrama de secuencia Modificar Configuración

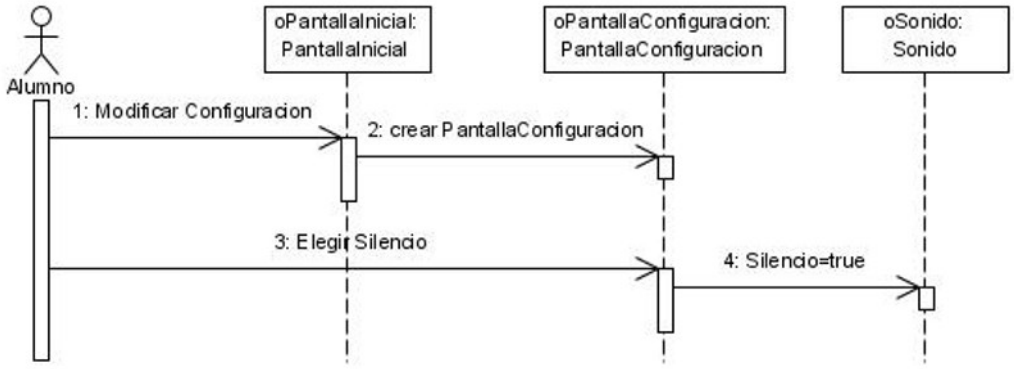


Figura 6.7: Diagrama de secuencia Modificar Configuración.

6.2.3. Diagrama de secuencia Aprender letras y números

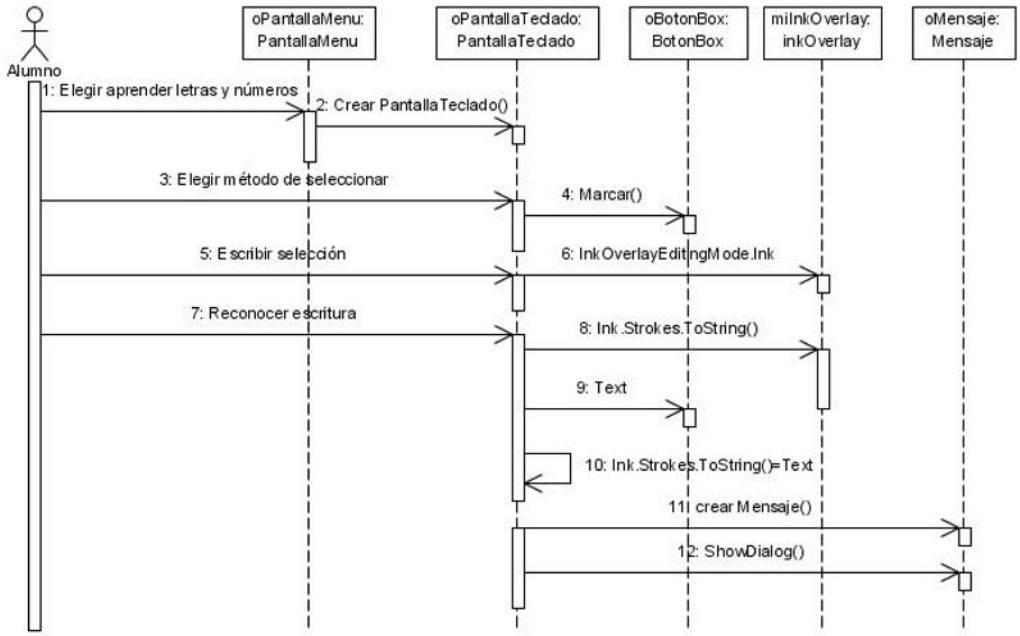


Figura 6.8: Diagrama de secuencia Aprender letras y números.

6.2.4. Diagrama de secuencia Seleccionar Automáticamente

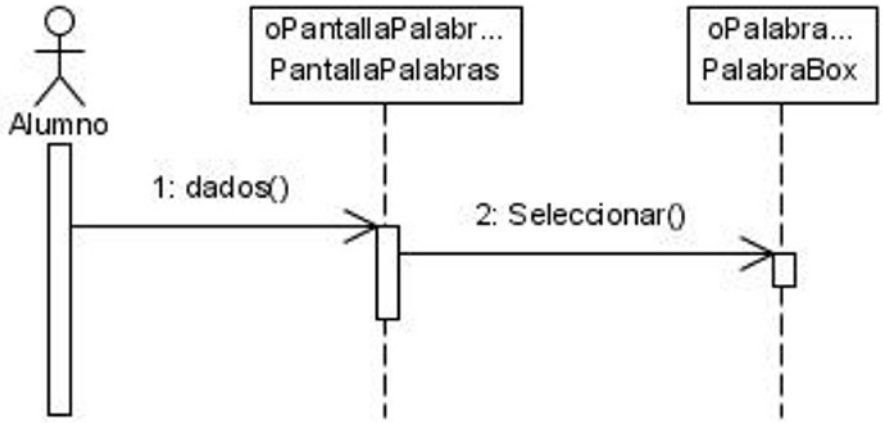


Figura 6.9: Diagrama de secuencia Seleccionar Automáticamente.

6.2.5. Diagrama de secuencia Guardar Fichero

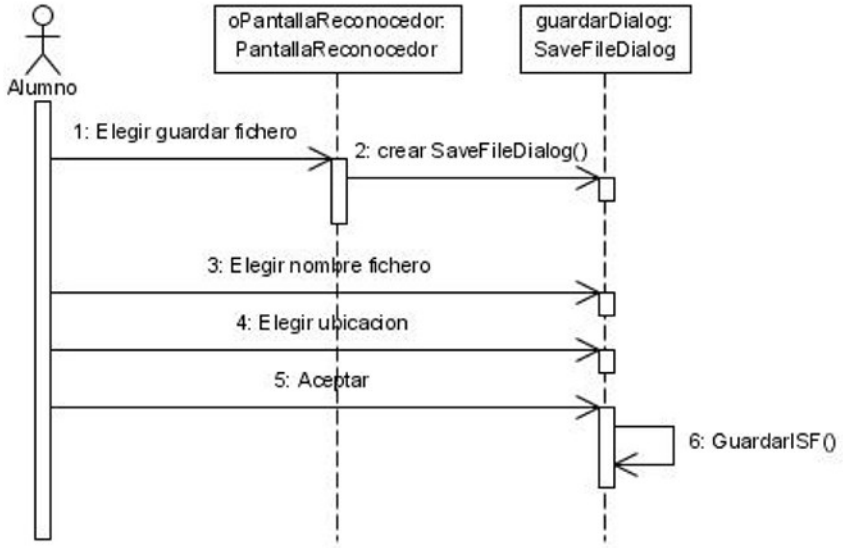


Figura 6.10: Diagrama de secuencia Guardar Fichero.

6.3. Diagrama de clases final

El primer paso en la elaboración de un modelo de objetos es encontrar las clases necesarias para la aplicación. Las clases suelen corresponderse con sustantivos o entidades físicas. Aunque hay que evitar estructuras propias de la computadora.
En nuestro caso la mayoría de las clases derivan de la clase Form y tenemos pocas clases creados por nosotros, debido a que este proyecto se basa en la utilización de librerías de Microsoft sobre la tinta digital.
A continuación se describen las clases encontradas con una breve descripción:

- **BotonBox:** Es un control de usuario que es utilizado para cargar los botones del Teclado en tiempo de ejecución
- **PalabraBox:** Es un control de usuario que es utilizado para cargar las palabras que se van a aprender en tiempo de ejecución
- **LetraAnimada:** Es un control de usuario creado para cargar en tiempo de ejecución los gifs animados de letras de la Pantalla Inicial y de la Pantalla Despedida.
- **Sonido:** Es una clase estática diseñada para controlar todo sobre los sonidos del programa y las librerías relacionadas con el sonido.
- **PantallaReconocedor:** representa la interfaz del reconocedor de escritura.
- **PantallaDibujo:** es la interfaz creada para dibujar.
- **PantallaAyuda:** es la pantalla que se muestra cuando el usuario solicite Consultar la ayuda.
- **PantallaConfiguracion:** Es la pantalla que se muestra cuando se decide modificar la configuración de las opciones posibles.
- **PantallaDespedida:** Es la interfaz que se muestra antes de salir de la aplicación.

- **PantallaInicial:** es la interfaz que se muestra al comenzar la aplicación y desde la que se tiene acceso al resto de pantallas
- **PantallaMenu:** es la interfaz que muestra las opciones que tiene el programa
- **PantallaMenuPalabras:** es la interfaz desde la que podremos elegir la familia de palabras que queremos que se cargué.
- **PantallaPalabras:** es la interfaz en la que se muestran las palabras con sus dibujos que se van a aprender.
- **PantallaTeclado:** es la interfaz que muestra las letras y números que se pueden aprender.
- **Mensaje:** es la pantalla creada para mostrar información al usuario.
- **MensajeGuardar:** creada para preguntar al usuario si desea guardar los cambios realizados a un documento que fue modificado desde la última vez que fue guardado.
- **MensajeSalida:**creada para preguntar al usuario si desea abandonar el programa para volver a Windows.
- **SobreAyuda:** es la interfaz creada para mostrar más información sobre las funciones de las pantallas en la Ayuda
- **SobreAyudaEjemplo:** es la interfaz creada para mostrar más información sobre el funcionamiento de las pantallas al consultar la Ayuda.
- **SobreAyudaMenu:** Interfaz creada para mostrar información sobre el funcionamiento de los menús al consultar la Ayuda

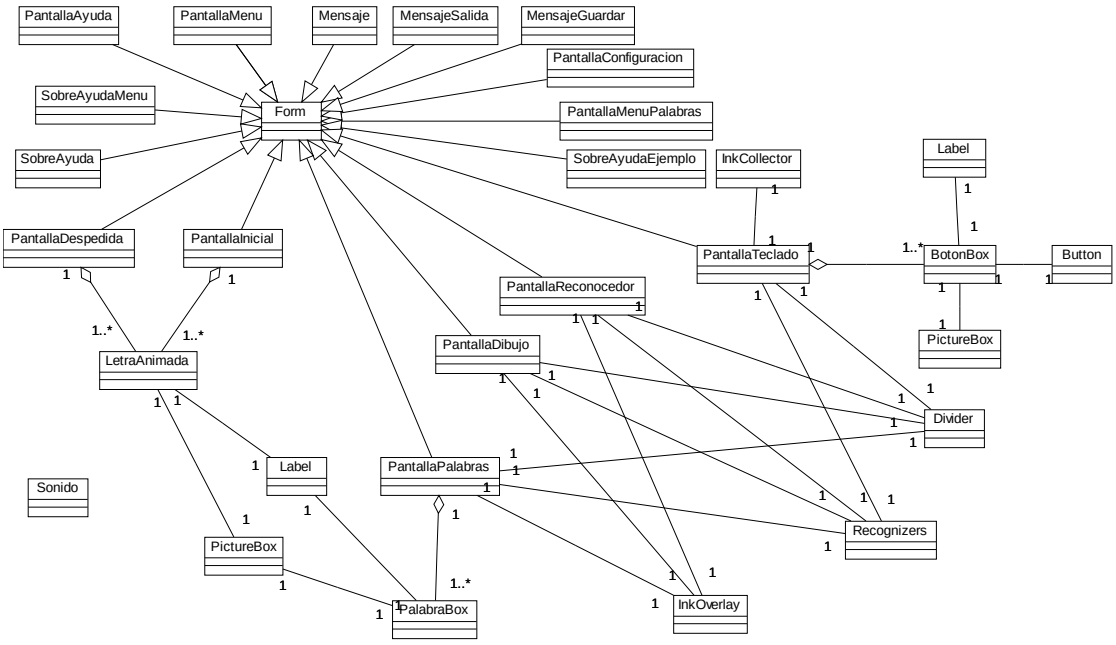


Figura 6.11 Diagrama de clases

6.4. Especificación de clases

Clase BotonBox
Es un control de usuario que es utilizado para cargar los botones del Teclado en tiempo de ejecución
Atributos
- Texto:String: indica el número o la letra que representa el botón
- FicheroImagen:String: indica el fichero que contiene la imagen del botón
- FicheroSonido:String: indica el fichero que contiene el sonido del botón
- Seleccionado: Bool: indica si el botón está seleccionado o no

Métodos
- LeerSonido(): es el encargado de leer el sonido
- Seleccionar(cierto:Bool): es el que selecciona la letra o el número que hay que escribir
- Marcar(cierto:Bool): es el que realiza la animación

Clase PalabraBox
Es un control de usuario que es utilizado para cargar las palabras que se van a aprender en tiempo de ejecución
Atributos
- Texto:String: indica el nombre de la palabra que representa
- FicheroImagen:String: indica el fichero que contiene la imagen de la palabra
- FicheroSonido:String: indica el fichero que contiene el sonido del palabra
- Seleccionado: Bool: indica si la palabra está seleccionada o no

Métodos
- LeerSonido(): es el encargado de leer el sonido
- Seleccionar(cierto:Bool): es el que selecciona la palabra que hay que escribir
- Marcar(cierto:Bool): es el que realiza la animación

Clase LetraAnimada
Es un control de usuario creado para cargar en tiempo de ejecución los gifs animados de letras de la Pantalla Inicial y de la Pantalla Despedida.
Atributos:
- Letra:Char: indica la letra que representa
- PathImagenes:String: indica la ruta de donde se encuentra el gif animado a representar

Clase Sonido:
Es una clase estática diseñada para controlar todo sobre los sonidos del programa y las librerías relacionadas con el sonido.
Atributos:
- NoMusica:Bool: indica si el usuario ha escogido la opción de que no se escuchen las canciones del programa
- Silencio:Bool: muestra si el usuario ha elegido que el programa se ejecute sin sonidos.

Métodos:
- Parar(): es el encargado de parar el sonido
- PlayAsync(String): se encarga de ejecutar el sonido del fichero indicado en el String de forma asíncrona
- PlayMusic(String): se encarga de ejecutar las músicas que hay en el programa
- PlaySound(String, IntPtr,SoundFlags): se encarga de ejecutar el sonido indicado en el String y lo ejecuta teniendo en cuenta el modo indicado en IntPtr(de forma asíncrona o síncrona) y las SoundFlags que se utilizan.

- PlaySync(String):se encarga de ejecutar el sonido del fichero indicado en el String de forma síncrona

Capítulo 7

IMPLEMENTACIÓN

7.1 Software necesario

7.1.1. Microsoft Tablet PC Development Kit 1.7

Versiones del Kit de Desarrollo Software (SDK)

- El lanzamiento de la versión 1.1 sucedió a la versión inicial 1.0, actualizándola solamente en cuanto a documentación se refiere. La plataforma binaria no varió entre la versión 1.1 del SDK y la versión exportada de Windows Tablet PC Edition. Las aplicaciones desarrolladas con este kit ya no necesitan redistribuir ningún componente para utilizar características propias de la plataforma.

- La versión 1.5 sucede a la versión 1.1 añadiendo características de análisis de tinta y el panel de entrada a través de lápiz. Si se instala esta versión debe establecerse una referencia al ensamblado Microsoft.Ink en aplicaciones escritas usando la versión anterior.

- La versión más actual es la 1.7 ampliando la versión 1.5 en los siguientes aspectos:

  - Añade el objeto ***RealTimeStylus*** que proporciona la oportunidad de creación e interpretación de tinta (*Ink*) a bajo nivel accediendo y modificando paquetes de datos (*packets*), esto quiere decir que dentro de una pincelada (*stroke*), propiedades como el número de puntos en la pincelada y datos de esos puntos describen lo que es la pincelada en si. El conjunto de datos enviados por el dispositivo Tablet ya sea en un ordenador Tablet PC o la tableta gráfica en nuestro caso hace corresponder a cada punto simple una pincelada formando paquetes.
  - **Controles Web** añadidos en la Biblioteca Administrada, para facilitar la ejecución de ensamblados administrados de páginas web.
  - **Context Tagging Tool** (Herramienta de marca de contexto) proporciona una manera de fijar la información de contexto para editar controles en aplicaciones.

7.1.2. Módulo de reconocimiento de escritura

El módulo de reconocimiento de escritura e interfaz de usuario multilingüe (MUI) para Microsoft Windows XP Tablet PC Edition se ha diseñado exclusivamente para entornos multilingües de Tablet PC.

El módulo de interfaz de usuario multilingüe (MUI) para Windows XP es un paquete que contiene cadenas de interfaz de usuario (menús, iconos, cuadros de diálogo, archivos de Ayuda) localizadas en 33 idiomas. Dicho módulo permite a los administradores seleccionar el conjunto de idiomas de la interfaz de usuario que se va a implementar y permite a los usuarios finales activar cualquiera de estos idiomas (o cambiar de uno a otro según sea necesario).

El módulo de reconocimiento de escritura y MUI para Tablet PC permite a los usuarios cambiar la configuración de la interfaz de usuario a los idiomas existentes admitidos por Tablet PC (inglés, francés, alemán, coreano, chino simplificado y chino tradicional), además de una serie de idiomas adicionales, incluidos danés, neerlandés, finlandés, español, italiano, noruego, portugués brasileño, portugués europeo y sueco. Los usuarios pueden instalar un reconocedor de escritura a mano para español de España, además de los ya instalados para inglés, francés, alemán, japonés, coreano, chino simplificado y chino tradicional. Asimismo, los usuarios pueden instalar un nuevo reconocedor de voz para chino tradicional, además de los existentes para inglés, japonés y chino simplificado.

El módulo de reconocimiento de escritura y MUI para Tablet PC contiene todos los motores de reconocimiento de escritura a mano y de voz disponibles en Windows XP Tablet PC Edition e incorpora un nuevo motor de escritura a mano para español (España) y un nuevo motor de voz para chino tradicional.

Dicho módulo incluye los ocho reconocedores de escritura a mano siguientes: inglés, francés, alemán, español (España), japonés, coreano, chino simplificado y chino tradicional. Incluye también los cuatro reconocedores de voz siguientes: inglés, japonés, chino simplificado y chino tradicional

7.2. Software utilizado

A continuación mencionamos las aplicaciones y paquetes que hemos usado:

- Microsoft Windows XP Edición Profesional.
- Microsoft Visual Studio.NET 2003
- Microsoft Word 2000
- Microsoft PhotoDraw 2000
- Microsoft PowerPoint
- Rational Rose C++ Demo 4.0
- Borland Together 4
- REM 1.2.2
- OpenOffice.org Writer
- Audacity
- Adobe Reader 6.0
- Grabadora de sonidos de Windows XP
- Microsoft .NET Framework SDK v. 1.1.

- Microsoft Tablet PC Platform SDK
- Visual Paradigm for UML

7.3. Hardware empleado

Equipo principal de desarrollo:

- S.O. Microsoft Windows XP Edición Profesional Service Pack 2
- Pentium IV 2,4 Mhz
- 256 Mb Ram
- HD 60Gb

Equipo secundario de desarrollo:

- S.O. Microsoft Windows XP Edición Profesional Service Pack 2
- Pentium M Tecnología Intel-Centrino 2.0 Mhz
- 1 Gb Ram
- HD 40 Gb

Tableta gráfica Wacom Graphire modelo CTE-430 (Sappire)

Impresora Epson Stylus Photo 810.

Capítulo 8

PRUEBAS

8.1. Pruebas durante el desarrollo

En el desarrollo de nuestro programa, cada vez que hemos realizado una rutina se ha probado su funcionamiento para donde fue diseñada por primera vez.

A continuación pongo algunos ejemplos de las pruebas que se realizaron:

- La rutina Marcar de la clase PalabraBox: esta rutina fue probada después de realizarla para comprobar que una PalabraBox se quedaba marcada como seleccionada. Esta rutina también se utiliza para mostrar al usuario que BotonBox es el seleccionado

- La rutina Dados de las clases PantallaTeclado y PantallaPalabras: esta rutina se probó después de su realización para comprobar que se seleccionaba aleatoriamente un BotonBox o una PalabraBox

8.2. Fase de pruebas del proyecto

Descripción	***Comprobación del botón Configuración***
Acción realizada	Pulsación del botón Configuración situado más a la izquierda de la Pantalla Inicial
Resultado esperado	Mostrar la Pantalla de Configuración
Resultado obtenido	Pantalla de Configuración mostrada
Observaciones	Correcto

Figura 8.1 Comprobación del botón Configuración

Descripción	***Comprobación del botón Ayuda***
Acción realizada	Pulsación del botón Ayuda de cualquier pantalla
Resultado esperado	Mostrar la Pantalla de Ayuda
Resultado obtenido	Pantalla de Ayuda mostrada
Observaciones	Correcto

Figura 8.2 Comprobación del botón Ayuda

Descripción	*Comprobación del botón Empezar*
Acción realizada	Pulsación del botón verde Empezar situado en el centro de la Pantalla Inicial
Resultado esperado	Mostrar la Pantalla de Menu
Resultado obtenido	Pantalla de Menu mostrada
Observaciones	Correcto

Figura 8.3 Comprobación del botón Empezar

Descripción	*Comprobación del botón Salir*
Acción realizada	Pulsación del botón rojo Salir situado en el centro de la Pantalla Inicial
Resultado esperado	Mostrar el Mensaje de salida
Resultado obtenido	Mensaje de salida mostrado
Observaciones	Correcto

Figura 8.4 Comprobación del botón Salir

Descripción	*Comprobación del botón Atrás*
Acción realizada	Pulsación del botón Atrás situado más a la izquierda de cualquier pantalla
Resultado esperado	Mostrar la pantalla anterior a la actual
Resultado obtenido	Pantalla anterior a la actual mostrada
Observaciones	Correcto

Figura 8.5 Comprobación del botón Atrás

Descripción	*Comprobación de la picture Reconocedor de escritura*
Acción realizada	Pulsación de la picture Reconocedor de escritura situado en la esquina inferior izquierda del menu de opciones
Resultado esperado	Mostrar la pantalla del Reconocedor de escritura
Resultado obtenido	Pantalla del Reconocedor de escritura mostrada
Observaciones	Correcto

Figura 8.6 Comprobación de la picture Reconocedor de escritura

Descripción	*Comprobación de la picture Dibujar*
Acción realizada	Pulsación de la picture Dibujar situado en la esquina superior izquierda del menu de opciones
Resultado esperado	Mostrar la pantalla del Dibujar
Resultado obtenido	Pantalla de Dibujar mostrada
Observaciones	Correcto

Figura 8.7 Comprobación de la picture Dibujar

Descripción	*Comprobación de la picture Aprender letras y números*
Acción realizada	Pulsación de la picture Aprender letras y números situado en la esquina superior derecha del menu de opciones
Resultado esperado	Mostrar la pantalla de Aprender letras y números
Resultado obtenido	Pantalla de Aprender letras y números mostrada
Observaciones	Correcto

Figura 8.8 Comprobación de la picture Aprender letras y números

Descripción	*Comprobación de la picture Aprender palabras*
Acción realizada	Pulsación de la picture Aprender palabras situado en la esquina inferior derecha del menu de opciones
Resultado esperado	Mostrar la pantalla de Menu de Familias de palabras
Resultado obtenido	Pantalla de Menu de Familias de palabras mostrada
Observaciones	Correcto

Figura 8.9 Comprobación de la picture Aprender palabras

Descripción	*Comprobación de los botones para Elegir una familia de palabras*
Acción realizada	Pulsación de los botones situados a la izquierda del Menu de Familias de palabras
Resultado esperado	Mostrar una picture de una familia de palabras distinta a las que están.
Resultado obtenido	Picture de una familia de palabras distinta a las que están mostrada
Observaciones	Correcto

Figura 8.10 Comprobación de los botones para Elegir una familia de palabras

Descripción	*Comprobación de elegir aleatoriamente lo que vamos a aprender*
Acción realizada	Pulsación del botón dados situado en las pantalla de aprender palabras o en la pantalla de aprender letras y números
Resultado esperado	Seleccionar una palabra o un letra o número aleatoriamente
Resultado obtenido	Una palabra o una letra o un número seleccionada aleatoriamente
Observaciones	Correcto

Figura 8.11 Comprobación de elegir aleatoriamente lo que vamos a aprender

Descripción	*Comprobación de elegir manualmentelo que vamos a aprender*
Acción realizada	Pulsación de una picture de una palabra situado en las pantalla de aprender palabras o pulsación de un botón del teclado en la pantalla de aprender letras y números.
Resultado esperado	Seleccionar una palabra o una letra o número manualmente
Resultado obtenido	Una palabra una letra o número seleccionada manualmente
Observaciones	Correcto

Figura 8.12 Comprobación de elegir manualmentelo que vamos a aprender

Descripción	*Comprobación del botón Abrir fichero existente*
Acción realizada	Pulsación del botón Abrir fichero existente situado en las pantallas del Reconocedor de escritura o de Dibujar
Resultado esperado	Mostrar un fichero que existe
Resultado obtenido	Un fichero existente mostrado
Observaciones	Correcto

Figura 8.13 Comprobación del botón Abrir fichero existente

Descripción	*Comprobación del botón Abrir un fichero nuevo*
Acción realizada	Pulsación del botón Abrir un fichero nuevo situado en las pantallas del Reconocedor de escritura o de Dibujar
Resultado esperado	Mostrar un fichero nuevo
Resultado obtenido	Un fichero nuevo mostrado
Observaciones	Correcto

Figura 8.14 Comprobación del botón Abrir fichero nuevo

Descripción	*Comprobación del botón Guardar fichero*
Acción realizada	Pulsación del botón Guardar fichero situado en las pantallas del Reconocedor de escritura o de Dibujar
Resultado esperado	Guardar fichero
Resultado obtenido	Fichero guardado
Observaciones	Correcto

Figura 8.15 Comprobación del botón Guardar fichero

Descripción	*Comprobación del botón Imprimir fichero*
Acción realizada	Pulsación del botón Imprimir situado en las pantallas del Reconocedor de escritura o de Dibujar
Resultado esperado	Imprimir fichero
Resultado obtenido	Fichero impreso
Observaciones	Correcto

Figura 8.16 Comprobación del botón Imprimir fichero

Descripción	*Comprobación del botón Vista preliminar*
Acción realizada	Pulsación del botón Vista preliminar situado en las pantallas del Reconocedor de escritura o de Dibujar
Resultado esperado	Mostrar una vista previa al fichero que se va a imprimir
Resultado obtenido	Vista previa al fichero que se va a imprimir mostrada
Observaciones	Correcto

Figura 8.17 Comprobación del botón Vista preliminar

Descripción	*Comprobación del botón Reconocer escritura*
Acción realizada	Pulsación del botón Reconocer escritura situado en las pantalla del Reconocedor de escritura
Resultado esperado	Reconocer la tinta digital
Resultado obtenido	Tinta digital reconocida
Observaciones	Correcto

Figura 8.18 Comprobación del botón Reconocer escritura

Descripción	*Comprobación del botón Escribir tinta*
Acción realizada	Pulsación del botón Lápiz o Pincel situado en las pantalla del Reconocedor de escritura y de Dibujar respectivamente
Resultado esperado	Escribir tinta digital
Resultado obtenido	Tinta digital escribir
Observaciones	Correcto

Figura 8.19 Comprobación del botón Escribir tinta

Descripción	*Comprobación del botón Borrar tinta*
Acción realizada	Pulsación del botón Goma situado en las pantalla del Reconocedor de escritura y de Dibujar
Resultado esperado	Borrar tinta digital
Resultado obtenido	Tinta digital borrada
Observaciones	Correcto

Figura 8.20 Comprobación del botón Borrar tinta

Descripción	*Comprobación del botón Cortar tinta*
Acción realizada	Pulsación del botón Cortar situado en las pantalla del Reconocedor de escritura y de Dibujar
Resultado esperado	Cortar tinta digital
Resultado obtenido	Tinta digital cortada
Observaciones	Correcto

Figura 8.21 Comprobación del botón Cortar tinta

Descripción	*Comprobación del botón Pegar tinta*
Acción realizada	Pulsación del botón Pegar situado en las pantalla del Reconocedor de escritura y de Dibujar
Resultado esperado	Pegar tinta digital
Resultado obtenido	Tinta digital pegada
Observaciones	Correcto

Figura 8.22 Comprobación del botón Pegar tinta

Descripción	*Comprobación del botón Copiar tinta*
Acción realizada	Pulsación del botón Copiar situado en las pantalla del Reconocedor de escritura y de Dibujar
Resultado esperado	Copiar tinta digital
Resultado obtenido	Tinta digital copiada
Observaciones	Correcto

Figura 8.23 Comprobación del botón Copiar tinta

Descripción	*Comprobación del botón Seleccionar tinta*
Acción realizada	Pulsación del botón Seleccionar situado en las pantalla del Reconocedor de escritura y de Dibujar
Resultado esperado	Seleccionar tinta digital
Resultado obtenido	Tinta digital seleccionada
Observaciones	Correcto

Figura 8.24 Comprobación del botón Seleccionar tinta

Descripción	*Comprobación de los botones de Cambiar color de la tinta*
Acción realizada	Pulsación de algún botón Cambiar el color de la tinta situado en las pantalla del Reconocedor de escritura y de Dibujar
Resultado esperado	Cambiar el color de la tinta digital
Resultado obtenido	Color de la tinta digital cambiado
Observaciones	Correcto

Figura 8.25 Comprobación del botón Cambiar color de la tinta

Descripción	*Comprobación de los botones de Cambiar grosor de la tinta*
Acción realizada	Pulsación de algún botón Cambiar el grosor de la tinta situado en las pantalla del Reconocedor de escritura y de Dibujar
Resultado esperado	Cambiar el grosor de la tinta digital
Resultado obtenido	Grosor de la tinta digital cambiado
Observaciones	Correcto

Figura 8.26 Comprobación del botón Cambiar grosor de la tinta

Descripción	*Comprobación del botón de Cambiar color de fondo*
Acción realizada	Pulsación del botón Fondo situado en las pantalla del Reconocedor de escritura y de Dibujar
Resultado esperado	Cambiar el color del fondo
Resultado obtenido	Color de fondo cambiado
Observaciones	Correcto

Figura 8.27 Comprobación del botón Cambiar color de fondo

Descripción	*Comprobación del botón de Limpiar fondo*
Acción realizada	Pulsación del botón Blanco situado en las pantalla del Reconocedor de escritura
Resultado esperado	Limpiar el fondo
Resultado obtenido	Fondo limpiado
Observaciones	Correcto

Figura 8.28 Comprobación del botón Limpiar fondo

Descripción	*Comprobación de los botones de Zoom*
Acción realizada	Pulsación de algún botón de Zoom situado en las pantalla del Reconocedor de escritura y de Dibujar
Resultado esperado	Cambiar el tamaño de la tinta digital
Resultado obtenido	El tamaño de la tinta digital cambiado
Observaciones	Correcto

Figura 8.29 Comprobación de los botones de Zoom

Descripción	*Comprobación de los botones de Usar guías de escritura*
Acción realizada	Pulsación del botón Lineas o Cuadros situados en las pantalla del Reconocedor de escritura
Resultado esperado	Mostrar guías de escritura
Resultado obtenido	Guías de escritura mostradas
Observaciones	Correcto

Figura 8.30 Comprobación de los botones de Usar guías de escritura

Descripción	*Comprobación del botón de Colorear dibujos*
Acción realizada	Pulsación del botón Colorear dibujos situado en las pantalla de Dibujar
Resultado esperado	Mostrar un dibujo para colorear
Resultado obtenido	Dibujo para colorear mostrado
Observaciones	Correcto

Figura 8.31 Comprobación del botón Colorear dibujos

Descripción	*Comprobación del botón Borrar caja de texto*
Acción realizada	Pulsación del botón Borrar texto situado en las pantalla del Reconocedor de escritura
Resultado esperado	Borrar el contenido de la caja de texto
Resultado obtenido	Contenido de la caja de texto borrado
Observaciones	Correcto

Figura 8.32 Comprobación del botón Borrar caja de texto

Parte IV

Manual de Usuario

Capítulo 9

MANUAL DE USUARIO

9.1.Descripción de la aplicación

Este programa le permitirá llevar a cabo las siguientes opciones:

Reconocimiento de escritura manuscrita
- Reconocimiento de texto escrito
- Reconocimiento de letras y dígitos
- Reconocimiento de palabras

Dibujar

9.2. Guía de instalación

A continuación se detalla cómo instalar el software necesario para el funcionamiento de la tinta digital en un PC de sobremesa y la instalación aplicación elaborada para el proyecto.

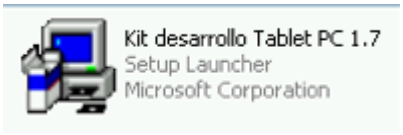
Todo el software necesario para la instalación de la aplicación en un PC de sobremesa se proporciona en el CD-ROM que se adjunta con la memoria, no obstante puede obtenerse también en www.microsoft.com/donwloads para Tablet PC.

Es importante mencionar que el Sistema Operativo sobre el cual deben ser instalados todos los contenidos el **Windows XP Edición Profesional** ya que es el sistema operativo soporte para Tablet PC. Además debe haberse instalado previamente la tableta gráfica que vaya a usarse para el manejo de la aplicación, aunque este paso puede omitirse si se pretende manejar la aplicación usando el ratón que incorpora el PC de sobremesa.

A continuación detallamos los pasos requeridos para la instalación de los requisitos software previos y de la aplicación:

Paso 1: Instalación de Microsoft Tablet PC (SDK) 1.7
Paso 2: Instalación del Módulo de reconocimiento.
Paso 3: Instalación de la aplicación Escribe, yo leo.

9.2.1. Instalación de Microsoft Tablet PC (SDK) 1.7



Seleccionando el icono correspondiente mostrado en la figura, sólo tenemos que seguir los pasos que nos indique el sistema operativo, mediante el programa de instalación.

Figura 9.1 Icono del SDK

El instalador de SDK crea una carpeta en el menú de los programas denominada **Microsoft Tablet PC Platform SDK**. Esta carpeta contiene notas de documentación en **Microsoft Tablet PC Platform SDK Documentation**, así como aplicaciones de muestra y código fuente en **Samples and Source Code.** Los ejecutables binarios compilados para las aplicaciones de muestra están instalados en la carpeta de **SDK\Bin**. Por último se instala una carpeta denominada **Context Tagging Tool** herramienta utilizada para mejorar el reconocimiento de escritura a mano en el panel de entrada, asignando a cada control una lista de frases o expresiones regulares.

9.2.2. Instalación del Módulo de Reconocimiento de Escritura

De igual modo procederemos con el módulo de reconocimiento, seleccionando el icono como el mostrado en la figura y siguiendo los pasos que nos sean indicados en el programa de instalación.

Figura 9.2 Icono del Módulo de Reconocimiento

En primer lugar, el programa de instalación analiza el sistema para averiguar los idiomas de interfaz de usuario instalados para Windows XP. Por cada idioma de interfaz de usuario instalado, la rutina de instalación del módulo de reconocimiento de escritura instala los archivos de recursos de la interfaz de usuario necesarios .

A continuación, el programa instala los reconocedores de escritura a mano y de voz. Durante el proceso de instalación, el usuario puede seleccionar e instalar cualquiera de los motores de reconocimiento disponibles. Se puede instalar un reconocedor aunque el idioma de la interfaz de usuario correspondiente no se haya instalado. Por ejemplo, un usuario puede instalar los reconocedores de escritura a mano y de voz para japonés sin instalar la interfaz de usuario de este idioma.

9.2.3. Instalación de la Aplicación Escribe, yo leo

Vamos a describir de forma detallada la instalación de la aplicación de la que es objeto nuestro proyecto, dentro del subdirectorio Instalación Aplicación que contiene el CD-ROM se encuentran las carpetas y archivos mostrados en la figura, deberemos abrir la carpeta **Escribe, yo leo_Setup.**

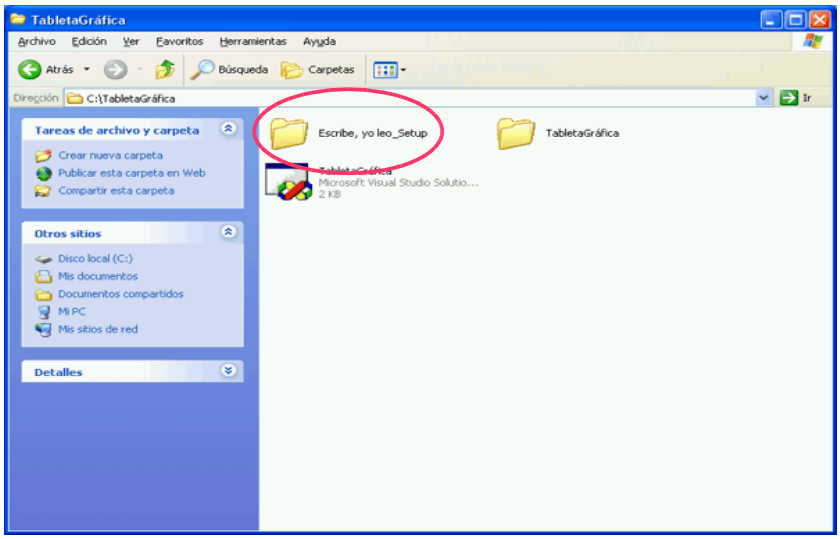


Figura 9.3 Contenido del Subdirectorio Instalación de la Aplicación

Se nos presenta la siguiente pantalla debemos seleccionar y abrir la carpeta **Debug**.

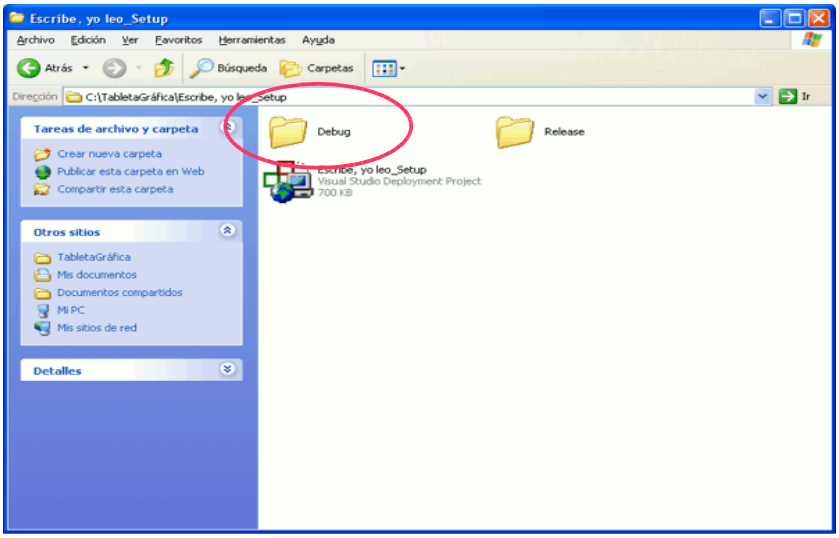


Figura 9.4. Contenido de la carpeta Escribe, yo leo_Setup

Dentro de la carpeta Debug se encuentra el programa de instalación de la aplicación que deberemos seleccionar y ejecutar.

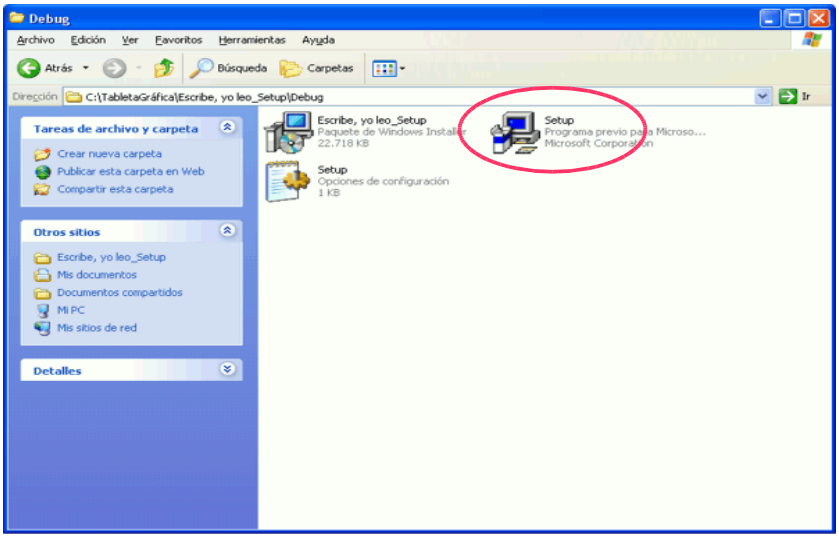


Figura 9.5. Contenido de la carpeta Debug

La ejecución de dicho programa nos presentará las siguientes pantallas:

Es la primera pantalla del programa de instalación, debemos pulsar la tecla "siguiente" para continuar con la instalación de la aplicación.

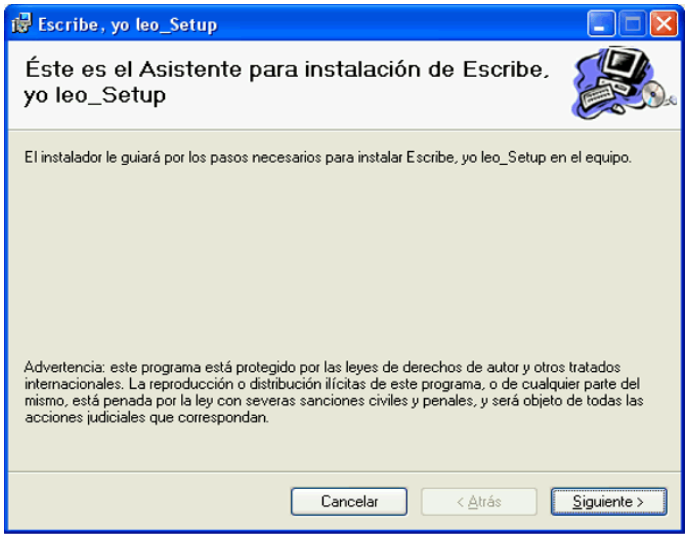
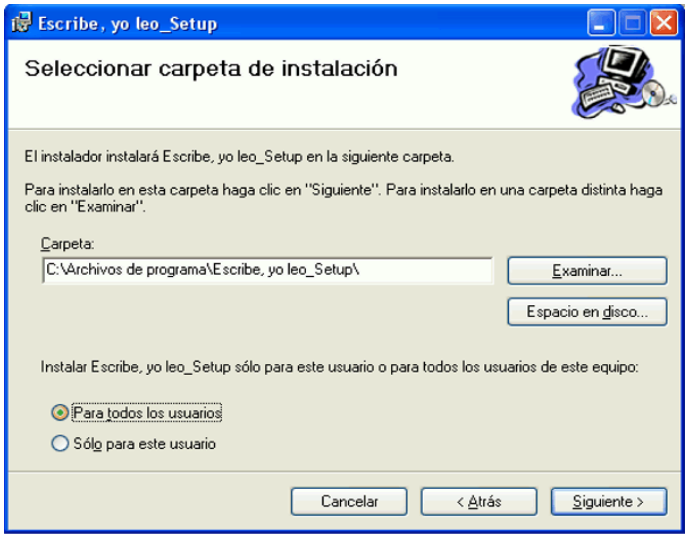
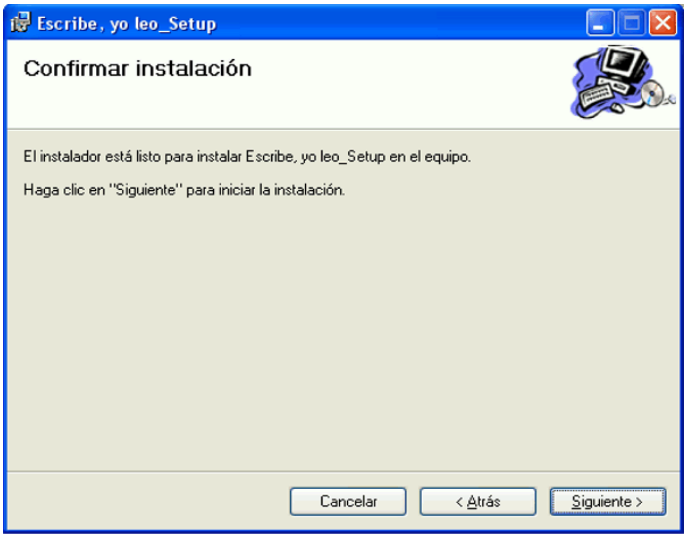


Figura 9.6 Pantalla inicial del instalador



En esta pantalla deberemos seleccionar en que lugar del árbol de directorios de nuestro ordenador queremos que se instale la aplicación y si se realiza la instalación para un único usuario o para todos los usuarios.

Figura 9.7 Pantalla de selección de carpeta de instalación



Si finalmente queremos confirmar la instalación deberemos pulsar “Siguiente”, en caso contrario podemos volver atrás para realizar cambios.

Figura 9.8 Pantalla de confirmación de la instalación.

Deberemos esperar unos segundos mientras se instala el programa en el lugar elegido.

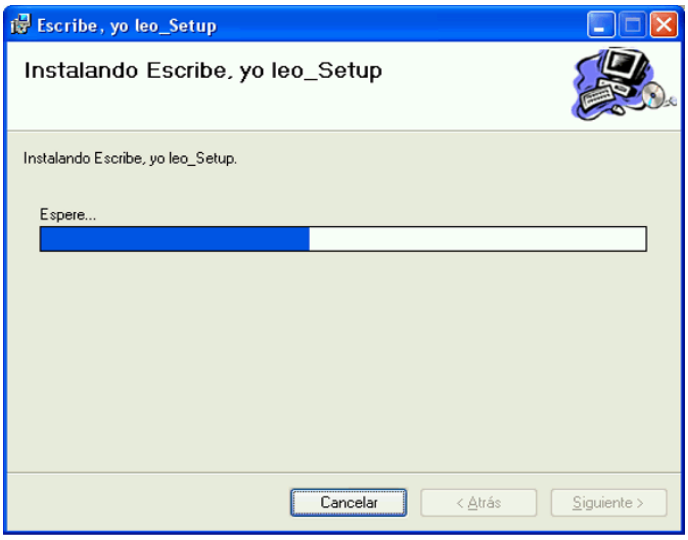
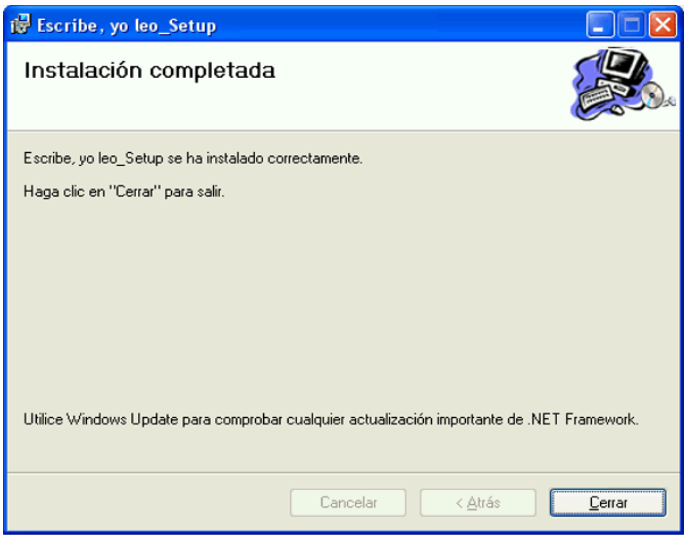


Figura 9.9 Instalando la aplicación.



Una vez finalizada la instalación deberemos cerrar el programa de instalación.

Figura 9.10. Pantalla de finalización

Al finalizar la instalación se muestra en el escritorio el icono de la aplicación lista para ser usada, además también puede accederse a la misma desde **Inicio/Programas/Escribe, yo leo**.

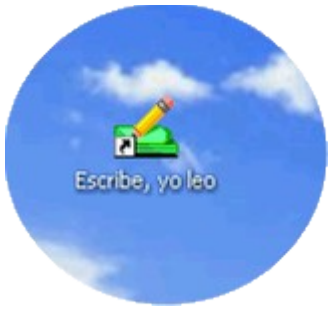


Figura 9.11 Icono de la aplicación en el Escritorio Windows

9.3. Manual de usuario de la aplicación

A continuación se presenta el manual de usuario donde se explica el uso de la aplicación, para un manejo correcto, presentando las sucesivas pantallas, así como los botones incluidos en las mismas y su funcionalidad.

Se han incluido ejemplos gráficos ilustrativos del manejo para facilitar la comprensión y claridad.

9.3.1. Pantalla Inicial

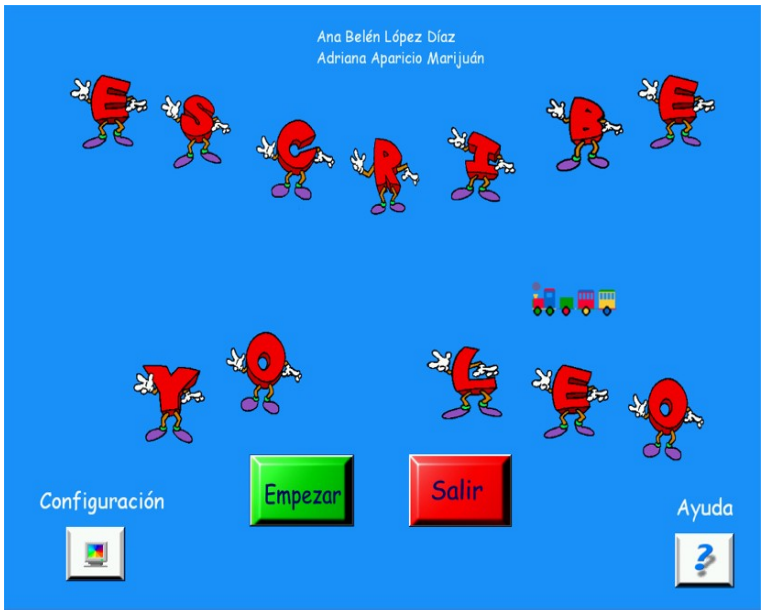


Figura 9.12 Pantalla Inicial

Esta es la pantalla que se muestra al *arrancar la aplicación*, dispone de cuatro botones cuyo uso detallamos a continuación:

Botón Salir
Mediante el uso de este botón se muestra una pantalla de salida con dos botones.

Figura 9.13 Botón Salir

Si pulsamos sobre el botón SI finaliza la aplicación mostrando una pantalla de despedida, pulsando el botón NO permanecemos en la pantalla inicial.

Figura 9.14 Pantalla de Salida

9.3.2. Pantalla Final

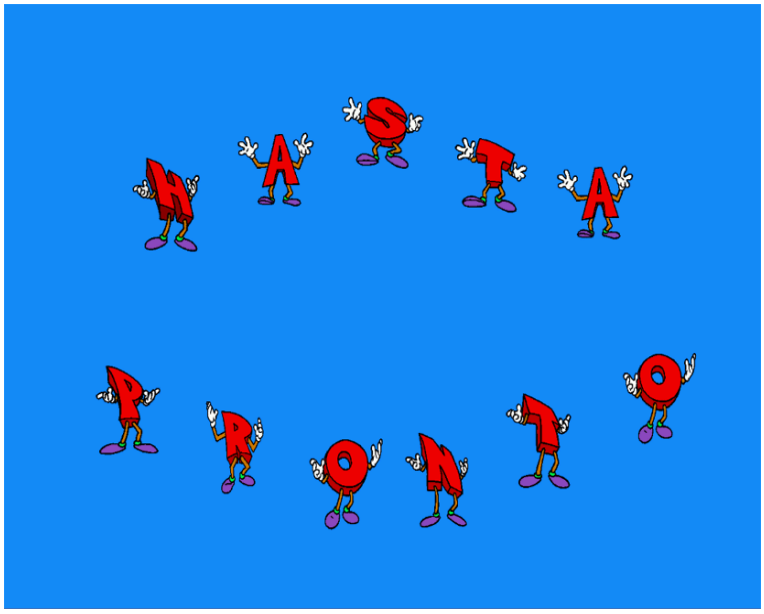


Figura 9.15 Pantalla Final

Esta es la pantalla que se muestra al *finalizar la ejecución de la aplicación* si se elige el botón SI en la pantalla de salida.

Botón Ayuda
Este botón situado a la derecha de la pantalla inicial, nos muestra la ayuda de la pantalla inicial, donde se detalla el uso de cada botón de la misma, el funcionamiento de las pantallas de ayuda se detalla mas adelante.

Figura 9.16 Botón Ayuda

Botón Configuración
Al pulsar sobre este botón situado a la izquierda de la pantalla inicial, se nos presenta la pantalla de configuración de la aplicación.

Figura 9.17 Botón Configuración

9.3.3. Pantalla de Configuración

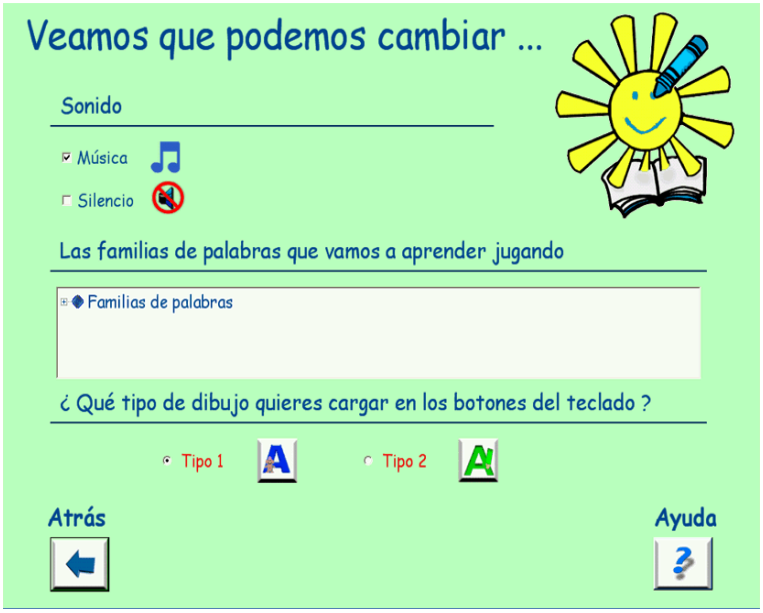


Figura 9.18 Pantalla de Configuración

A continuación se detallan las posibles *opciones de configuración* implementadas para la aplicación.

En la caja de texto se muestras las *familias de palabras* que podemos aprender, así como todos sus datos a modo de consulta, tanto las imágenes como los sonidos mostrados para cada palabra que puede ser reconocida en la parte de reconocimiento de palabras "Aprender palabras".
Para consultar estos datos hay que desplegar el árbol de datos.

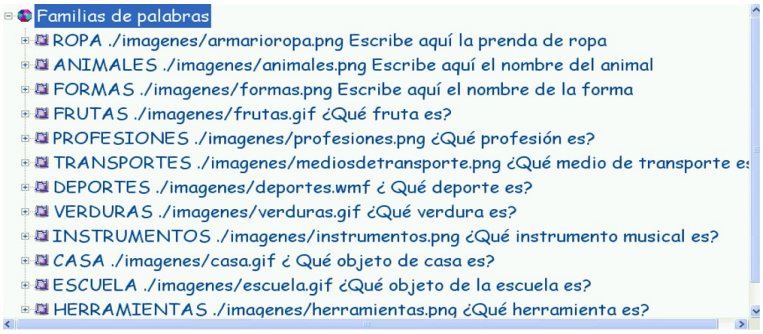


Figura 9.19 Árbol de palabras Familias

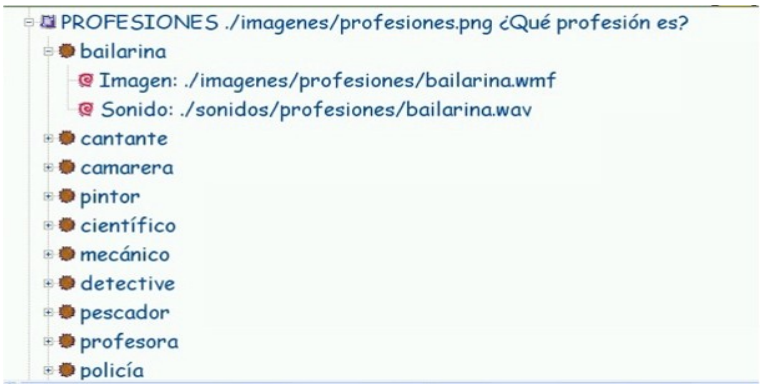


Figura 9.20 Árbol de Palabras Desplegar Familia

Al desplegar se muestra la imagen y el sonido correspondiente a cada palabra dentro de cada grupo de palabras.

Las opciones de configuración posibles son:

Sonido

Música

Podemos *habilitar o no la música* en la aplicación, marcando sobre la casilla correspondiente, que nos muestra el icono de la figura, señalar que sólo quitaremos la música de la aplicación y no otros sonidos de que dispone, al deshabilitar esta opción.

Figura 9.21 Botón Música

Silencio

Si marcamos sobre esta casilla *deshabilitamos por completo los sonidos* de que dispone la aplicación tanto la música como el sonido incorporado a los botones e imágenes, incluidos los posibles mensajes de respuesta que provocan los distintos eventos.

Figura 9.22 Botón Silencio

Botones del teclado

Podemos elegir entre dos tipos de botones para la parte de reconocimiento de letras y números " Aprender letras y números", dependiendo del tipo elegido se cargarán un tipo u otro de imágenes.

Tipo 1

Figura 9.23 Botones de Tipo 1

Tipo 2

Figura 9.24 Botones de Tipo 2

Figura 9.25 Botón Empezar

Botón Empezar
Al pulsar sobre este botón comenzamos el uso de la aplicación, se mostrará una nueva pantalla de menú con cuatro opciones para elegir:

Reconocedor de Escritura
Dibujar
Aprender letras y números
Aprender palabras

9.3.4. Pantalla Menú

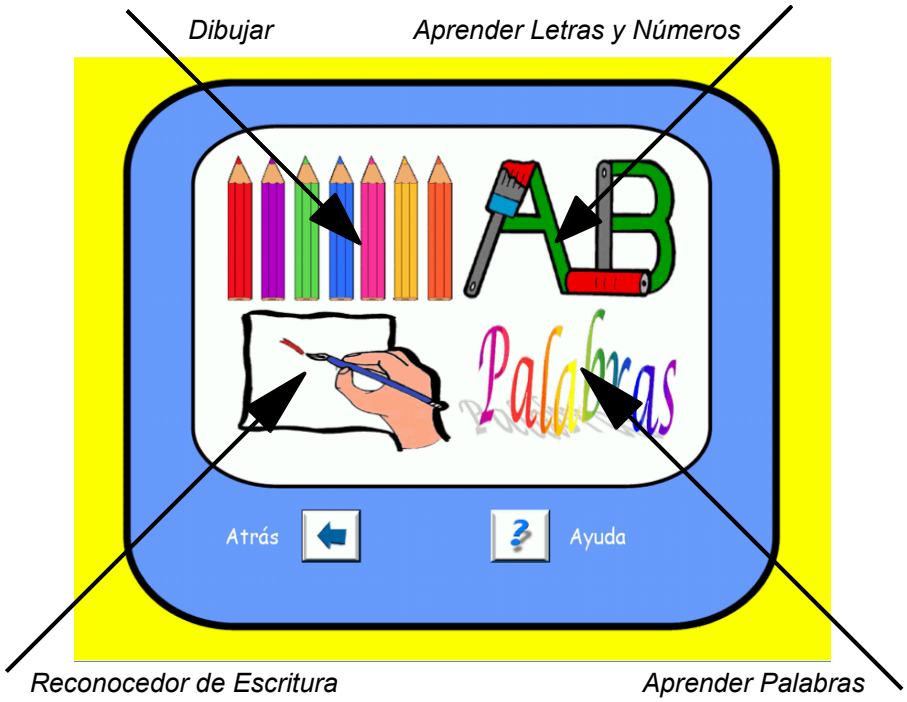


Figura 9.26 Pantalla Menú

En esta pantalla se muestran las cuatro opciones de que dispone la aplicación, para seleccionar cada una de ellas basta pulsar sobre la imagen que las representa. Dispone además de dos botones:

Atrás
Volvemos a la pantalla anterior

Ayuda
Sobre el manejo de la pantalla

9.3.5. Pantalla Reconocedor de Escritura

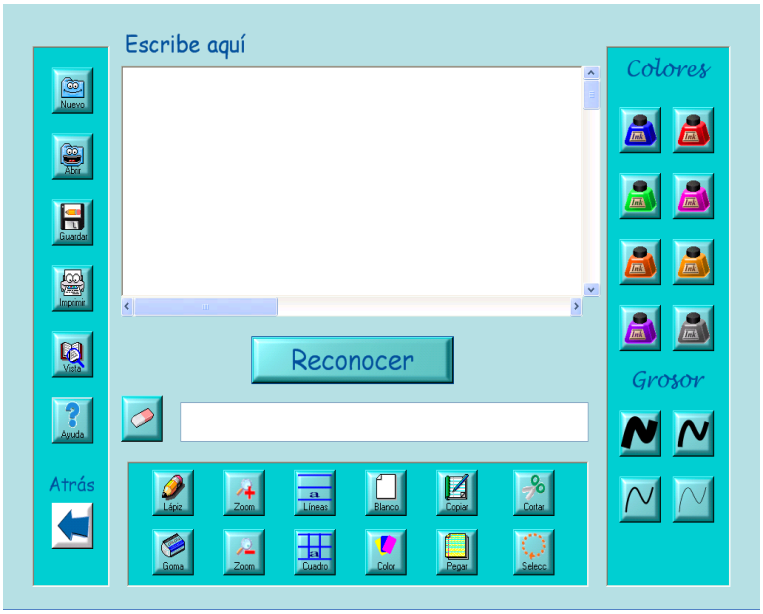
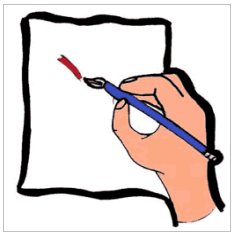


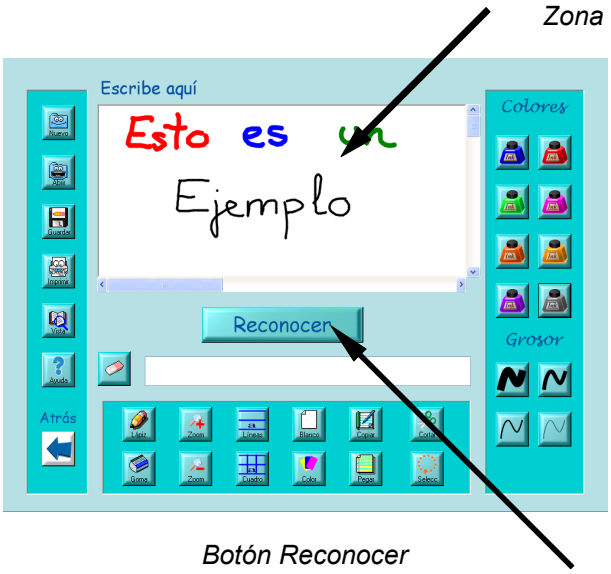
Figura 9.27 Pantalla Reconocedor de Escritura



Esta es la pantalla que se muestra al escoger la *opción Reconocedor de Escritura* en el menú, para ello deberemos pulsar sobre la imagen del menú mostrada en la figura.

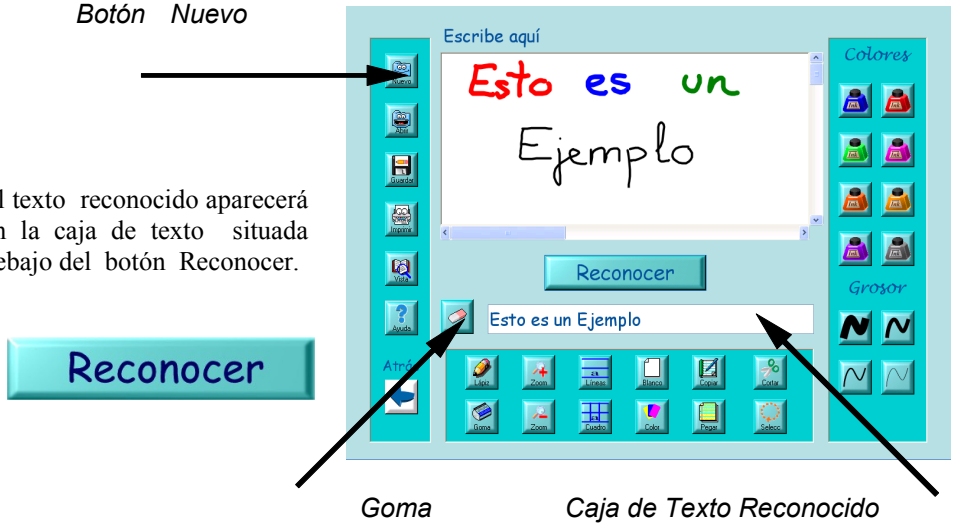
Figura 9.28 Imagen Reconocedor de Escritura

El objetivo fundamental de esta pantalla es el Reconocimiento de Escritura Manuscrita, podemos escribir un texto y la aplicación intentará reconocerlo, a continuación se detalla el funcionamiento de la misma.



El usuario de la aplicación debe escribir sobre la zona indicada, cuando quiera reconocer el texto escrito deberá pulsar sobre el botón Reconocer.

Figura 9.29 Escribir y Reconocer



El texto reconocido aparecerá en la caja de texto situada debajo del botón Reconocer.

Figura 9.30 Borrar Texto y Escritura

Para *borrar* el contenido de la *caja de texto reconocido*, deberá pulsar sobre la goma situada a la izquierda de la caja de texto.

Para limpiar el contenido de la zona de escritura deberá pulsar sobre el botón Nuevo situado en la esquina superior izquierda de la pantalla dentro del panel de Archivo, recuperando el estado inicial de la pantalla para empezar de nuevo.

La pantalla consta de tres paneles como muestra la figura, la funcionalidad de cada uno de ellos se detalla a continuación:

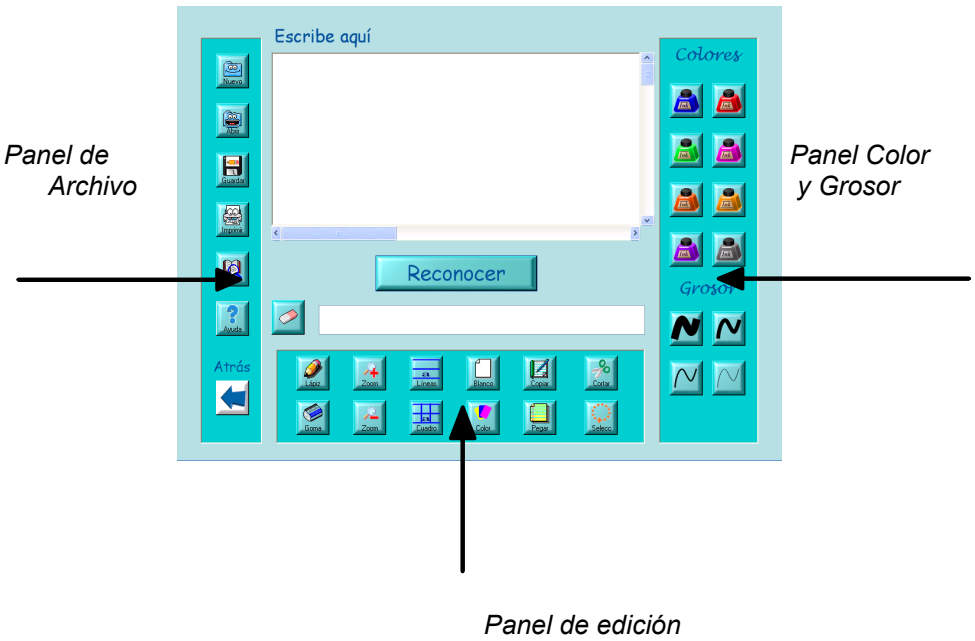


Figura 9.31 Paneles del Reconocedor

9.3.5.1 Panel de Edición

El panel de Edición esta situado en la parte inferior de la pantalla consta de doce botones su funcionamiento es el siguiente:

1. Botón Lápiz

cursor

Pulsando sobre este botón podremos hacer uso del elemento "tinta" de la aplicación observamos que cambia el cursor para convertirse en un *lápiz* esto significa que arrastrando el lápiz sobre la zona sensible de la Tableta Gráfica o bien botón izquierdo del ratón y arrastrando, podremos escribir texto en la zona de la pantalla del Reconocedor de Escritura destinada al efecto.

2. Botón Goma

cursor

Este es el botón destinado a *borrar los trazos de tinta* dentro de la zona de escritura, si nuestro deseo es borrar todo o parte del texto escrito sin que por ello afecte a la labor de reconocimiento. Al hacer uso de este botón podemos observar que el cursor cambia para convertirse en una *goma,* debemos indicar nuestro deseo de reanudar la escritura en su caso pulsando el botón lápiz.

3. Botón Zoom Más

Como su nombre indica cada vez que pulsamos este botón *aumentamos el tamaño del texto* que se encuentra dentro de la zona de escritura, puede ocurrir que deje de visualizarse en pantalla, la pantalla dispone de dos barras de desplazamiento en sentido vertical y horizontal para poder localizar el texto en el área de escritura.

4. Botón Zoom Menos

Este botón realiza la función inversa del botón anterior, *disminuye el tamaño del texto* hasta su tamaño original.

5. Botón Líneas

Para facilitar la escritura se han incluido dos tipos de guías, el botón que nos ocupa dibuja *líneas horizontales* similares a las líneas en un cuaderno real de escritura, se ha considerado incluir guías debido a su utilidad en los primeros pasos aprendiendo a escribir.

6. Botón Cuadros

Este botón muestra cuadros como guía, simulando hojas de *papel cuadriculado*, en cuanto a su funcionamiento es análogo al botón Líneas.

7. Botón Fondo Blanco

Pulsando este botón cambiaremos el color del área de escritura a blanco, si previamente habíamos puesto otro color de fondo, igualmente eliminaremos las líneas o cuadros de guía si previamente habíamos elegido esta opción.

8. Botón Fondo de Color

Al pulsar el botón Fondo se abre un cuadro de diálogo como el mostrado en la figura de modo que el usuario puede elegir en la paleta de colores.

El usuario podrá elegir entre los colores básicos mostrados en este Cuadro de Diálogo Standard de Windows el color de fondo que quiere que se muestre en el área de escritura, señalar que el color de fondo puede utilizarse aunque se pulsen los botones de líneas o cuadros como guía.

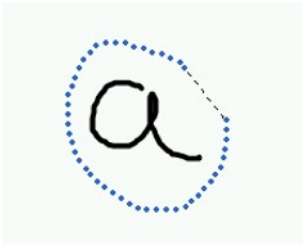
Por defecto el color de fondo de la zona de escritura es el blanco, color de fondo que se muestra al pulsar sobre el botón nuevo.

Figura 9.32 Diálogo de Color de Fondo

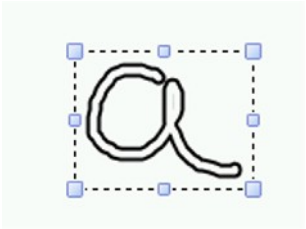
9. Botón Seleccionar

cursor

La misión de este botón es pasar a modo selección de "tinta", es decir rodeando la zona deseada con el lápiz de la Tableta Gráfica o pulsando el botón izquierdo del ratón y arrastrando seleccionamos texto escrito para copiar, cortar o pegar. Para volver a escribir deberemos pulsar de nuevo el botón Lápiz.



Observaremos que el cursor cambia para convertirse en una *flecha* mostrando una curva de puntos alrededor de la zona seleccionada mientras se realiza la selección.



Al soltar el botón izquierdo del ratón o levantar el lápiz de la tableta la zona seleccionada queda rodeada por un *marco*, indicando que dicha zona ha sido seleccionada para cortar, copiar o pegar, el color de la tinta se convierte en transparente.

Figura 9.33 Seleccionar y Marco

10. Botón Copiar

Una vez hemos realizado la selección podemos pulsar este botón para *copiar la selección* en el porta papeles y posteriormente cortar o pegar.

11. Botón Pegar

También podremos *pegar la selección* es decir añadir al texto que tenemos visualizado el texto que previamente hemos seleccionado.

12. Botón Cortar

Pulsando sobre las tijeras *cortamos la selección*, es decir la suprimimos aunque posteriormente podemos copiar o pegar.

13 Ejemplo Gráfico.

Veamos el funcionamiento mediante un **ejemplo gráfico**.

Una vez pulsado el botón Seleccionar, rodeamos la zona a seleccionar con el lápiz de la tableta gráfica o pulsando el botón izquierdo del ratón, se muestra una línea de puntos.

Aparece el texto seleccionado dentro de un marco indicando la selección realizada. Pulsamos sobre el botón Copiar, la selección se incluye en el porta papeles.

Figura 9.34 Seleccionar Texto

Pulsamos sobre el botón Pegar, vemos que la selección se desplaza a la esquina superior izquierda de la zona de escritura.

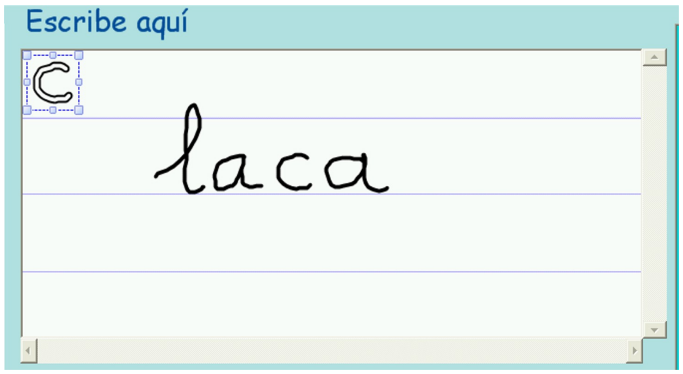


Figura 9.35 Desplazamiento de la Selección

Podemos entonces arrastrar, colocar y pegar en cualquier lugar de la zona de escritura y tantas veces como deseemos.

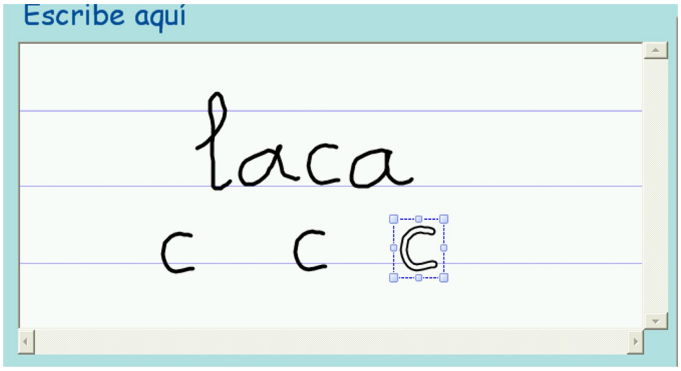


Figura 9.36 Copiar Texto

Si pulsamos en el botón Cortar desaparece el marco de selección y por tanto el texto seleccionado.

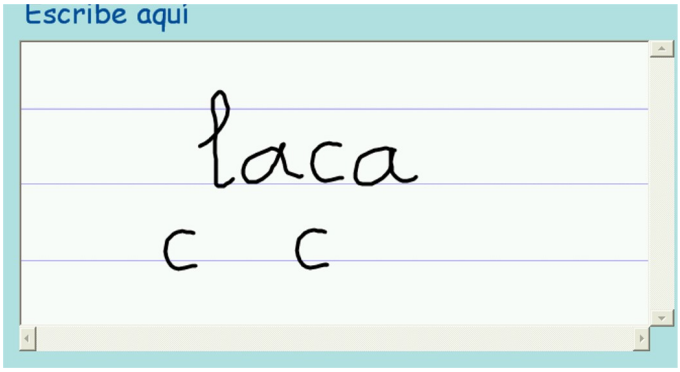


Figura 9.37 Cortar Texto

Si no disponemos de texto en el porta papeles para copiar, pegar o cortar el programa nos mostrará un mensaje de aviso como los siguientes:

No disponemos de texto en el porta papeles para COPIAR o aún no hemos realizado la selección.

Figura 9. 38 Mensaje No hay Tinta para Copiar

No disponemos de texto en el porta papeles para CORTAR o aún no hemos realizado la selección.

Figura 9.39 Mensaje No hay Tinta para Cortar

No disponemos de texto en el porta papeles para PEGAR o aún no hemos realizado la selección.

Figura 9.40 Mensaje No hay Tinta para Pegar

La selección realizada se guardará en el porta papeles lista para pegar o copiar hasta que realicemos una nueva selección.

9.3.5.2. Panel Color y Grosor

Este panel se encuentra situado en la parte derecha de la pantalla, dispone de doce botones agrupados en dos grupos, color y grosor.

1. Color

Disponemos de ocho colores de tinta para elegir que se muestran a continuación, solamente debemos pulsar sobre el botón que contiene el recipiente de tinta seleccionado y automáticamente la tinta adquiere el color deseado.

Tinta Roja

Tinta Azul

Tinta Naranja

Tinta Rosa

Tinta Amarilla

Tinta Verde

Tinta Negra

Tinta Morada

2. Grosor

En cuanto al grosor tenemos cuatro grosores de lápiz, de forma análoga el grosor de la tinta varia al pulsar el botón de grosor deseado, expresados en mm tenemos:


Fino 0.1 mm


Grueso 2 mm


Medio 1mm


Muy Grueso 3 mm

9.3.5.3 Panel Archivo

Este panel se sitúa en la parte izquierda de la pantalla comprende operaciones que podemos realizar con archivos en seis botones y el botón Atrás para volver a la pantalla anterior.

1. Botón Nuevo

Este botón nos sirve para *limpiar el área de escritura*, esto es si queremos recuperar el estado inicial de la pantalla debemos pulsar este botón.

2. Botón Abrir

Al pulsar sobre este botón se muestra un cuadro de diálogo Standard de Windows para abrir archivos, previamente debemos haber guardado un archivo para poderlo recuperar después mediante el uso de Abrir.

Este es un ejemplo de cuadro de diálogo mostrado por la aplicación aunque su formato puede variar dependiendo de donde tengamos guardados nuestros ficheros dentro del equipo.

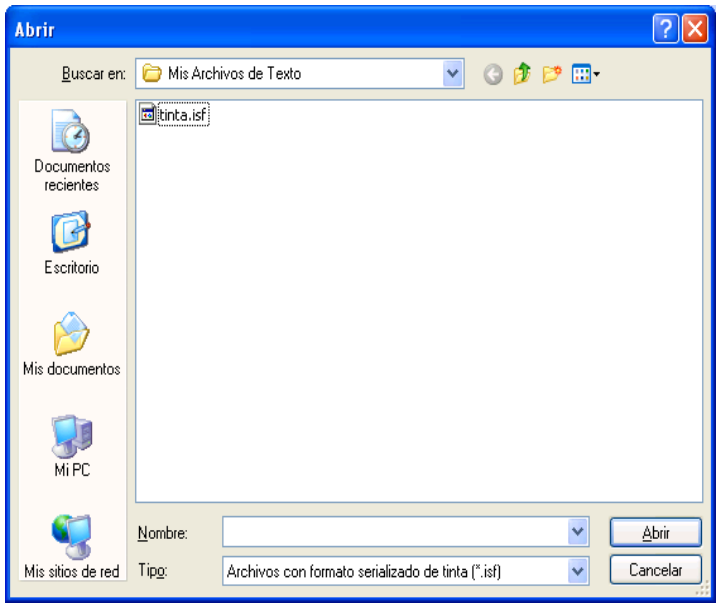


Figura 9.41 Diálogo Abrir Archivo

3. Botón Guardar

Este es el botón que debemos utilizar para *guardar un archivo* que hayamos creado con la aplicación, los archivos se guardan en un formato es denominado formato Serializado de Tinta cuya extensión por defecto es ISF.

El nombre de archivo se deja a la elección del usuario debiendo rellenar la caja de texto del cuadro de diálogo y a continuación pulsar abrir, el texto almacenado aparecerá en el área de escritura de la pantalla.

En la figura se muestra un cuadro similar al que puede aparecer en la aplicación.

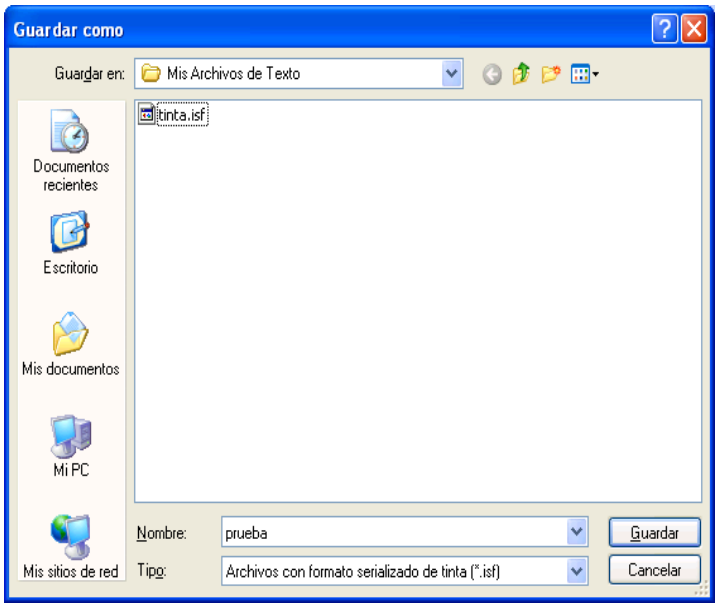


Figura 9.42 Guardar Archivo

Si intentamos guardar pero aún no hemos escrito nada en el área de escritura se muestra un mensaje de aviso como el mostrado a continuación.

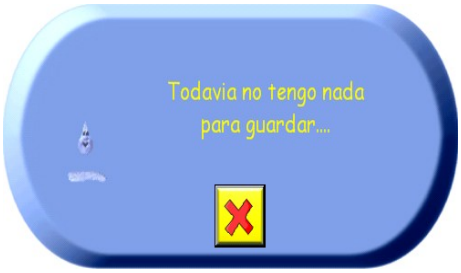
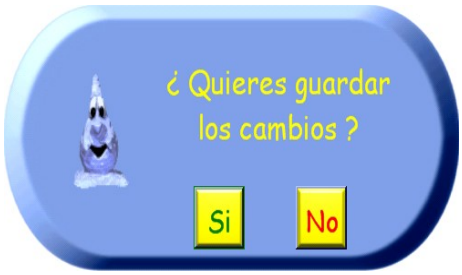


Figura 9.43 Guardar Archivo vacío

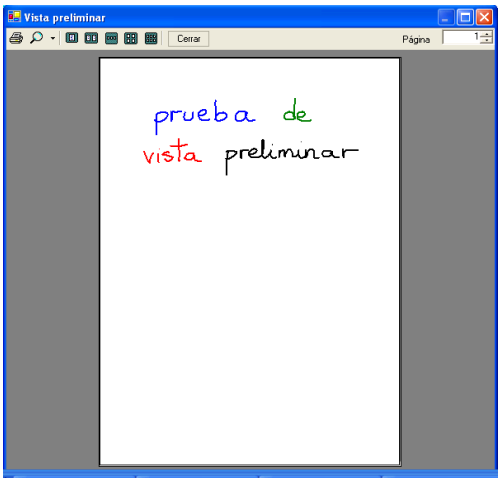


Si tenemos texto escrito y pulsamos en el botón Atrás nos preguntará si deseamos guardar el trabajo realizado mostrando el mensaje de aviso mostrado en la figura. Si se elige la opción SI se abrirá el Cuadro de Diálogo de Guardar Archivo, en caso contrario se presenta la pantalla de Menú.

Figura 9.44 Guardar cambios

4. Botón Vista Preliminar

La función de este botón es presentar un cuadro de Diálogo Standard de Windows de Vista Preliminar como el mostrado en la figura, muestra *cómo se plasmara en papel el texto* que hemos escrito en el área de escritura y que posteriormente imprimiremos.



Ejemplo del Cuadro de Diálogo mostrado cuando pulsamos sobre el botón Vista preliminar.

Figura 9.45 Diálogo Vista Preliminar

5. Botón Imprimir

Pulsando sobre este botón *enviamos a la impresora* el documento mostrado en la Vista Previa. Se mostrará el siguiente mensaje de aviso que nos indica que el documento ha sido enviado al dispositivo de impresión.

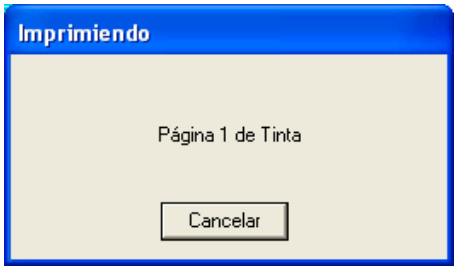


Figura 9.46 Mensaje de Impresión

6. Botón Ayuda

Cuando se pulsa este botón se *muestra la ayuda* sobre la Pantalla del Reconocedor de escritura, con una breve explicación de la función de cada botón. El manejo de las pantallas de ayuda se detalla mas adelante.

7. Botón Atrás

Este botón nos lleva a la *pantalla anterior*, en este caso a la Pantalla Menú.

9.3.6. Pantalla de Dibujo

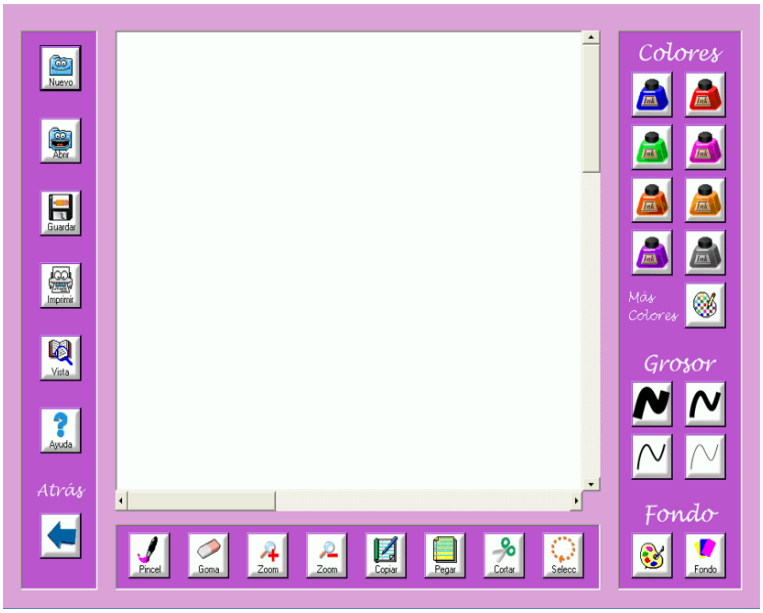


Figura 9.47 Pantalla de Dibujo

Como se puede ver, esta pantalla es similar a la Pantalla del Reconocedor de Escritura, la finalidad de la misma es emplear "tinta" para dibujar, el usuario de la aplicación puede crear en el área de pintura el dibujo que desee.

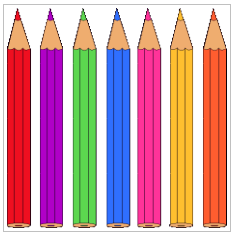


Figura 9.48 Dibujar

Esta es la pantalla que se presenta al elegir la *opción Dibujar en la pantalla Menú*, para ello debemos pulsar sobre la imagen mostrada en la figura

Se distribuye igual que la pantalla del Reconocedor de Escritura en tres paneles de botones.

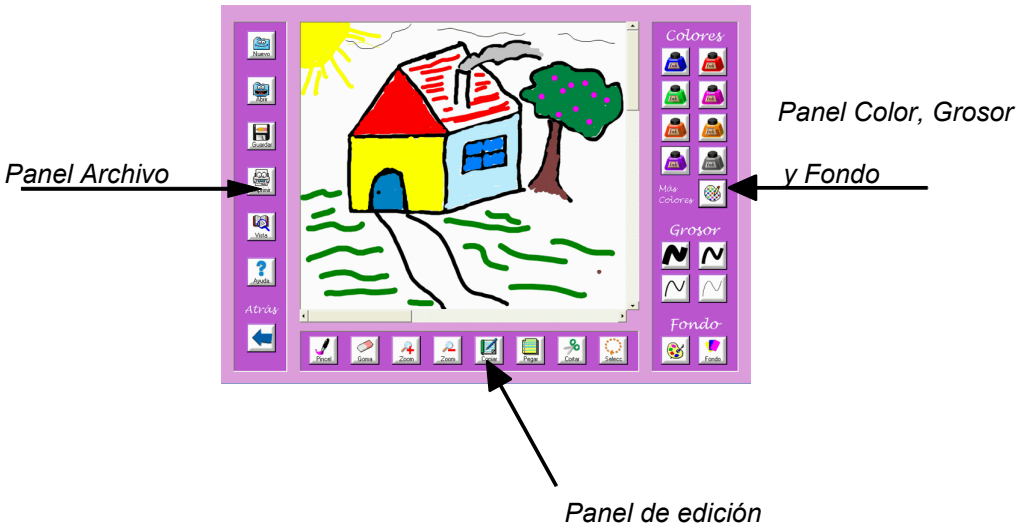


Figura 9.49 Paneles de la Pantalla de Dibujo

Debido a la similitud existente entre la Pantalla del Reconocedor de Escritura y la Pantalla de Dibujo, evitaremos la explicación de aquellos botones cuya funcionalidad es la misma en ambas pantallas.

En el Panel de edición disponemos ahora de dos iconos nuevos aunque conservan su funcionalidad.

1. Botón Pincel

cursor

Cuando pulsamos sobre el botón pincel podemos *dibujar en el área*. El cursor cambia para convertirse en un pincel indicando que podemos pintar.

2. Botón Goma

cursor

Al pulsar sobre la Goma podremos *borrar los trazos* que deseemos.

Disponemos ahora de dos botones nuevos en el Panel Colores, Grosor y Fondo son los siguientes:

3. Botón Colorear

Cuando pulsamos sobre el botón Colorear se muestra una lista de posibles *dibujos* que podemos elegir *para colorear*.

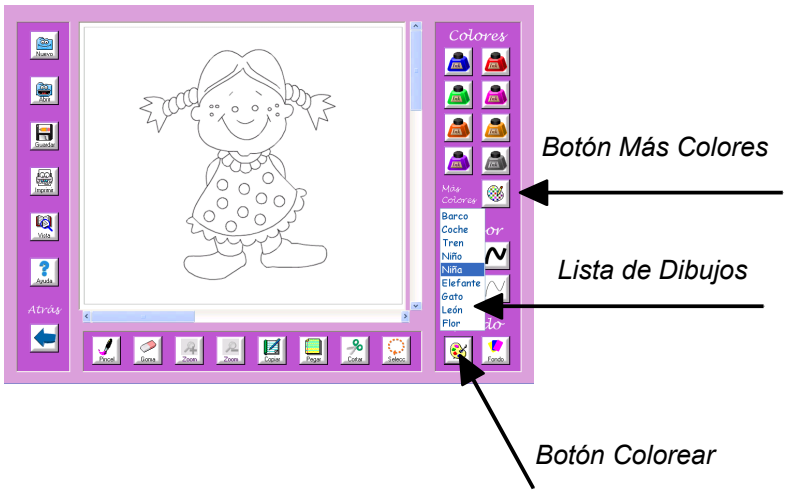


Figura 9.50 Botones Colorear y Más colores

En la figura se muestra al detalle la *lista de elección de dibujos para colorear*, haciendo doble click sobre el dibujo elegido se muestra en pantalla. Podemos ahora utilizar tinta de colores y grosor para retocar el dibujo como deseemos.

Figura 9.51 Lista de Dibujos

4. Botón Más Colores

En la pantalla disponemos de los colores básicos en los botones de tintas de colores, además podremos elegir *otros colores* si pulsamos este botón, se presenta un cuadro de Diálogo Standard de Windows de Color Podemos definir *Colores personalizados* pulsando en el botón destinado al efecto en el cuadro de diálogo, como se muestra en la figura.

Figura 9.52 Cuadro de Diálogo de Más Colores

En el ejemplo mostrado en la figura vemos la elección de un color del cuadro de diálogo, como se puede apreciar también podemos cambiar el color de fondo del dibujo utilizando el botón fondo.

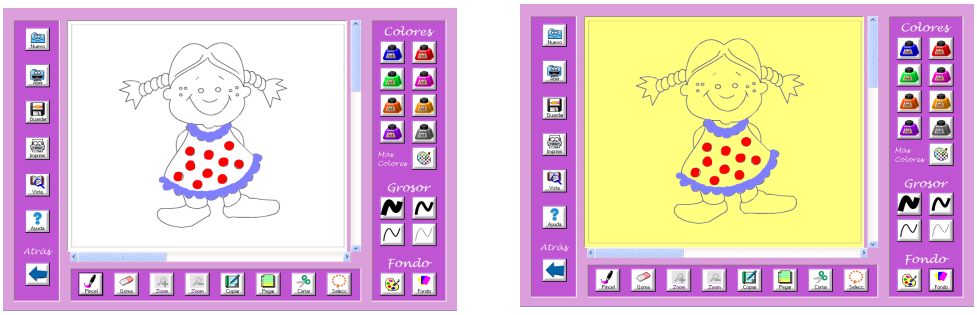
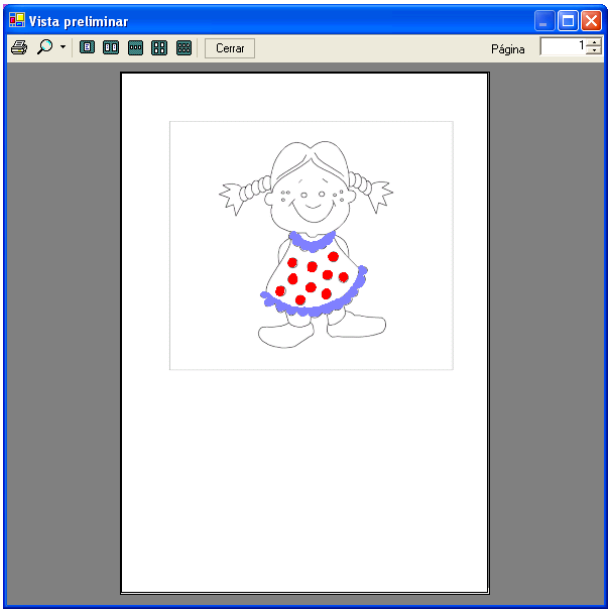


Figura 9.53 Dibujo coloreado y dibujo con fondo



Podemos imprimir el dibujo, si lo deseamos, como se muestra en la figura de la izquierda, veremos el resultado pulsando en el botón Vista Preliminar

Figura 9.54 Vista Preliminar de Dibujo

Con respecto al resto de los botones de que dispone la pantalla de Dibujo su funcionalidad es la misma que los de la Pantalla Reconocedor de Escritura, si es necesario realizar alguna consulta sobre ellos puede hacerse en esta parte del manual de usuario.

9.3.7. Pantalla Aprender Letras y Números

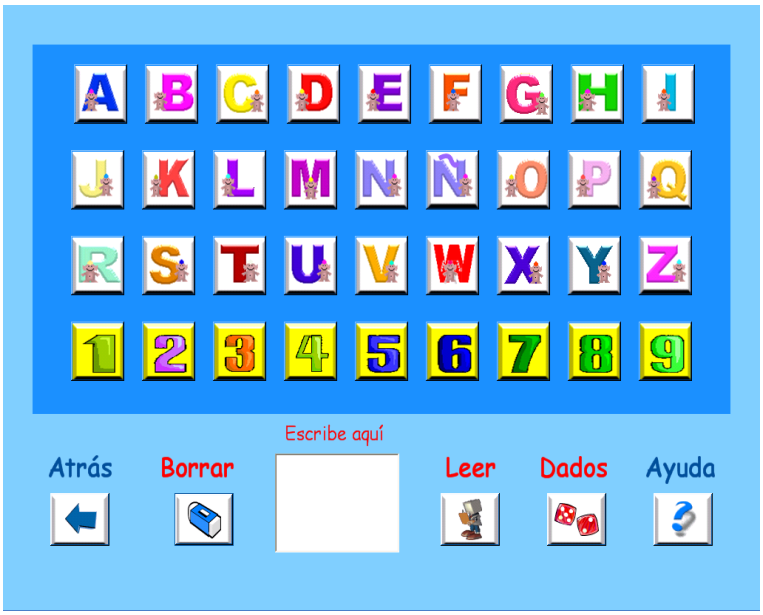


Figura 9.55 Pantalla Aprender Letras y Números botones *Tipo 1*

Figura 9.56 Pantalla Aprender Letras y Números botones *Tipo 2*

Estas son las dos opciones posibles de la pantalla Letras y Números, se mostrará una u otra dependiendo de la opción elegida para los botones en la pantalla de Configuración.

La opción de la pantalla Menú que nos lleva a esta pantalla es la representada por la figura.

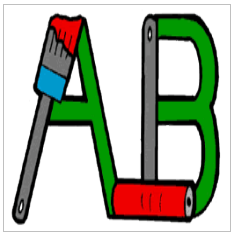


Figura 9.57 Opción Letras y Números

El objetivo de esta parte de la aplicación es el Reconocimiento de letras y números, la pantalla dispone de cinco botones que explicamos a continuación.

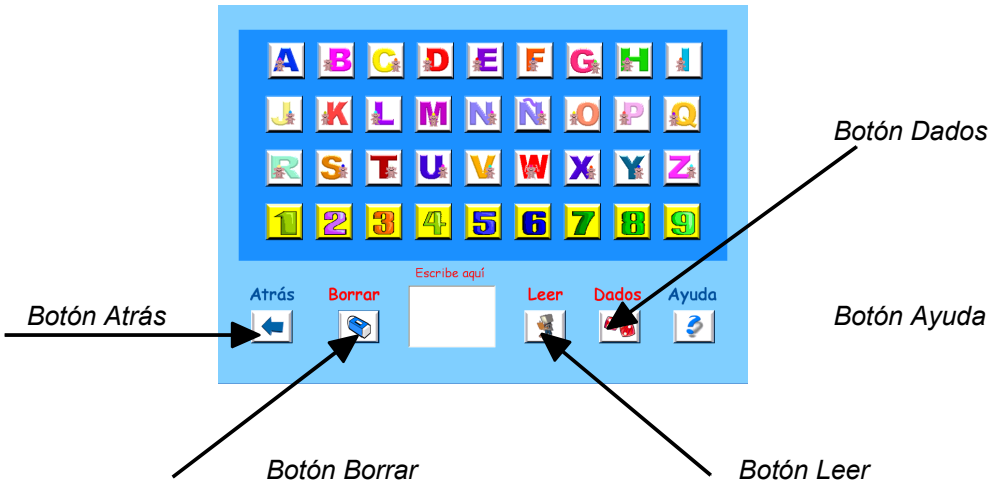


Figura 9.58 Botones Pantalla Teclado

1. Botón Atrás

Pulsando este botón volvemos a la *pantalla anterior*, la Pantalla de Menú

2. Botón Goma

Si nos hemos equivocado al escribir y queremos escribir de nuevo, debemos pulsar este este botón, *borra las pinceladas de tinta* y deja el área de escritura lista para escribir de nuevo.

Esta es la *zona de escritura* donde podemos escribir la letra o número. Cuando el lápiz de la tableta gráfica o el ratón entran en la zona de escritura el cursor cambia, aparece como un lápiz indicando que podemos escribir.

cursor 

Figura 9.59 Zona de Escritura

3. Botón Leer

Pulsando este botón la aplicación intentará *reconocer la letra o número* escrito en la zona de escritura.

4. Botón Dados

Pulsando este botón la letra o número son elegidos aleatoriamente por la aplicación.

5. Botón Ayuda

Este botón nos muestra la ayuda para esta pantalla, las pantallas de ayuda se detallan mas adelante.

6. Ejemplo

Veamos un **ejemplo ilustrativo** del manejo de esta pantalla.

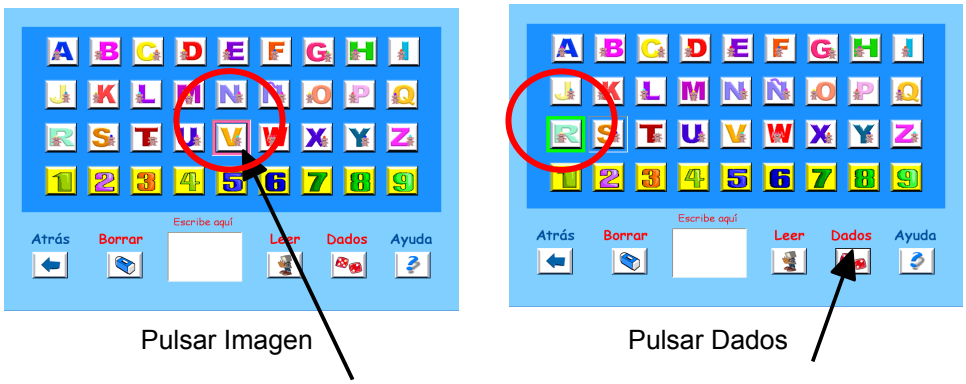


Figura 9.60 Pulsar botón de letra o número, o pulsar dados para elegir

El usuario puede elegir directamente la letra o número que desea escribir *pulsando sobre el botón* correspondiente como se muestra en la figura de la izquierda o *pulsando sobre el botón dados*, de esta forma la letra o número se eligen de forma aleatoria como se muestra en la figura de la derecha, los botones se van iluminando hasta llegar al botón seleccionado por la aplicación. Si en las opciones de configuración no hemos quitado el sonido de la aplicación, escucharemos la letra o número que debemos escribir en la zona de escritura.

Si no elegimos directamente letra o número ni tampoco pulsamos el botón Dados, cuando tratamos de leer no dispondremos de letra o número la aplicación mostrará un mensaje de aviso como el de la figura.

Si hemos elegido letra o número y pulsamos el botón Leer antes de escribir la letra o número correspondiente en la zona de escritura también se mostrará un mensaje de aviso.

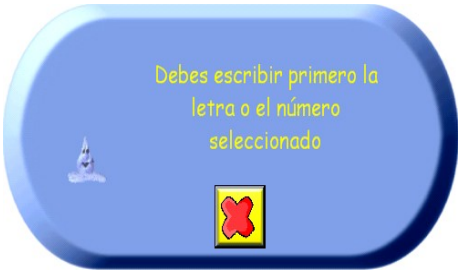
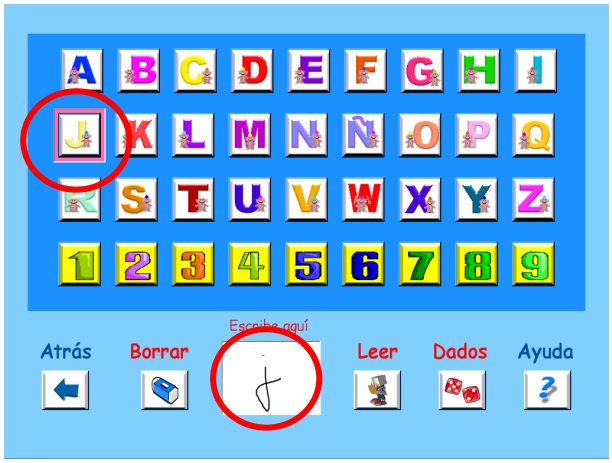


Figura 9.61 Mensajes de aviso



Si ya tenemos seleccionada la letra o el número por uno de los dos métodos vistos, podemos escribir en la zona de escritura y pulsar el botón leer. Se mostrarán dos posibles mensajes como los siguientes.

Figura 9.62 Escribir en la zona de escritura

Si la escritura es reconocida y corresponde por tanto con la letra o número elegido, la aplicación muestra un mensaje indicando que la escritura ha sido realizada correctamente.

Figura 9.63 Mensaje Afirmativo

Si nos equivocamos al escribir y la aplicación no reconoce la letra o número nos mostrará un mensaje indicando que la escritura no ha sido correcta, invitándonos a intentarlo de nuevo

Figura 9.64 Mensaje Negativo

9.3.8. Pantalla Menú de Palabras

Figura 9.65 Pantalla Menú Palabras

Esta es la pantalla que se mostrará al elegir en la pantalla de menú la opción Aprender Palabras que representa la figura:

Figura 9.66 Opción Aprender Palabras del Menú

El objetivo de esta parte de la aplicación es el *Reconocimiento de Palabras*, para ello se dispone de doce familias de palabras, cada familia consta de quince palabras, el contenido de cada familia puede consultarse en la Pantalla de Configuración desplegando el árbol de palabras.

En la Pantalla Menú de Palabras se muestran cuatro opciones para elegir, para cambiar las familias presentadas en el Menú basta con pulsar en los botones situados a la izquierda de la pantalla, con lo que cambia la configuración de las familias presentadas.

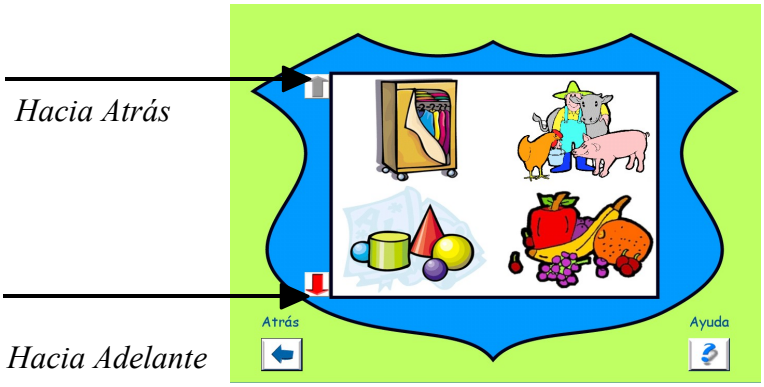


Figura 9.67 Botones de cambio de Familias de Palabras

1. Botón Hacia Atrás

Al pulsar sobre este botón, nos movemos *hacia atrás* en la lista de familias representadas por las imágenes en la pantalla Menú Palabras hasta la primera imagen, tal y como se ve en la figura anterior, el botón se deshabilita indicando que hemos llegado al principio de la lista.

2. Botón Hacia Adelante

Pulsando sucesivamente este botón nos movemos *hacia adelante* en la lista de familias, las imágenes van rotando a medida que se incorporan al menú presentándose en todo momento cuatro opciones para elegir.

La pantalla consta además de dos botones ya conocidos:

3. Botón Atrás

Volvemos a la pantalla anterior, la Pantalla de Menú en este caso.

4. Botón Ayuda

Sobre el manejo de la pantalla, detallada más adelante.

Veamos su funcionamiento mediante un **ejemplo:**

Pulsar Botón Hacia Adelante 1ª Vez *Pulsar Botón Hacia Adelante 2ª Vez*

Pulsar Botón Hacia Adelante 3ª Vez *Pulsar Botón Hacia Adelante 4ª Vez*

Figura 9.68 Cambio de imágenes en Pantalla Menú Palabras

Como vemos en la figura anterior pulsando sucesivamente sobre el botón Hacia Adelante una nueva imagen se incorpora en la esquina inferior derecha, las imágenes van rotando de forma que la imagen que ocupa la esquina superior izquierda desaparece al incorporarse una nueva imagen, esta operación se repite hasta presentar las doce familias de palabras.

Para volver hacia atrás debemos pulsar el botón Hacia Atrás sucesivamente hasta la pantalla inicial.

Para elegir una opción de las cuatro presentadas basta con seleccionar utilizando el lápiz de la tableta gráfica o haciendo click con el botón izquierdo del ratón en la imagen elegida.

Las familias de palabras representadas mediante imágenes son:

Prendas de vestir

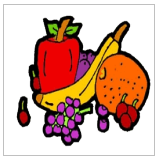
Deportes

Animales

Herramientas

Formas y Colores

Verduras



Frutas

Profesiones

Objetos de Casa

Instrumentos Musicales

Material de la Escuela

Medios de Transporte

9.3.9. Pantallas Aprender Palabras

A continuación se presentan las doce pantallas de que consta esta parte de la aplicación, se accede a cada una de ellas pulsando la imagen correspondiente en la Pantalla Menú de Palabras.

Pulsando sobre la imagen mostrada se accede a la Pantalla *Prendas de Ropa*

Figura 9.69 Pantalla Prendas de Ropa

Pulsando sobre la imagen mostrada se accede a la Pantalla *Formas y Colores*

Figura 9.70 Pantalla Formas y Colores

Pulsando sobre la imagen mostrada se accede a la Pantalla *Frutas*

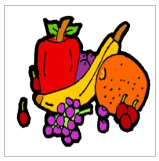


Figura 9.71 Pantalla Frutas

Pulsando sobre la imagen mostrada se accede a la Pantalla *Animales*

Figura 9. 72 Pantalla Animales

Pulsando sobre la imagen mostrada se accede a la Pantalla de *Profesiones*

Figura 9.73 Pantalla Profesiones

Pulsando sobre la imagen mostrada se accede a la Pantalla *Medios de Transporte*

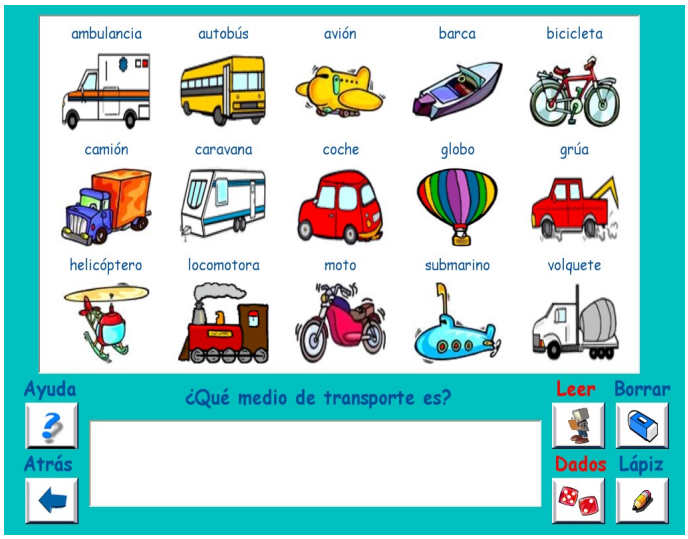


Figura 9.74 Pantalla Medios de Transporte

Pulsando sobre la imagen mostrada se accede a la Pantalla *Deportes*

Figura 9.75 Pantalla Deportes

Pulsando sobre la imagen mostrada se accede a la Pantalla *Verduras*

Figura 9.76 Pantalla Verduras

Pulsando sobre la imagen mostrada se accede a la Pantalla *Instrumentos Musicales*

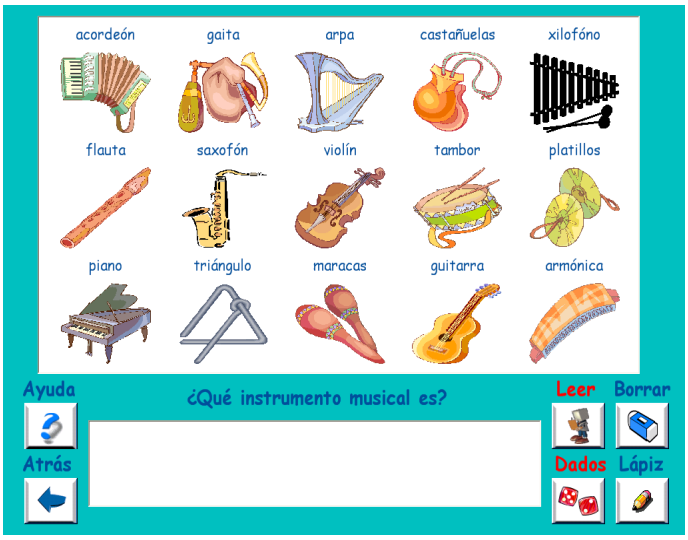


Figura 9.77 Pantalla Instrumentos Musicales

Pulsando sobre la imagen mostrada se accede a la Pantalla *Objetos de la Casa*

Figura 9.78 Pantalla Objetos que hay en la casa

Pulsando sobre la imagen mostrada se accede a la Pantalla *Material de la Escuela*

Figura 9.79 Pantalla Material de la Escuela

Pulsando sobre la imagen mostrada se accede a la Pantalla *Herramientas*

Figura 9.80 Pantalla Herramientas

El objetivo de estas doce pantallas es *Reconocer palabras escritas*, el usuario debe reproducir la palabra en la *zona de escritura* de que dispone la pantalla, cada imagen tiene asociada la palabra correspondiente, podemos por tanto ver como se escribe y si tenemos habilitado el sonido escucharla.

La pantalla dispone de seis botones, los cuatro primeros ya conocidos puesto que su funcionamiento es similar al de la Pantalla Aprender Letras y Números.

1. Botón Atrás

Pulsando este botón volvemos a la *pantalla anterior*, la Pantalla Menú Palabras.

2. Botón Ayuda

Este botón nos muestra la ayuda para esta pantalla, las pantallas de ayuda se detallan mas adelante.

3. Botón Dados

Pulsando este botón la palabra es elegida aleatoriamente por la aplicación.

4. Botón Leer

Pulsando este botón la aplicación intentará *reconocer la palabra escrita* en la zona de escritura.

5. Botón Lápiz

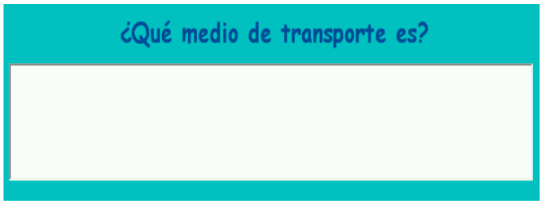
cursor

Como en otras pantallas usando este botón podremos *escribir la palabra* en la zona de escritura arrastrando el lápiz sobre la zona sensible de la Tableta Gráfica o bien botón izquierdo del ratón y arrastrando. Observamos que el cursor cambia a modo lápiz.

6. Botón Goma

cursor

Este es el botón destinado a *borrar los trazos de tinta* dentro de la zona de escritura, si nuestro deseo es borrar todo o parte de la palabra escrita sin que por ello afecte a la labor de reconocimiento. Al hacer uso de este botón podemos observar que el cursor cambia para convertirse en una *goma,* debemos indicar nuestro deseo de reanudar la escritura en su caso pulsando el botón lápiz.



Esta es la zona de escritura de esta pantalla cuando el lápiz de la Tableta Gráfica entra en ella y hemos pulsado el botón lápiz el cursor cambia, indicando que podemos escribir.

Veamos el funcionamiento de este tipo de pantallas mediante un **ejemplo ilustrativo**:

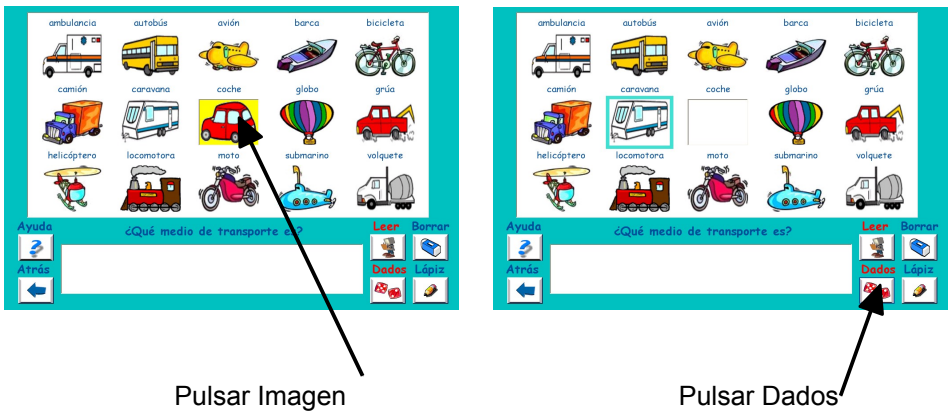


Figura 9.81 Pulsar imagen o Pulsar dados para elegir

El usuario puede elegir directamente la palabra que desea escribir *pulsando sobre la imagen* correspondiente como se muestra en la figura de la izquierda o *pulsando sobre el botón dados*, de esta forma la palabra se elige de forma aleatoria como se muestra en la figura de la derecha, las imágenes se van iluminando hasta llegar a la imagen seleccionada por la aplicación. Si en las opciones de configuración no hemos quitado el sonido de la aplicación, escucharemos la palabra que debemos escribir en la zona de escritura.

Si no elegimos directamente la palabra ni tampoco pulsamos el botón Dados, cuando tratamos de leer no dispondremos de palabra y la aplicación mostrará un mensaje de aviso como el de la figura

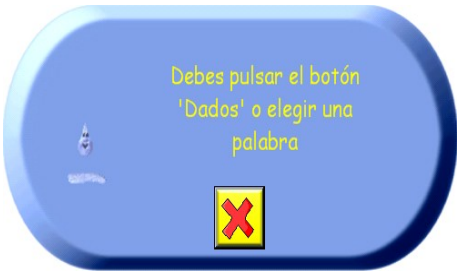


Figura 9.82 Mensaje de Aviso

Si hemos elegido la palabra y pulsamos el botón Leer antes de escribir la palabra correspondiente en la zona de escritura también se mostrará un mensaje de aviso.

Figura 9.83 Mensaje de Aviso

Figura 9.84 Escribir en la Zona de Escritura

Si ya tenemos *seleccionada la palabra* por uno de los dos métodos vistos, podemos escribir en la zona de escritura y pulsar el botón leer para que la aplicación reconozca la palabra escrita.

Figura 9.85 Borrar un error con el Botón Goma

Si nos equivocamos al escribir podemos corregir el error pulsando en el botón Goma, nos permite borrar el error en una letra o la palabra entera si lo deseamos, el cursor cambia para convertirse en una goma de borrar, una vez corregido el error para escribir de nuevo deberemos pulsar el botón Lápiz y a continuación pulsar el botón Leer para que la aplicación reconozca la palabra ya corregida.

Si la escritura es reconocida y corresponde por tanto con palabra elegida por uno de los dos métodos, la aplicación muestra un mensaje indicando que la escritura ha sido realizada correctamente.

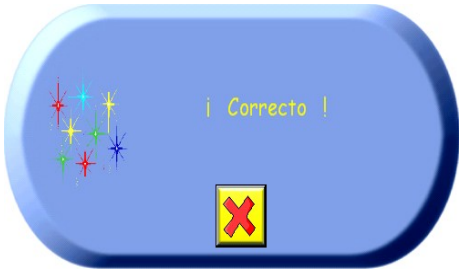


Figura 9.86 Mensaje Afirmativo

Si nos equivocamos al escribir y la aplicación no reconoce la palabra nos mostrará un mensaje indicando que la escritura no ha sido correcta, invitándonos a intentarlo de nuevo.

Figura 9.87 Mensaje Negativo

9.3.10. Pantallas de Ayuda

Todas las pantallas de la aplicación disponen de botones de Ayuda ya mencionados en cada pantalla y representados mediante una de estas dos imágenes.

Figura 9.88 Botones de Ayuda

En cada una de estas pantallas se proporciona:

- Información sobre los botones de que dispone y una breve explicación de funcionamiento.
- Ejemplo de manejo de la pantalla representado por la imagen de la figura y que muestra como se trabaja con cada pantalla.

Figura 9.89 Ejemplo

- En aquellas pantallas donde se ha considerado necesario un breve explicación supletoria sobre los botones mas complejos.

La ayuda sobre cualquier pantalla puede ser consultada desde cualquiera de las pantallas, éstas se presentan mediante pestañas su nombre, pulsando sobre la pestaña que se desea consultar se presenta la pantalla de ayuda elegida, podemos además desplazar las pestañas pulsando sobre los botones adelante y atrás situados a la derecha de las pestañas.

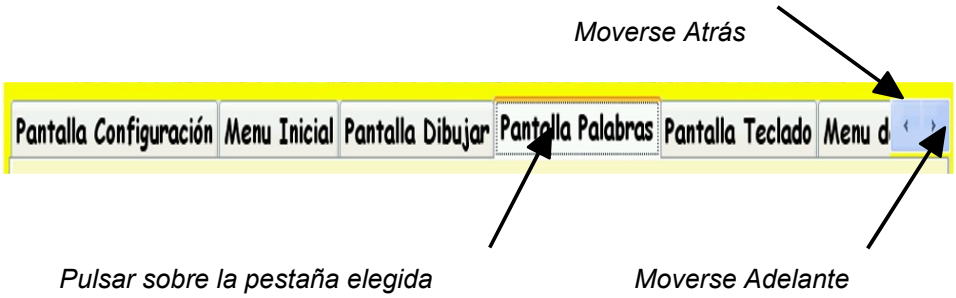


Figura 9.90 Pestañas de Pantallas de Ayuda

A continuación se muestran las pantallas de Ayuda de la Aplicación:

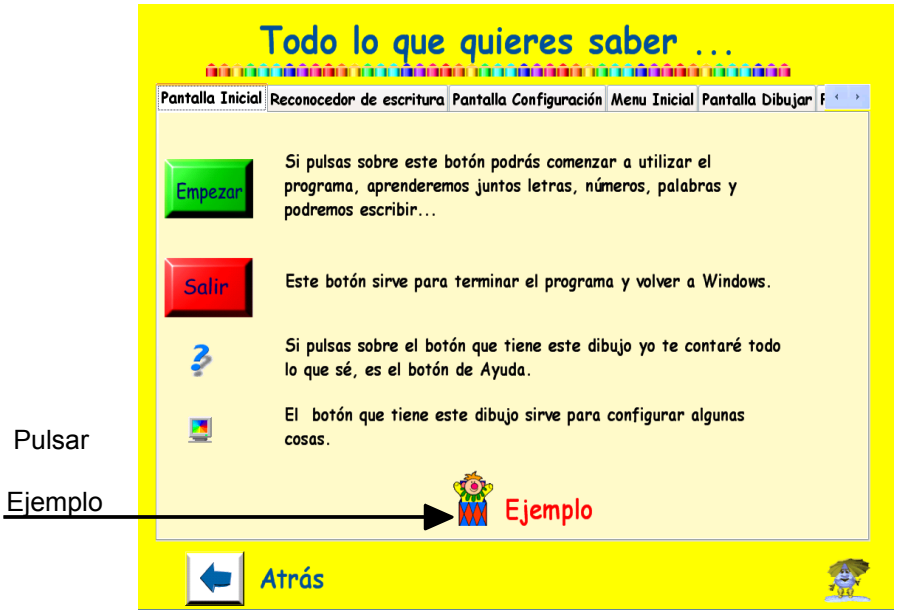
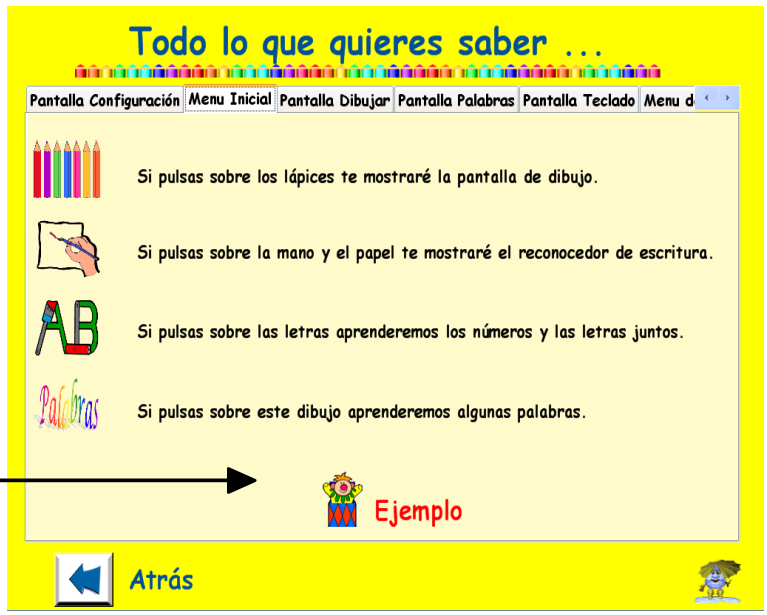


Figura 9.91 Pantalla de Ayuda de la Pantalla Inicial

Botón Cerrar

Figura 9.92 Ejemplo de la Pantalla de Ayuda Inicial

Al pulsar sobre el Ejemplo de la Pantalla de Ayuda de la pantalla Inicial se muestra la siguiente pantalla, para ocultarla debemos pulsar sobre el *botón Cerrar* que se encuentra en la esquina superior derecha.
La Ayuda muestra los botones activos .

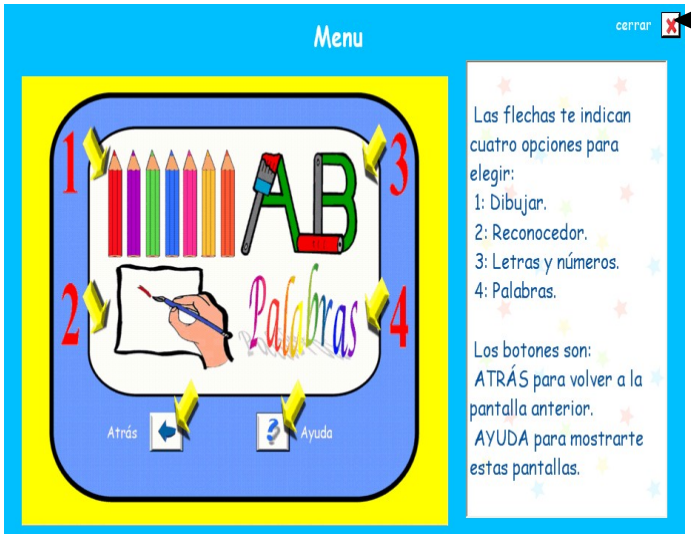


Pulsar

Ejemplo

Figura 9.93 Pantalla de Ayuda de la Pantalla Menú

Botón Cerrar



Al pulsar sobre el Ejemplo de la Pantalla de Ayuda de Menú Inicial se muestra la siguiente pantalla, para ocultarla debemos pulsar sobre el *botón Cerrar* que se encuentra en la esquina superior derecha.

Figura 9.94 Ejemplo de la Pantalla Menú Inicial

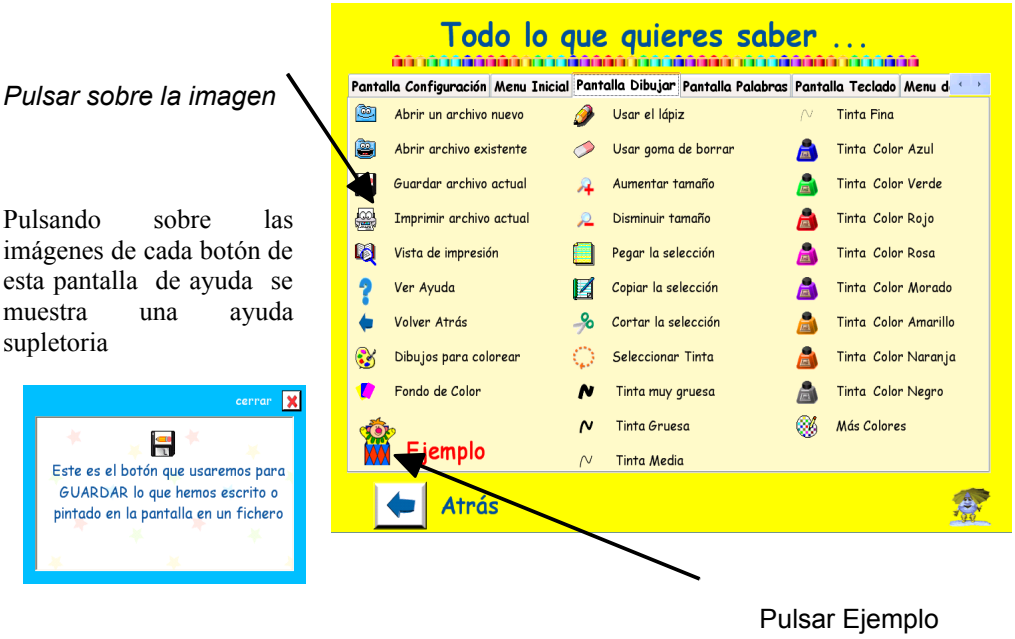


Figura 9.95 Pantalla de Ayuda de la Pantalla de Dibujo

Botón Cerrar

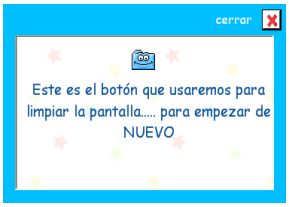
Al pulsar sobre el Ejemplo de la Pantalla de Ayuda de la pantalla de Dibujo se muestra la siguiente pantalla, para ocultarla debemos pulsar sobre el *botón Cerrar* que se encuentra en la esquina superior derecha.

El ejemplo muestra una breve explicación de los que podemos hacer en dicha pantalla.

Figura 9.96 Ejemplo de la Pantalla Dibujar

Pulsar sobre la imagen

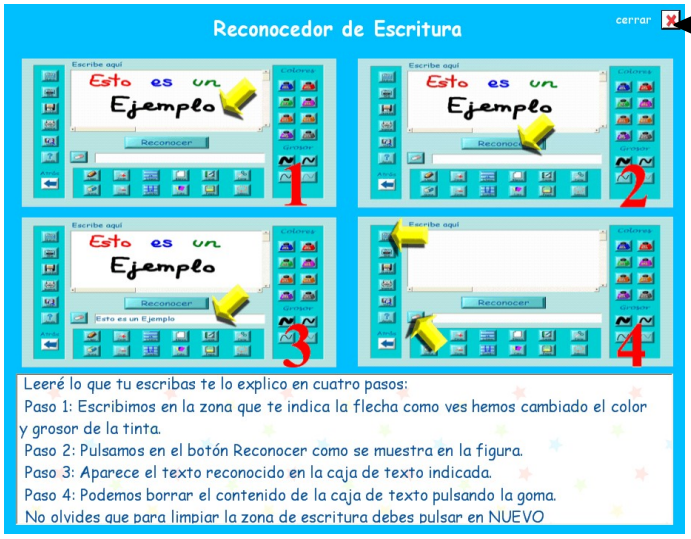
Pulsando sobre las imágenes que muestran los botones de esta pantalla de ayuda se muestra una ayuda supletoria



Pulsar Ejemplo

Figura 9.97 Pantalla de Ayuda de la Pantalla de Reconocedor de Escritura

Botón Cerrar



Al pulsar sobre el Ejemplo de la Pantalla de Ayuda de la pantalla Reconocedor de Escritura se muestra la siguiente pantalla, para ocultarla debemos pulsar sobre el *botón Cerrar* que se encuentra en la esquina superior derecha.

El ejemplo muestra una breve explicación de los que podemos hacer en dicha pantalla.

Figura 9.98 Ejemplo de la Pantalla Reconocedor de Escritura

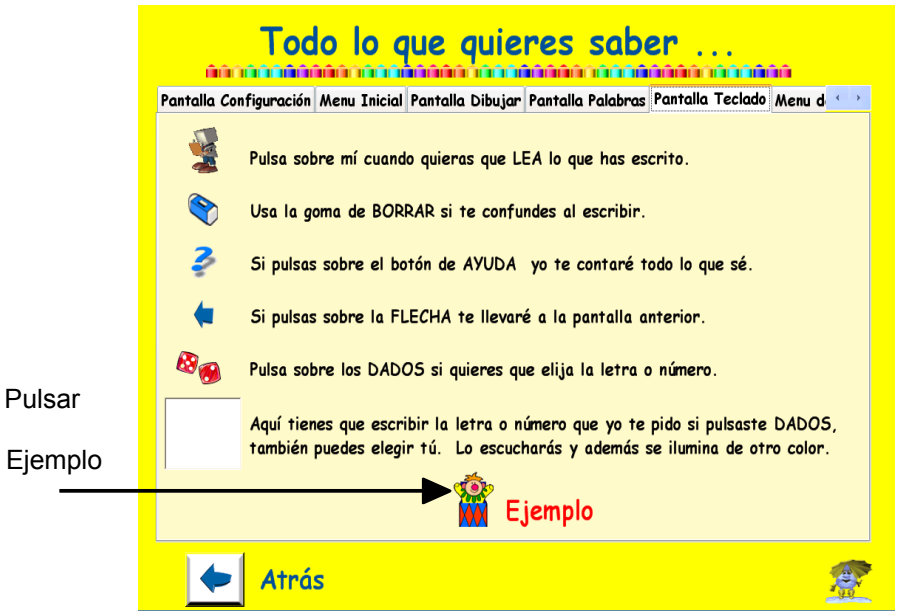


Figura 9.99 Pantalla de Ayuda de la Pantalla Aprender Letras y Números

Al pulsar sobre el Ejemplo de la Pantalla de Ayuda de la pantalla Aprender Letras y Números se muestra la siguiente pantalla, para ocultarla debemos pulsar sobre el *botón Cerrar* que se encuentra en la esquina superior derecha.
El ejemplo muestra cómo manejar la pantalla y los botones activos.

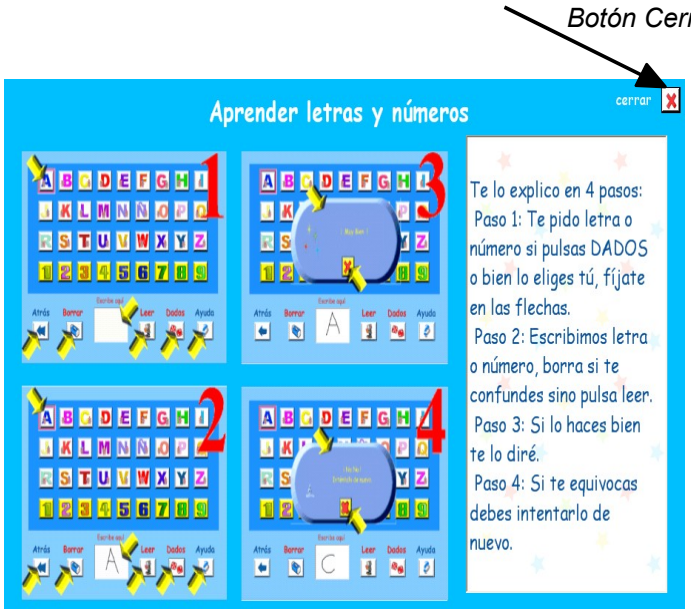


Figura 9.100 Ejemplo de la Pantalla Aprender Letras y Números

Figura 9.101 Pantalla de Ayuda de la Pantalla Menú de Palabras

Botón Cerrar

Al pulsar sobre el Ejemplo de la Ayuda de la pantalla Menú de Palabras se muestra la siguiente pantalla, para ocultarla debemos pulsar sobre el *botón Cerrar* que se encuentra en la esquina superior derecha.

Figura 9.102 Ejemplo de la Pantalla Menú de Palabras

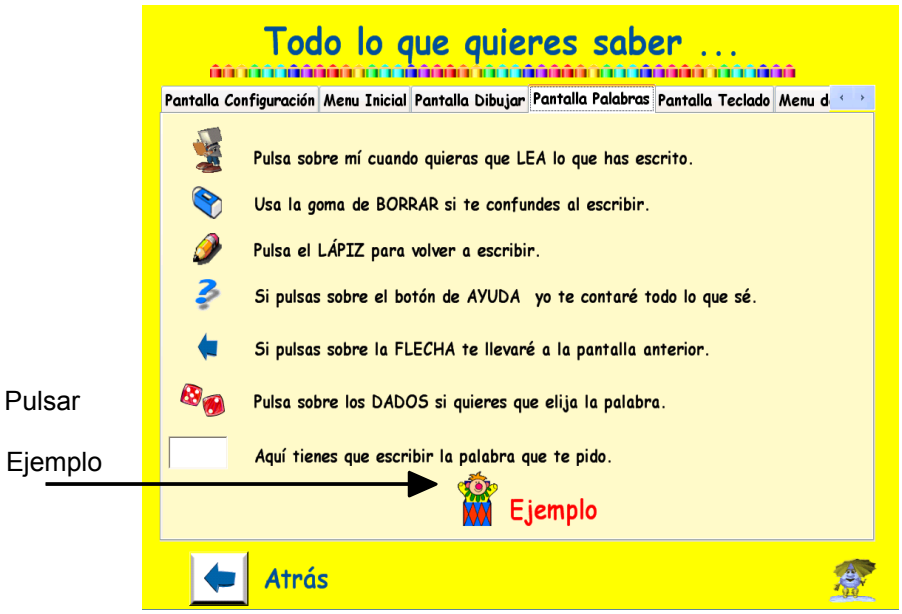


Figura 9.103 Pantalla de Ayuda de la Pantalla Aprender Palabras

Botón Cerrar

Figura 9.104 Ejemplo de la Pantalla Palabras

Puesto que todas las pantallas de palabras son similares se ha escogido una pantalla para mostrar como se manejan este tipo de pantallas.
La pantalla mostrada en la figura se obtiene al pulsar sobre *Ejemplo* en la pantalla de Ayuda

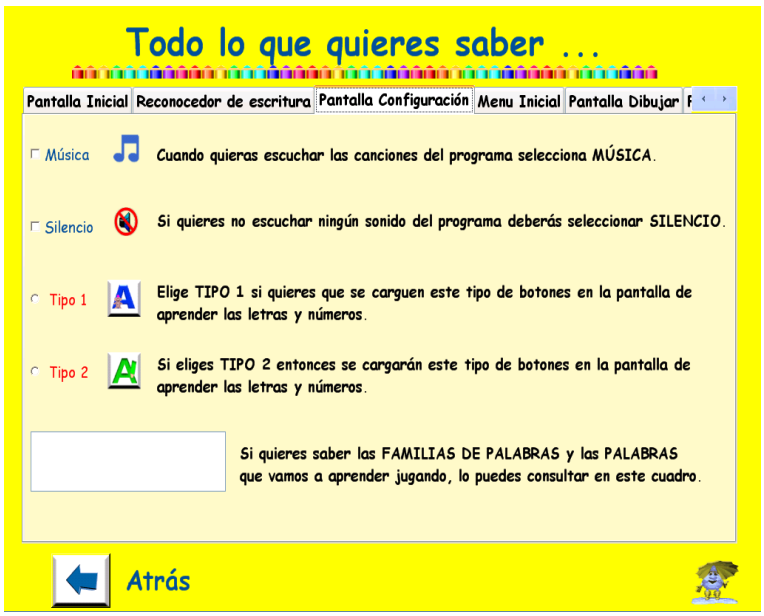


Figura 9.105 Pantalla de Ayuda de la Pantalla de Configuración

La pantalla de Ayuda de Configuración nos proporciona una serie de indicaciones sobre como manejar la pantalla , las opciones posibles de configuración y cómo realizarlas.

Parte V

Conclusiones

Capítulo 10

CONCLUSIONES

10.1. Objetivos alcanzados

- En cuanto al primero de nuestros objetivos hemos conseguido profundizar en el estudio de la librería Microsoft.Ink que es el assembly proporcionado por la plataforma para Tablet PC sobre escritura manual, sobre su contenido y funcionamiento al menos en lo que a tinta, escritura y reconocimiento de escritura manual se refiere.

- Hemos conseguido realizar una aplicación que a nuestro juicio puede ser útil para personas con discapacidad como era nuestro objetivo, proporcionando una nueva forma de aprender letras, números y escritura manual de un numeroso grupo de palabras.

- Hemos logrado desarrollar una aplicación asequible y atractiva, no solo para personas con discapacidad, también puede ser utilizada por niños de corta edad para facilitar el aprendizaje tanto de la escritura como del conocimiento de nuevas palabras.

- La incorporación de una pantalla de dibujo cubre en parte el objetivo de facilitar el desarrollo y mejorar la calidad de vida de las personas con discapacidad, consideramos que también era importante incorporar un poco de diversión a la aplicación para hacerla más atractiva ya que dibujar también es otra forma de comunicarse la primera que utilizan lo niños cuando disponen de lápiz y papel.

- Por otra parte, pretendíamos abrir una nueva vía de estudio para futuros proyectos, más adelante detallamos otros posibles caminos a seguir.

- Por último hemos conseguido aplicar y desarrollar los conocimientos que hemos adquirido, siempre nos han proporcionado la mejor base para enfrentarnos a un entorno de trabajo nuevo y un nuevo lenguaje C#.

10.2. Conclusiones de tipo técnico

Hemos tenido que aprender a trabajar con .NET y su entorno de desarrollo una nueva forma de crear aplicaciones reales con la que no estábamos familiarizadas, partir de cero para desarrollar una aplicación real desde el diseño hasta su entrega al usuario final algo que al principio nos parecía complicado.

Capítulo 11

POSIBLES MEJORAS

En este capitulo mencionaremos posibles mejoras que podrían realizarse utilizando otras partes de la librería Microsoft.Ink que hemos estudiado, enfocadas a mejorar la aplicación y otras posibles vías de estudio para aplicaciones posteriores teniendo presente el objetivo de facilitar la comunicación a personas con discapacidad. Proponemos a continuación algunas de ellas:

11.1 Gestures

Como hemos podido comprobar en el estudio previo del capítulo 4, el objeto gestures proporciona la capacidad de reconocimiento de formas, gestos en aplicaciones y también gestos asociados con el Sistema, esta capacidad es muy interesante ya que proporciona una forma de indicar acciones al sistema con un solo movimiento del lápiz. Característica que puede ser muy útil en el entorno de la discapacidad. El reconocimiento de gestos en el entorno de una aplicación como formas geométricas o flechas también ofrece muchas posibilidades.

11.2 Reconocimiento de voz

Puesto que el módulo de reconocimiento de escritura que debe instalarse para la parte de reconocimiento de escritura a mano incorpora además reconocimiento de voz podría utilizarse esta herramienta en un futuro como otra vía de estudio o ampliación al reconocimiento de escritura, aunque a fecha de hoy únicamente esta disponibles los cuatro reconocedores de voz siguientes: inglés, japonés, chino simplificado y chino tradicional. En un futuro próximo Microsoft incorporará más idiomas. Sin profundizar demasiado en el tema existe también un kit de desarrollo de voz Microsoft Speech SDK, version 5.1. que puede ofrecer muchas posibilidades.

11.3. Creación de un diccionario de palabras a medida

Como hemos mencionado en el capítulo 4: Estudio previo: Tinta digital, mediante el control InkEdit es posible crear un “diccionario a medida”, de forma que las palabras se almacenen en un objeto *WordList* , y podemos introducirlas en una aplicación escribiéndolas a mano, combinando esto con *RecognitionAlternates* que proporciona varias alternativas al texto reconocido.

Esto podría ser muy útil para personas discapacitadas que tienen mermada la capacidad de hablar, podría introducirse como parte del diccionario el vocabulario de uso común y facilitar así un nuevo mecanismo de comunicación que posibilitará la capacidad de conversar.

11.4. Panel de entrada

El objeto PenInputPanel como hemos mencionado previamente podría utilizarse para crear una aplicación con un teclado adaptado para personas discapacitadas, añadiéndole las funcionalidades que proporciona la plataforma Tablet PC, ya que es capaz de emular un teclado hardware, mediante el uso de la tableta gráfica es más fácil seleccionar las teclas en pantalla que en un teclado físico.

Capítulo 12

Bibliografía y fuentes WEB

12.1 Bibliografía

- Miguel Angel Laguna Serrano. Apuntes de la asignatura de Ingeniería del Sofware II.
- C. Larman, "UML y Patrones" Ed Pearson Educación, 2002
- Charles Wright , "Superutilidades para C#" Ed McGraw Hill.
- Grady Booch, James Rumbaugh, Ivan Jacobson, " El lenguaje unificado de desarrollo software", Ed. Addison Wesley Iberoamericana.
- John Sharp, Jon Jagger "Microsoft Visual C#.NET Aprenda ya", Ed McGraw Hill.

12.2. Fuentes WEB

- www.ceapat.org página del Centro Estatal de Autonomía personal y ayudas técnicas del Ministerio de Trabajo y Asuntos Sociales. [21-11-2005]

- www.nichcy.org página del Centro Nacional de Diseminación de Información para Niños con Discapacidades EEUU.[21-11-2005]

- http://www.sindromedown.net página que aborda la problemática de esta discapacidad en concreto.[21-11-2005]

- http://www.campus-oei.org Revista Iberoamericana de educación, lugar de consulta de artículos interesantes sobre al aprendizaje de la lectura en niños.[21-11-2005]

- www.eunate.org/creena.htm, Centro de Recursos de Educación Especial de Navarra, se encarga de proporcionar recursos necesarios de apoyo en los colegios.[21-11-2005]

- www.microsoft.com, página oficial de Microsoft en la que podemos obtener información sobre el entorno de desarrollo, herramientas y sistema operativo para Tablet PC consultando MSDN y sus artículos técnicos, donde se proporciona además el software necesario para la ejecución de la aplicación.[21-11-2005]

- http://www.c-sharpcorner.com, página muy completa sobre C# donde podemos encontrar información sobre .NET y artículos interesantes sobre el lenguaje.[21-11-2005]

- http://www.clikear.com página donde se encuentra un manual web sobre C#. [21-11-2005]
- www.microsoft.com/latam/accesibilidad sitio donde se proporciona información interesante sobre accesibilidad a las tecnologías de la información y aplicaciones de Microsoft. [21-11-2005]
- www.once.es La ONCE y HP desarrollan una experiencia educativa basada en Tablet Pcs adaptados (21-9-05).[21-11-2005]

APÉNDICES

Apéndice A

PANORÁMICA GENERAL .NET

1. Plataforma.NET

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

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

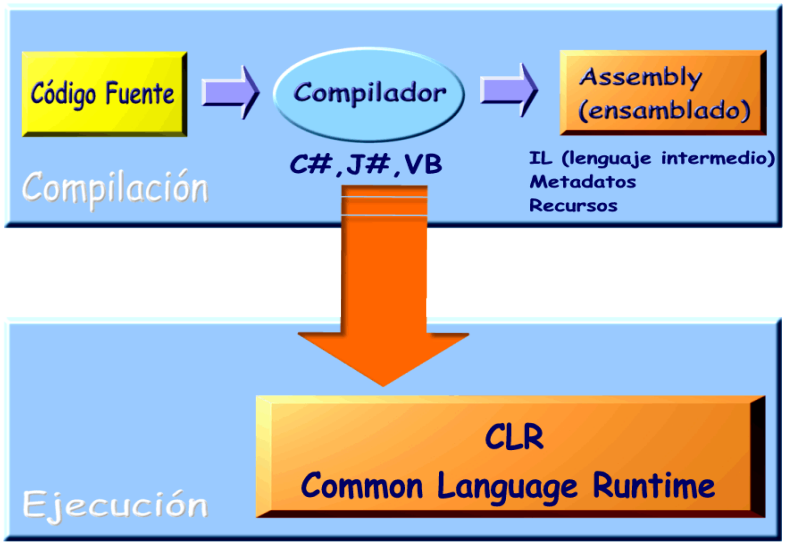


Figura 1 .NET Framework gráficamente

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

1.1 ¿Y como lo hace .NET?

Permite ejecutar software en cualquier lenguaje sobre cualquier dispositivo.
Internet puede hacer los negocios más eficientes y proporcionar servicios a los consumidores mediante el concepto y modelo de programación Servicio Web XML

Propone nuevas formas de interactuar con Pcs (uso de la voz, reconocimiento de escritura...)

Lanzando un nueva plataforma software con Internet como sustrato esencial, con acceso a la información desde cualquier sitio y con gran variedad de dispositivos.

- Nuevo nivel: **.NET Framework** para compartir características y niveles de abstracción ya que se incluyen cambios de formato de los ejecutables, cambios de compiladores y su filosofía de trabajo , y cambio de la biblioteca de clases básicas.
- Nuevos dispositivos: teléfonos, PDAs, Tablet Pcs
- Entorno de hospedaje de Servicios Web personales: Servicios Web básicos como autentificación y almacenamiento de datos. Subscripción a software como Servicio a través de .NET myServices.

Frente a la situación actual:

- Lenguajes de programación y compiladores frente a archivos fuente binarios
- Ejecutables y enlace dinámico en DLLs.
- Modelos de componentes y encapsulamiento con tipos, interfaces, clases y objetos
- Aplicaciones distribuidas, Arquitecturas cliente/servidor en tres niveles
- Internet: Páginas activas, Middleware, seguridad, transacciones, atributos, Máquinas virtuales, interpretación y librerías de abstracción.

Aunque con **algunos problemas** como:

- IDLs y Librerías de Tipos complejos, se separa el interfaz de la implementación (ejecutable)
- Cada entorno de desarrollo debe implementar costosos mecanismos de infraestructura con factoría de clases y de interfaces.
- Control explícito del flujo de ejecución

- En cuanto a Windows:

  Visual C++, Visual C#, Visual J#....(Visual Studio), punteros y API Win32

  Algunas incompatibilidades de tipos entre String y char array

  Reutilización de implementación: herencia en un solo nivel

  Distintas versiones de las DLLs (Infierno de las DLLs)

  Complejidad de instalación

  Construir seguridad implícitamente sobre el sistema

Soluciones:

Compartir características y niveles de abstracción usando **.NET Framewok**

Múltiples lenguajes compilados

Servicios en el desarrollo y ejecución de código, usando una nueva máquina virtual multilenguaje (**Common Language Runtime CLR**)

Lenguaje Intermedio (**MSIL**)

Espacios de nombres (**Namespaces**), librerías de clases base y tipos unificados

Metadatos y ensamblados (assemblies)

Modelos de programación ASP.NET para formularios Web y servicios Web XML

En resumen simplificar y unificar.

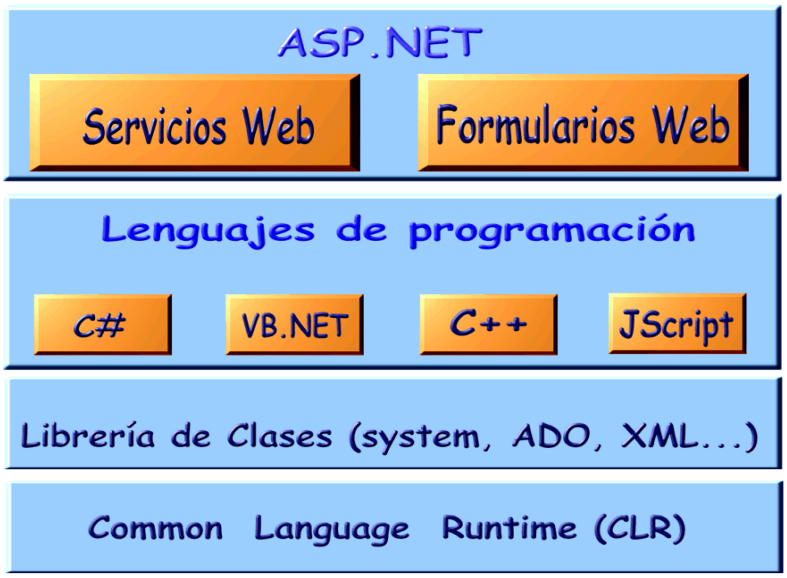


Figura 2 Desarrollo de software con .NET

1.2. NET Framework

El .Net Framework es un componente integral de Windows que proporciona apoyo para construir y ejecutar la nueva generación de aplicaciones y servicios Web XML. Esta diseñado para cumplir los siguiente objetivos:

- Proveer de un contexto consistente de programación orientada a objetos tanto si el objeto se almacena y ejecuta localmente, se ejecuta localmente pero se distribuye via Internet o se ejecuta remotamente.
- Proveer un contexto de ejecución de código que minimiza la implementación de software y los conflictos entre versiones.
- Proveer de un contexto de ejecución de código seguro.
- Para hacer la experiencia del desarrollador coherente variando los tipos de aplicaciones entre aplicaciones basadas en Windows y aplicaciones basadas en Web.
- Construir toda comunicación dentro de normas estándar para asegurar que el código basado en .NET Framework puede integrarse con cualquier otro código.

.Net Framework tiene dos componentes principales: el **Common Language Runtime (CLR)** y **.Net Framework Class Library** o biblioteca de clase base (**BCL**).

El CLR maneja código en el tiempo de ejecución, gestión de memoria, de hilo, y remoting, al también implementar seguridad estricta de tipo promueve seguridad y robustez. De hecho, el concepto de gestión de código es un principio básico del CLR. El código que apunta al CLR es conocido como código administrado, mientras código que no le apunta es conocido como código no administrado. La biblioteca de clase, el otro componente principal de .NET Framework, es una colección orientada a objetos de tipos reusables para desarrollar aplicaciones tanto aplicaciones de interfaz de usuario de línea de comando o gráficas (la Interfaz Gráfica del Usuario GUI) para las aplicaciones se basa en las últimas innovaciones proporcionadas por ASP.NET, como Web Forms y los servicios de Web XML.

- **Desarrollo de aplicaciones Cliente**

Las aplicaciones cliente son lo más cercano al estilo tradicional de aplicación basado en Windows. Estos son los tipos de aplicaciones basados en ventanas o forms de escritorio que facilitan al usuario la realización de una tarea, incluyen aplicaciones como procesadores de texto, hojas de cálculo, etc que utilizan ventanas, menús , botones y otros elementos de la Interfaz Gráfica de Usuario (GUI) y acceden a los recursos locales como el sistema de archivos o periféricos como impresoras.

1.3 Common Language Runtime (CLR)

El **Common Language Runtime** (**CLR**) es el núcleo de la plataforma .NET. Es el motor encargado de gestionar la ejecución de las aplicaciones para ella desarrolladas y a las que ofrece numerosos servicios que simplifican su desarrollo y favorecen su fiabilidad y seguridad. Las principales características y servicios que ofrece el CLR son:

- **Modelo de programación consistente:** A todos los servicios y facilidades ofrecidos por el CLR se accede de la misma forma: a través de un modelo de programación orientado a objetos. Esto es una diferencia importante respecto al modo de acceso a los servicios ofrecidos por los algunos sistemas operativos actuales (por ejemplo, los de la familia Windows), en los que a algunos servicios se les accede a través de llamadas a funciones globales definidas en DLLs y a otros a través de objetos (objetos COM en el caso de la familia Windows)

- **Modelo de programación sencillo:** Con el CLR desaparecen muchos elementos complejos incluidos en los sistemas operativos actuales (registro de Windows, GUIDs, HRESULTS, IUnknown, etc.) El CLR no es que abstraiga al programador de estos conceptos, sino que son conceptos que no existen en la plataforma .NET

- **Eliminación del "infierno de las DLLs":** En la plataforma .NET desaparece el problema conocido como "infierno de las DLLs" que se da en los sistemas operativos actuales de la familia Windows, problema que consiste en que al sustituirse versiones viejas de DLLs compartidas por versiones nuevas puede que aplicaciones que fueron diseñadas para ser ejecutadas usando las viejas dejen de funcionar si las nuevas no son 100% compatibles con las

anteriores. En la plataforma .NET las versiones nuevas de las DLLs pueden coexistir con las viejas, de modo que las aplicaciones diseñadas para ejecutarse usando las viejas podrán seguir usándolas tras instalación de las nuevas. Esto, obviamente, simplifica mucho la instalación y desinstalación de software.

- **Ejecución multiplataforma:** El CLR actúa como una máquina virtual, encargándose de ejecutar las aplicaciones diseñadas para la plataforma .NET. Es decir, cualquier plataforma para la que exista una versión del CLR podrá ejecutar cualquier aplicación .NET. Microsoft ha desarrollado versiones del CLR para la mayoría de las versiones de Windows: Windows 95, Windows 98, Windows ME, Windows NT 4.0, Windows 2000, Windows XP y Windows CE (que puede ser usado en CPUs que no sean de la familia x86) Por otro lado Microsoft ha firmado un acuerdo con Corel para portar el CLR a Linux y también hay terceros que están desarrollando de manera independiente versiones de libre distribución del CLR para Linux. Asimismo, dado que la arquitectura del CLR está totalmente abierta, es posible que en el futuro se diseñen versiones del mismo para otros sistemas operativos.

- **Integración de lenguajes:** Desde cualquier lenguaje para el que exista un compilador que genere código para la plataforma .NET es posible utilizar código generado para la misma usando cualquier otro lenguaje tal y como si de código escrito usando el primero se tratase. Microsoft ha desarrollado un compilador de C# que genera código de este tipo, así como versiones de sus compiladores de Visual Basic (Visual Basic.NET) y C++ (C++ con extensiones gestionadas) que también lo generan y una versión del intérprete de JScript (JScript.NET) que puede interpretarlo. La integración de lenguajes es tal que es posible escribir una clase en C# que herede de otra escrita en Visual Basic.NET que, a su vez, herede de otra escrita en C++ con extensiones gestionadas.

- **Gestión de memoria:** El CLR incluye un **recolector de basura** que evita que el programador tenga que tener en cuenta cuándo ha de destruir los objetos que dejen de serle útiles. Este recolector es una aplicación que se activa cuando se quiere crear algún objeto nuevo y se detecta que no queda memoria libre para hacerlo, caso en que el recolector recorre la memoria dinámica asociada a la aplicación, detecta qué objetos hay en ella que no puedan ser accedidos por el código de la aplicación, y los elimina para limpiar la memoria de "objetos basura" y permitir la creación de otros nuevos. Gracias a este recolector se evitan errores de programación muy comunes como intentos de borrado de objetos ya borrados, agotamiento de memoria por olvido de eliminación de objetos inútiles o solicitud de acceso a miembros de objetos ya destruidos.

- **Seguridad de tipos:** El CLR facilita la detección de errores de programación difíciles de localizar comprobando que toda conversión de tipos que se realice durante la ejecución de una aplicación .NET se haga de modo que los tipos origen y destino sean compatibles.

- **Aislamiento de procesos:** El CLR asegura que desde código perteneciente a un determinado proceso no se pueda acceder a código o datos pertenecientes a otro, lo que evita errores de programación muy frecuentes e impide que unos procesos puedan atacar a otros. Esto se consigue gracias al sistema de seguridad de tipos antes comentado, pues evita que se pueda convertir un objeto a un tipo de mayor tamaño que el suyo propio, ya que al tratarlo como un objeto de mayor tamaño podría accederse a espacios en memoria ajenos a él que podrían pertenecer a otro proceso. También se consigue gracias a que no se permite acceder a posiciones arbitrarias de memoria.

- **Tratamiento de excepciones:** En el CLR todo los errores que se puedan producir durante la ejecución de una aplicación se propagan de igual manera: mediante excepciones. Esto es muy diferente a como se venía haciendo en los sistemas Windows hasta la aparición de la plataforma .NET, donde ciertos errores se transmitían mediante códigos de error en formato Win32, otros mediante HRESULTs y otros mediante excepciones.

  El CLR permite que excepciones lanzadas desde código para .NET escrito en un cierto lenguaje se puedan capturar en código escrito usando otro lenguaje, e incluye mecanismos de depuración que pueden saltar desde código escrito para .NET en un determinado lenguaje a código escrito en cualquier otro. Por ejemplo, se puede recorrer la pila de llamadas de una excepción aunque ésta incluya métodos definidos en otros módulos usando otros lenguajes.

- **Soporte multihilo:** El CLR es capaz de trabajar con aplicaciones divididas en múltiples hilos de ejecución que pueden ir evolucionando por separado en paralelo o intercalándose, según el número de procesadores de la máquina sobre la que se ejecuten. Las aplicaciones pueden lanzar nuevos hilos, destruirlos, suspenderlos por un tiempo o hasta que les llegue una notificación, enviarles notificaciones, sincronizarlos, etc.

- **Distribución transparente:** El CLR ofrece la infraestructura necesaria para crear objetos remotos y acceder a ellos de manera completamente transparente a su localización real, tal y como si se encontrasen en la máquina que los utiliza.

- **Seguridad avanzada:** El CLR proporciona mecanismos para restringir la ejecución de ciertos códigos o los permisos asignados a los mismos según su procedendecia o el usuario que los ejecute. Es decir, puede no darse el mismo nivel de confianza a código procedente de Internet que a código instalado localmente o procedente de una red local; puede no darse los mismos permisos a código procedente de un determinado fabricante que a código de otro; y puede no darse los mismos permisos a un mismo códigos según el usuario que lo esté ejecutando o según el rol que éste desempeñe. Esto permite asegurar al administrador de un sistema que el código que se esté ejecutando no pueda poner en peligro la integridad de sus archivos, la del registro de Windows, etc.

- **Interoperabilidad con código antiguo:** El CLR incorpora los mecanismos necesarios para poder acceder desde código escrito para la plataforma .NET a código escrito previamente a la aparición de la misma y, por tanto, no preparado para ser ejecutando dentro de ella. Estos mecanismos permiten tanto el acceso a objetos COM como el acceso a funciones sueltas de DLLs preexistentes (como la API Win32)

Como se puede deducir de las características comentadas, el CLR lo que hace es gestionar la ejecución de las aplicaciones diseñadas para la plataforma .NET. Por esta razón, al código de estas aplicaciones se le suele llamar **código gestionado o administrado**, y al código no escrito para ser ejecutado directamente en la plataforma .NET se le suele llamar **código no gestionado o no administrado**.

1.4 Microsoft Intermediate Language (MSIL)

Ninguno de los compiladores que generan código para la plataforma .NET produce código máquina para CPUs x86 ni para ningún otro tipo de CPU concreta, sino que generan código escrito en el lenguaje intermedio conocido como **Microsoft Intermediate Lenguage** (**MSIL**) El CLR da a las aplicaciones la sensación de que se están ejecutando sobre una máquina virtual, y precisamente MSIL es el código máquina de esa máquina virtual. Es decir, MSIL es el único código que es capaz de interpretar el CLR, y por tanto cuando se dice que un compilador genera código para la plataforma .NET lo que se está diciendo es que genera MSIL.
La principal ventaja del MSIL es que facilita la ejecución multiplataforma y la integración entre lenguajes al ser independiente de la CPU y proporcionar un formato común para el código máquina generado por todos los compiladores que generen código para .NET. Sin embargo, dado que las CPUs no pueden ejecutar directamente MSIL, antes de ejecutarlo habrá que convertirlo al código nativo de la CPU sobre la que se vaya a ejecutar. De esto se encarga un componente del CLR conocido como compilador JIT (Just-In-Time) o jitter que va convirtiendo dinámicamente el código MSIL a ejecutar en código nativo según sea necesario.

1.5 Metadatos

En la plataforma .NET se distinguen dos tipos de **módulos** de código compilado: **ejecutables** (extensión **.exe**) y **librerías de enlace dinámico** (extensión **.dll** generalmente) Ambos son ficheros que contienen definiciones de tipos de datos, y la diferencia entre ellos es que sólo los primeros disponen de un método especial que sirve de punto de entrada a partir del que es posible ejecutar el código que contienen haciendo una llamada desde la línea de comandos del sistema operativo. A ambos tipos de módulos se les suele llamar **ejecutables portables** (PE), ya que su código puede ejecutarse en cualquiera de los diferentes sistemas operativos de la familia Windows para los que existe alguna versión del CLR.

El contenido de un módulo no es sólo MSIL, sino que también consta de otras dos áreas muy importantes: la cabecera de CLR y los metadatos:

- La **cabecera de CLR** es un pequeño bloque de información que indica que se trata de un módulo gestionado e indica es la versión del CLR que necesita, cuál es su firma digital, cuál es su punto de entrada (si es un ejecutable), etc.

- Los **metadatos** son un conjunto de datos organizados en forma de tablas que almacenan información sobre los tipos definidos en el módulo, los miembros de éstos y sobre cuáles son los tipos externos al módulo a los que se les referencia en el módulo. Los metadatos de cada modulo los genera automáticamente el compilador al crearlo.

El significado de los metadatos es similar al de otras tecnologías previas a la plataforma .NET como lo son los ficheros IDL. Sin embargo, los metadatos tienen dos ventajas importantes sobre éstas: contiene más información y siempre se almacenan incrustados en el módulo al que describen, haciendo imposible la separación entre ambos. Además es posible tanto consultar los metadatos de cualquier módulo a través de las clases del espacio de nombres **System.Reflection** de la BCL como añadirles información adicional mediante **atributos.**

1.6 Ensamblados

Un **ensamblado** es una agrupación lógica de uno o más módulos o ficheros de recursos (ficheros .GIF, .HTML, etc.) que se engloban bajo un nombre común. Un programa puede acceder a información o código almacenados en un ensamblado sin tener que conocer cuál es el fichero en concreto donde se encuentran, por lo que los ensamblados nos permiten abstraernos de la ubicación física del código que ejecutemos o de los recursos que usemos. Por ejemplo, podemos incluir todos los tipos de una aplicación en un mismo ensamblado pero colocando los más frecuentemente usados en un cierto módulo y los menos usados en otro, de modo que sólo se descarguen de Internet los últimos si es que se van a usar.

Todo ensamblado contiene un **manifiesto**, que son metadatos con información sobre las características del ensamblado. Este manifiesto puede almacenarse en cualquiera de los módulos que formen el ensamblado o en uno específicamente creado para ello, siendo lo último necesario cuando sólo contiene recursos (**ensamblado satélite**)

Hay dos tipos de ensamblados: **ensamblados privados** y **ensamblados compartidos**. Los privados se almacenan en el mismo directorio que la aplicación que los usa y sólo puede usarlos ésta, mientras que los compartidos se almacenan en un **caché de ensamblado global** (GAC) y pueden usarlos cualquiera que haya sido compilada referenciándolos.

También para evitar problemas, en el GAC se pueden mantener múltiples versiones de un mismo ensamblado. Así, si una aplicación fue compilada usando una cierta versión de un determinado ensamblado compartido, cuando se ejecute sólo podrá hacer uso de esa versión del ensamblado y no de alguna otra más moderna que se hubiese instalado en el GAC. De esta forma se soluciona el problema del **infierno de las DLL.**

1.7 Librería de clase base (BCL)

La Librería de Clase Base (BCL) es una librería incluida en el *.NET Framework* formada por cientos de tipos de datos que permiten acceder a los servicios ofrecidos por el CLR y a las funcionalidades más frecuentemente usadas a la hora de escribir programas. Además, a partir de estas clases prefabricadas el programador puede crear nuevas clases que mediante herencia extiendan su funcionalidad y se integren a la perfección con el resto de clases de la BCL. Por ejemplo, implementando ciertos interfaces podemos crear nuevos tipos de colecciones que serán tratadas exactamente igual que cualquiera de las colecciones incluidas en la BCL.

Esta librería está escrita en MSIL, por lo que puede usarse desde cualquier lenguaje cuyo compilador genere MSIL. A través de las clases suministradas en ella es posible desarrollar cualquier tipo de aplicación, desde las tradicionales aplicaciones de ventanas, consola o servicio de Windows NT hasta los novedosos servicios Web y páginas ASP.NET. Es tal la riqueza de servicios que ofrece que incluso es posible crear lenguajes que carezcan de librería de clases propia y sólo se basen en la BCL -como C#.

Dada la amplitud de la BCL, ha sido necesario organizar las clases en ella incluida en **espacios de nombres** que agrupen clases con funcionalidades similares.

Por ejemplo, los espacios de nombres más usados son:

Espacio de nombres	Utilidad de los tipos de datos que contiene
System	Tipos muy frecuentemente usados, como los los tipos básicos, tablas, excepciones, fechas, números aleatorios, recolector de basura, entrada/salida en consola, etc.
System.Collections	Colecciones de datos de uso común como pilas, colas, listas, diccionarios, etc.
System.Data	Manipulación de bases de datos. Forman la denominada arquitectura ADO.NET.
System.IO	Manipulación de ficheros y otros flujos de datos.
System.Net	Realización de comunicaciones en red.
System.Reflection	Acceso a los metadatos que acompañan a los módulos de código.
System.Runtime.Remoting	Acceso a objetos remotos.
System.Security	Acceso a la política de seguridad en que se basa el CLR.
System.Threading	Manipulación de hilos.
System.Web.UI.WebControls	Creación de interfaces de usuario basadas en ventanas para aplicaciones Web.
System.Windows.Forms	Creación de interfaces de usuario basadas en ventanas para aplicaciones estándar.
System.XML	Acceso a datos en formato XML.

Tabla 1 Espacios de nombres de la BCL más usados

1.8 Common Type System (CTS)

El **Common Type System** (CTS) o Sistema de Tipo Común es el conjunto de reglas que han de seguir las definiciones de tipos de datos para que el CLR las acepte. Es decir, aunque cada lenguaje gestionado disponga de su propia sintaxis para definir tipos de datos, en el MSIL resultante de la compilación de sus códigos fuente se han de cumplir las reglas del CTS. Algunos ejemplos de estas reglas son:

- Cada tipo de dato puede constar de cero o más miembros. Cada uno de estos miembros puede ser un campo, un método, una propiedad o un evento.

- No puede haber herencia múltiple, y todo tipo de dato ha de heredar directa o indirectamente de **System.Object**.

- Los modificadores de acceso admitidos son:

Modificador	**Código desde el que es accesible el miembro**
public	Cualquier código
private	Código del mismo tipo de dato
family	Código del mismo tipo de dato o de hijos de éste.
assembly	Código del mismo ensamblado
family and assembly	Código del mismo tipo o de hijos de éste ubicado en el mismo ensamblado
family or assembly	Código del mismo tipo o de hijos de éste, o código ubicado en el mismo ensamblado

Tabla 2 Modificadores de acceso a miembros admitidos por el CTS

1.9 Common Language Specification (CLS)

El **Common Language Specification** (CLS) o Especificación del Lenguaje Común es un conjunto de reglas que han de seguir las definiciones de tipos que se hagan usando un determinado lenguaje gestionado o administrado si se desea que sean accesibles desde cualquier otro lenguaje gestionado. Obviamente, sólo es necesario seguir estas reglas en las definiciones de tipos y miembros que sean accesibles externamente, y no la en las de los privados. Además, si no importa la interoperabilidad entre lenguajes tampoco es necesario seguirlas. A continuación se listan algunas de reglas significativas del CLS:

- Los tipos de datos básicos admitidos son **bool**, **char**, **byte**, **short**, **int**, **long**, **float**, **double**, **string** y **object** .

- Las tablas han de tener una o más dimensiones, y el número de dimensiones de cada tabla ha de ser fijo. Además, han de indexarse empezando a contar desde 0.

- Se pueden definir tipos abstractos y tipos sellados. Los tipos sellados no pueden tener miembros abstractos.

- Las excepciones han de derivar de **System.Exception**, los delegados de **System.Delegate**, las enumeraciones de **System.Enum**, y los tipos por valor que no sean enumeraciones de **System.ValueType**.

- Los métodos de acceso a propiedades en que se traduzcan las definiciones get/set de éstas han de llamarse de la forma **get_X** y **set_X** respectivamente, donde X es el nombre de la propiedad; los de acceso a indizadores han de traducirse en métodos **get_Item** y **set_Item**; y en el caso de los eventos, sus definiciones add/remove han de traducirse en métodos **add_X** y **remove_X**.

- En las definiciones de atributos sólo pueden usarse enumeraciones o datos de los siguientes tipos: **System.Type**, **string**, **char**, **bool**, **byte**, **short**, **int**, **long**, **float**, **double** y **object**.

- En un mismo ámbito no se pueden definir varios identificadores cuyos nombres sólo difieran en la capitalización usada. De este modo se evitan problemas al acceder a ellos usando lenguajes no sensibles a mayúsculas.

- Las enumeraciones no pueden implementar interfaces, y todos sus campos han de ser estáticos y del mismo tipo. El tipo de los campos de una enumeración sólo puede ser uno de estos cuatro tipos básicos: **byte**, **short**, **int** o **long**.

2. El lenguaje C#

2.1 Origen y necesidad de un nuevo lenguaje

C# (leído en inglés "C Sharp" y en español "C Almohadilla") es el nuevo lenguaje de propósito general diseñado por Microsoft para su plataforma .NET. Sus principales creadores son Scott Wiltamuth y Anders Hejlsberg, éste último también conocido por haber sido el diseñador del lenguaje Turbo Pascal y la herramienta RAD Delphi.

Aunque es posible escribir código para la plataforma .NET en muchos otros lenguajes, C# es el único que ha sido diseñado específicamente para ser utilizado en ella, por lo que programarla usando C# es mucho más sencillo e intuitivo que hacerlo con cualquiera de los otros lenguajes ya que C# carece de elementos heredados innecesarios en .NET. Por esta razón, se suele decir que C# es el **lenguaje nativo de .NET**

La sintaxis y estructuración de C# es muy parecida a la de C++ o Java, puesto que la intención de Microsoft es facilitar la migración de códigos escritos en estos lenguajes a C# y facilitar su aprendizaje a los desarrolladores habituados a ellos. Sin embargo, su sencillez y el alto nivel de productividad son comparables con los de Visual Basic.

En resumen, C# es un lenguaje de programación que toma las mejores características de lenguajes preexistentes como Visual Basic, Java o C++ y las combina en uno solo. El hecho de ser relativamente reciente no implica que sea inmaduro, pues Microsoft ha escrito la mayor parte de la BCL usándolo, por lo que su compilador es el más depurado y optimizado de los incluidos en el *.NET Framework SDK*

2.2 Características de C#

A continuación se recoge de manera resumida las principales características de C# Alguna de las características aquí señaladas no son exactamente propias del lenguaje sino de la plataforma .NET en general, y si aquí se comentan es porque tienen una repercusión directa en el lenguaje:

- **Sencillez:** C# elimina muchos elementos que otros lenguajes incluyen y que son innecesarios en .NET. Por ejemplo:
  - El código escrito en C# es **autocontenido**, lo que significa que no necesita de ficheros adicionales al propio fuente tales como ficheros de cabecera o ficheros IDL
  - El tamaño de los tipos de datos básicos es fijo e independiente del compilador, sistema operativo o máquina para quienes se compile lo que facilita la portabilidad del código.
  - No se incluyen elementos poco útiles de lenguajes como C++ tales como macros, herencia múltiple o la necesidad de un operador diferente del punto (**.**) acceder a miembros de espacios de nombres.
- **Modernidad:** C# incorpora en el propio lenguaje elementos que a lo largo de los años ha ido demostrándose son muy útiles para el desarrollo de aplicaciones y que en otros lenguajes como Java o C++ hay que simular, como un tipo básico **decimal** que permita realizar operaciones de alta precisión con reales de 128 bits (muy útil en el mundo financiero), la inclusión de una instrucción **foreach** que permita recorrer colecciones con facilidad y es ampliable a tipos definidos por el usuario, la inclusión de un tipo básico **string** para representar cadenas o la distinción de un tipo **bool** específico para representar valores lógicos.
- **Orientación a objetos:** Como todo lenguaje de programación de propósito general actual, C# es un lenguaje orientado a objetos, aunque eso es más bien una característica del CTS que de C#. Una diferencia de este enfoque orientado a objetos respecto al de otros lenguajes como C++ es que el de C# es más puro en tanto que no admiten ni funciones ni variables globales sino que todo el código y datos han de definirse dentro de definiciones de tipos de datos, lo que reduce problemas por conflictos de nombres y facilita la legibilidad del código.

C# soporta todas las características propias del paradigma de programación orientada a objetos: **encapsulación**, **herencia** y **polimorfismo**.

En lo referente a la encapsulación es importante señalar que aparte de los típicos modificadores **public**, **private** y **protected**, C# añade un cuarto modificador llamado **internal**, que puede combinarse con **protected** e indica que al elemento a cuya definición precede sólo puede accederse desde su mismo ensamblado.

Respecto a la herencia C# sólo admite herencia simple de clases ya que la múltiple provoca más quebraderos de cabeza que facilidades y en la mayoría de los casos su utilidad puede ser simulada con facilidad mediante herencia múltiple de interfaces. De todos modos, esto vuelve a ser más bien una característica propia del CTS que de C#.

Por otro lado y a diferencia de Java, en C# se ha optado por hacer que todos los métodos sean por defecto sellados y que los redefinibles hayan de marcarse con el modificador **virtual**, lo que permite evitar errores derivados de redefiniciones accidentales. Además, un efecto secundario de esto es que las llamadas a los métodos serán más eficientes por defecto al no tenerse que buscar en la tabla de funciones virtuales la implementación de los mismos a la que se ha de llamar. Otro efecto secundario es que permite que las llamadas a los métodos virtuales se puedan hacer más eficientemente al contribuir a que el tamaño de dicha tabla se reduzca.

- **Orientación a componentes:** La propia sintaxis de C# incluye elementos propios del diseño de componentes que otros lenguajes tienen que simular mediante construcciones más o menos complejas. Es decir, la sintaxis de C# permite definir cómodamente **propiedades** (similares a campos de acceso controlado), **eventos** (asociación controlada de funciones de respuesta a notificaciones) o **atributos** (información sobre un tipo o sus miembros)

- **Gestión automática de memoria:** Todo lenguaje de .NET tiene a su disposición el recolector de basura del CLR. Esto tiene el efecto en el lenguaje de que no es necesario incluir instrucciones de destrucción de objetos. Sin embargo, dado que la destrucción de los objetos a través del recolector de basura es indeterminista y sólo se realiza cuando éste se active –ya sea por falta de memoria, finalización de la aplicación o solicitud explícita en el fuente-, C# también proporciona un mecanismo de liberación de recursos determinista a través de la instrucción **using**.

- **Seguridad de tipos:** C# incluye mecanismos que permiten asegurar que los accesos a tipos de datos siempre se realicen correctamente, lo que permite evita que se produzcan errores difíciles de detectar por acceso a memoria no perteneciente a ningún objeto y es especialmente necesario en un entorno gestionado por un recolector de basura. Para ello se toman medidas del tipo:

    - Sólo se admiten **conversiones entre tipos compatibles**. Esto es, entre un tipo y antecesores suyos, entre tipos para los que explícitamente se haya definido un operador de conversión, y entre un tipo y un tipo hijo suyo del que un objeto del primero almacenase una referencia del segundo (**downcasting**) Obviamente, lo último sólo puede comprobarlo en tiempo de ejecución el CLR y no el compilador, por lo que en realidad el CLR y el compilador colaboran para asegurar la corrección de las conversiones.

    - No se pueden usar **variables no inicializadas**. El compilador da a los campos un valor por defecto consistente en ponerlos a cero y controla mediante análisis del flujo de control del fuente que no se lea ninguna variable local sin que se le haya asignado previamente algún valor.

    - Se comprueba que todo **acceso a los elementos de una tabla** se realice con índices que se encuentren dentro del rango de la misma.

    - Se puede controlar la **producción de desbordamientos** en operaciones aritméticas, informándose de ello con una excepción cuando ocurra.

- C# incluye **delegados**, que son similares a los punteros a funciones de C++ pero siguen un enfoque orientado a objetos, pueden almacenar referencias a varios métodos simultáneamente, y se comprueba que los métodos a los que apunten tengan parámetros y valor de retorno del tipo indicado al definirlos.

- Pueden definirse métodos que admitan un número indefinido de parámetros de un cierto tipo, y a diferencia lenguajes como C/C++, en C# siempre se comprueba que los valores que se les pasen en cada llamada sean de los tipos apropiados.

- **Instrucciones seguras:** Para evitar errores muy comunes, en C# se han impuesto una serie de restricciones en el uso de las instrucciones de control más comunes. Por ejemplo, la guarda de toda condición ha de ser una expresión condicional y no aritmética, con lo que se evitan errores por confusión del operador de igualdad (==) con el de asignación (=); y todo caso de un **switch** ha de terminar en un **break** o **goto** que indique cuál es la siguiente acción a realizar, lo que evita la ejecución accidental de casos y facilita su reordenación.

- **Sistema de tipos unificado: E**n C# todos los tipos de datos que se definan siempre derivarán, aunque sea de manera implícita, de una clase base común llamada **System.Object**, por lo que dispondrán de todos los miembros definidos en ésta clase (es decir, serán "objetos")

  El hecho de que todos los tipos del lenguaje deriven de una clase común facilita enormemente el diseño de colecciones genéricas que puedan almacenar objetos de cualquier tipo.

- **Extensibilidad de tipos básicos:** C# permite definir, a través de **estructuras**, tipos de datos para los que se apliquen las mismas optimizaciones que para los tipos de datos básicos. Es decir, que se puedan almacenar directamente en pila (luego su creación, destrucción y acceso serán más rápidos) y se asignen por valor y no por referencia. Para conseguir que lo último no tenga efectos negativos al pasar estructuras como parámetros de métodos, se da la posibilidad de pasar referencias a pila a través del modificador de parámetro **ref**.

- **Extensibilidad de operadores:** Para facilitar la legibilidad del código y conseguir que los nuevos tipos de datos básicos que se definan a través de las estructuras estén al mismo nivel que los básicos predefinidos en el lenguaje, C# permite redefinir el significado de la mayoría de los operadores -incluidos los de conversión, tanto para conversiones implícitas como explícitas- cuando se apliquen a diferentes tipos de objetos.

  Las redefiniciones de operadores se hacen de manera inteligente, de modo que a partir de una única definición de los operadores ++ y -- el compilador puede deducir automáticamente como ejecutarlos de manera prefijas y postifja; y definiendo operadores simples (como +), el compilador deduce cómo aplicar su versión de asignación compuesta (+=) Además, para asegurar la consistencia, el compilador vigila que los operadores con opuesto siempre se redefinan por parejas (por ejemplo, si se redefine ==, también hay que redefinir **!=**).

  También se da la posibilidad, a través del concepto de **indizador**, de redefinir el significado del operador **[]** para los tipos de dato definidos por el usuario, con lo que se consigue que se pueda acceder al mismo como si fuese una tabla. Esto es muy útil para trabajar con tipos que actúen como colecciones de objetos.

- **Extensibilidad de modificadores:** C# ofrece, a través del concepto de **atributos**, la posibilidad de añadir a los metadatos del módulo resultante de la compilación de cualquier fuente información adicional a la generada por el compilador que luego podrá ser consultada en tiempo ejecución a través de la librería de reflexión de .NET . Esto, que más bien es una característica propia de la plataforma .NET y no de C#, puede usarse como un mecanismo para definir nuevos modificadores.

- **Versionable:** C# incluye una **política de versionado** que permite crear nuevas versiones de tipos sin temor a que la introducción de nuevos miembros provoquen errores difíciles de detectar en tipos hijos previamente desarrollados y ya extendidos con miembros de igual nombre a los recién introducidos.

  Si una clase introduce un nuevo método cuyas redefiniciones deban seguir la regla de llamar a la versión de su padre en algún punto de su código, difícilmente seguirían esta regla miembros de su misma signatura definidos en clases hijas previamente a la definición del mismo en la clase padre; o si introduce un nuevo campo con el mismo nombre que algún método de una clase hija, la clase hija dejará de funcionar. Para evitar que esto ocurra, en C# se toman dos medidas:

  - Se obliga a que toda redefinición deba incluir el modificador **override**, con lo que la versión de la clase hija nunca sería considerada como una redefinición de la versión de miembro en la clase padre ya que no incluiría **override**. Para evitar que por accidente un programador incluya este modificador, sólo se permite incluirlo en miembros que tengan la misma signatura que miembros marcados como redefinibles mediante el modificador **virtual**. Así además se evita el error tan frecuente en Java de creerse haber redefinido un miembro, pues si el miembro con **override** no existe en la clase padre se producirá un error de compilación.

  - Si no se considera redefinición, entonces se considera que lo que se desea es ocultar el método de la clase padre, de modo que para la clase hija sea como si nunca hubiese existido. El compilador avisará de esta decisión a través de un mensaje de aviso que puede suprimirse incluyendo el modificador **new** en la definición del miembro en la clase hija para así indicarle explícitamente la intención de ocultación.

- **Eficiente:** En principio, en C# todo el código incluye numerosas restricciones para asegurar su seguridad y no permite el uso de punteros.En C# es posible saltarse dichas restricciones manipulando objetos a través de punteros. Para ello basta marcar regiones de código como inseguras (modificador **unsafe**) y podrán usarse en ellas punteros de forma similar a cómo se hace en C++, lo que puede resultar vital para situaciones donde se necesite una eficiencia y velocidad procesamiento muy grandes.

- **Compatible:** Para facilitar la migración de programadores, C# no sólo mantiene una sintaxis muy similar a C, C++ o Java que permite incluir directamente en código escrito en C# fragmentos de código escrito en estos lenguajes, sino que el CLR también ofrece, a través de los llamados **Platform Invocation Services** (**PInvoke**), la posibilidad de acceder a código nativo escrito como funciones sueltas no orientadas a objetos tales como las DLLs de la API Win32. Nótese que la capacidad de usar punteros en código inseguro permite que se pueda acceder con facilidad a este tipo de funciones, ya que éstas muchas veces esperan recibir o devuelven punteros.

También es posible acceder desde código escrito en C# a objetos COM. Para facilitar esto, el *.NET Framework SDK* incluye una herramientas llamadas **tlbimp** y **regasm** mediante las que es posible generar automáticamente clases proxy que permitan, respectivamente, usar objetos COM desde .NET como si de objetos .NET se tratase y registrar objetos .NET para su uso desde COM.

3. Visual Studio.NET 2003

Visual Studio.Net 2003 es una completa herramienta para crear aplicaciones basadas en Microsoft.NET para Microsoft Windows, Web y dispositivos con compatibilidad nativa con Microsoft.NET Compact Framework y compatibilidad inherente con servicios Web XML.

Sus principales características son:

- **Variedad de lenguajes**

  Eficaces e interoperables como Visual Basic.NET, Visual C++.NET, Visual C#.NEt y Visual J#.NET

- **Software profesional para Windows, Web y dispositivos**

El diseño visual de formularios agiliza la creación de aplicaciones de escritorio completas para Windows, aplicaciones Web y aplicaciones para gran variedad de dispositivos.

- **Rápido desarrollo para los niveles de servidor y datos**

  El Diseñador de componenetes y el Explorador de servidores trabajan unidos para permitir la composición visual de componentes lógicos empresariales de nivel medio.

ADO.NET y Visual Database Tools permiten crear software profesional controlados por datos.

- **Implementación y mantenimiento de aplicaciones simplificados**

La implementación “sin tocar” permite distribuir aplicaciones basadas en Windows con la facilidad de las aplicaciones Web, mientras que la implementación de aplicaciones en paralelo reduce los problemes de versiones de las DLL. La compatibilidad integrada con la tecnología Windows Installer proporciona opciones avanzadas para crear paquetes de implementación para Windows y Web.

- **Confiabilidad y seguridad**

  Basado en la plataforma probada de .NET Framework, Visual Studio.NET utiliza una directiva de seguridad detallada para modelos de seguridad de acceso del código, basados en funciones y en usuarios.

- **Compatibilidad con versiones existentes**

  La actualización sin problemas de Visual Studio.NET 2002, la interopelabilidad con software existente basado en COM y la tecnología mejorada de actualización de Visual Basic garantizan el aprovechamiento de versiones existentes de proyectos.

- **Desarrollo para dispositivos inteligentes**

  La compatibilidad nativa con .NET Compact Framework permite desarrollar, depurar e implementar automáticamente aplicaciones en dispositivos inteligentes, incluidos dispositivos que utilicen Microsoft Windows CE .NET y Pocket PC. Un sólido emulador garantiza el desarrollo rápido y preciso de aplicaciones para dispositivos móviles sin la necesidad de tener el dispositivo.

- **Desarrollo de aplicaciones Web para dispositivos móviles**

  La compatibilidad con dispositivos inalámbricos permite ampliar fácilmente aplicaciones Web nuevas o existentes . Los controles de ASP:NET para dispositivos móviles representan de manera inteligente una amplia gama de dispositivos, liberando así a los programadores de preocupaciones sobre las funciones exclusivas de cada dispositivo.

El proyecto que nos ocupa ha sido desarrollado con Visual Studio.NET 2003 y Microsoft .NET Framework SDK 1.1 aunque actualmente ya existen en el mercado nuevas versiones publicadas por Microsoft Visual Studio 2005 Beta 2 y .NET Framework 2.0 Beta 2.

3.1 Visual C#.NET

Visual C#.NET forma parte de Visual Studio.NET 2003, es una herramienta y un lenguaje moderno e innovador que permite generar software conectado a .NET para Microsoft Windows, Web y una amplia gama de servicios.

A pesar de que el paquete ofrece una amplia gama de posibilidades y lenguajes, únicamente nos centraremos en proporcionar una breve explicación de las utilidades que hemos usado para la realización de la aplicación.

No pretende ser un manual exhaustivo sino una breve explicación de las herramientas mas comunes que hemos utilizado para la realización de la aplicación.

- **Empezar a trabajar con Visual Studio**

Una pantalla similar a la mostrada en la figura se presenta al arrancar Visual Studio:

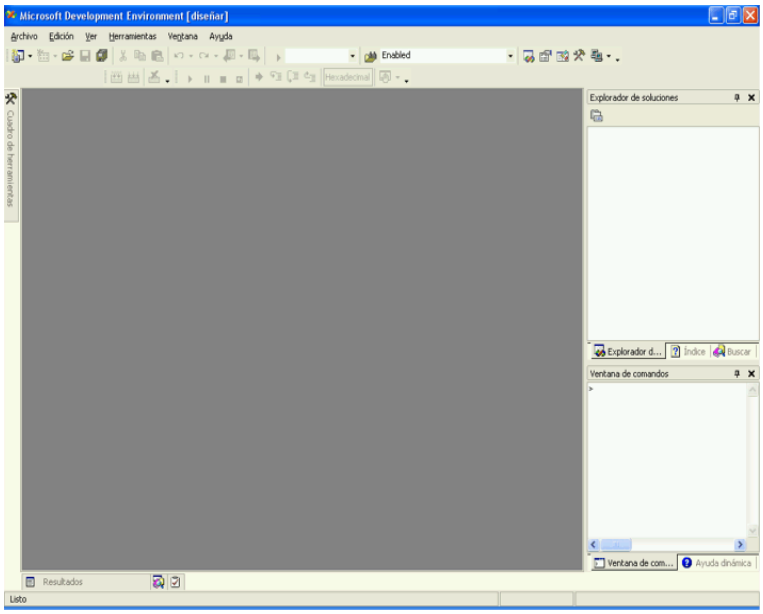


Figura 3 . Pantalla Inicial de Visual Studio.NET

Eligiendo la opción de menú Archivo Nuevo Proyecto se presenta la pantalla siguiente:

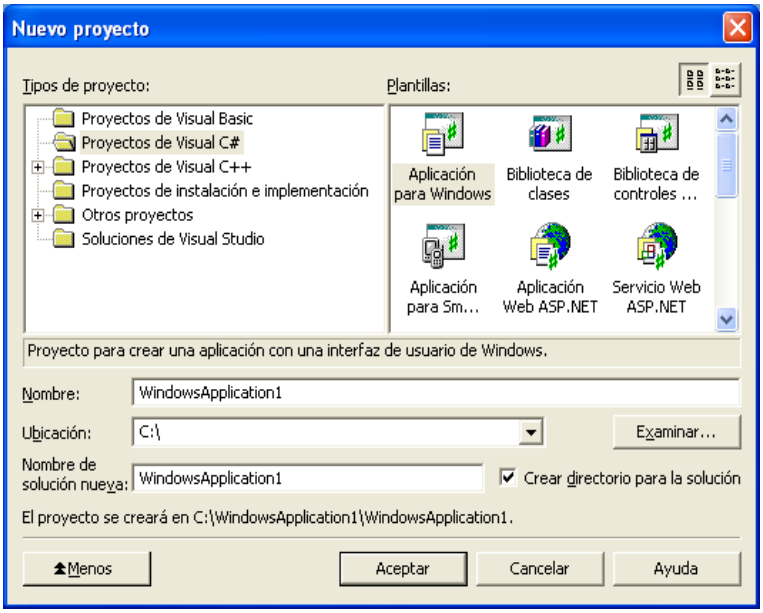


Figura 4
Pantalla Nuevo Proyecto

En esta pantalla de las posibles opciones elegiremos Aplicación para Windows, destacar que podemos elegir nombre y ubicación para nuestro proyecto, al pulsar en Aceptar se mostrará la pantalla siguiente, con la que podemos empezar a trabajar:

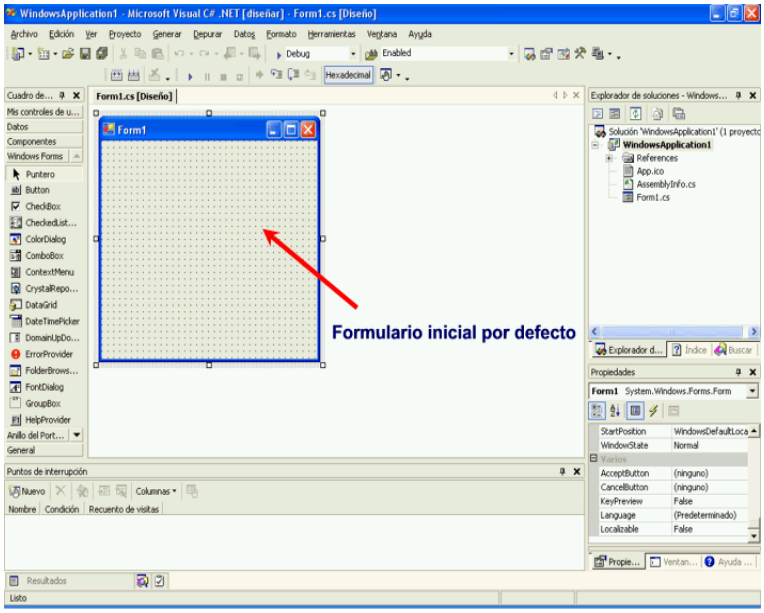


Figura 5. Pantalla de Aplicación Windows Form

- **¿Qué son los formularios de Windows?**

Los formularios de Windows constituyen un marco de trabajo para la creación de aplicaciones cliente de Windows que utilizan Common Language Runtime. Las aplicaciones de los formularios de Windows se pueden escribir en cualquiera de los idiomas compatibles con Common Language Runtime. Algunas de las ventajas de utilizar los formularios de Windows son las siguientes:

· **Simplicidad y potencia:** Los formularios de Windows constituyen un modelo de programación para el desarrollo de aplicaciones de Windows que combinan la simplicidad del modelo de programación de Visual Basic 6.0 con la potencia y la flexibilidad de Common Language Runtime.

· **Menor coste total de propiedad:** Los formularios de Windows se aprovechan de las características de implementación y de versión de Common Language Runtime con el fin de ofrecer costes de implementación reducidos y mayor solidez de las aplicaciones. Esto reduce significativamente los costes de mantenimiento (TCO) de aquellas aplicaciones escritas en los formularios de Windows.

· **Arquitectura de los controles:** Los formularios de Windows ofrecen una arquitectura para los controles y los contenedores de controles de acuerdo con una implementación específica de las clases contenedoras y de los controles. Esto reduce significativamente las cuestiones de interoperabilidad entre los contenedores y los controles.

· **Seguridad:** Los formularios de Windows se aprovechan por completo de las características de seguridad de Common Language Runtime. Esto significa que se pueden utilizar los formularios de Windows para implementar todo desde un control que no es de confianza, que se ejecuta en el explorador, a una aplicación de plena confianza instalada en el disco duro del usuario.

· **Compatibilidad con los servicios Web de XML:** Los formularios de Windows ofrecen una completa compatibilidad para establecer conexiones con servicios Web de XML de forma rápida y sencilla.

· **Gráficos con formato enriquecido:** Los formularios de Windows constituyen el primer vehículo de transmisión de GDI+, una nueva versión de la interfaz de dispositivo gráfico (GDI) de Windows que es compatible con la mezcla alfa, los pinceles de textura, las transformaciones avanzadas y el soporte de texto enriquecido, entre otros.

· **Controles flexibles:** Los formularios de Windows ofrecen un conjunto de controles enriquecido que engloba a todos los controles ofrecidos por Windows. Estos controles también ofrecen nuevas características, como estilos de "aspecto liso" para botones, botones de radio y casillas de verificación.

· **Conocimiento de los datos:** Los formularios de Windows ofrecen plena compatibilidad con el modelo de datos de ADO.NET.

· **Compatibilidad con controles ActiveX:** Los formularios de Windows ofrecen plena compatibilidad con los controles ActiveX. Es posible albergar fácilmente a controles ActiveX en una aplicación de formularios de Windows. También es posible albergar a controles de formularios de Windows como si se tratara de controles ActiveX.

· **Licencia:** Los formularios de Windows se aprovechan del modelo de licencia mejorado de Common Language Runtime.

· **Impresión:** Los formularios de Windows ofrecen un marco de trabajo de impresión que facilita que las aplicaciones proporcionen informes completos.

· **Accesibilidad:** Los controles de los formularios de Windows implementan interfaces definidas por Microsoft Active Accessibility (MSAA), lo que simplifica la creación de aplicaciones que sean compatibles con las ayudas de accesibilidad, como lectores de pantalla.

· **Compatibilidad en tiempo de diseño:** Los formularios de Windows se aprovechan plenamente de las características de los modelos de componentes y metadatos ofrecidas por Common Language Runtime, con el fin de proporcionar compatibilidad mediante el tiempo de diseño tanto a usuarios de controles como a implementadores de controles.

• Modelo de Aplicación de Formularios de Windows

El modelo de programación de aplicaciones para formularios de Windows se compone principalmente de formularios, controles y sus correspondientes eventos.

Formularios

En los formularios de Windows, la clase **Form** es una representación de cualquier ventana mostrada en la aplicación. Es posible utilizar la propiedad **BorderStyle** de la clase **Form** para crear ventanas flotantes, sin bordes, de herramientas y estándar. También se puede utilizar la clase **Form** para crear ventanas modales, como cuadros de diálogo. Hay un tipo especial de formulario, MDI form, que se puede crear estableciendo la propiedad **MDIContainer** de la clase **Form**. Un formulario MDI puede contener otros formularios, conocidos como formularios MDI secundarios, dentro de su área de cliente. La clase **Form** proporciona soporte integrado para el control del teclado (orden de tabulación) y desplazamiento del contenido del formulario.
Al diseñar la interfaz de usuario de una aplicación, normalmente se crea una clase que se deriva de **Form**. A continuación, se pueden agregar controles, establecer propiedades, crear controladores de eventos y agregar lógica de programación al formulario en cuestión.

Controles

Cada componente que se agrega a un formulario, como **Button**, **TextBox** o **RadioButton**, se denomina control. Los formularios de Windows incluyen todos los controles que normalmente se encuentran asociados a Windows, así como los controles personalizados como **DataGrid** de los formularios de Windows.
Normalmente se interactúa con controles estableciendo propiedades para alterar su aspecto y comportamiento.Los formularios proporcionan restricciones limitadas cuando se pueden establecer propiedades en lo que se refiere a controles.
Los formularios de Windows aseguran que el código que se crea es válido. Por ejemplo,si se establece una propiedad que a su vez establece un bit de estilo de Windows para un control de Windows, que sólo se puede establecer cuando el control se encuentre creado, el control de formularios de Windows descarta el control de Windows subyacente y crea un nuevo control.

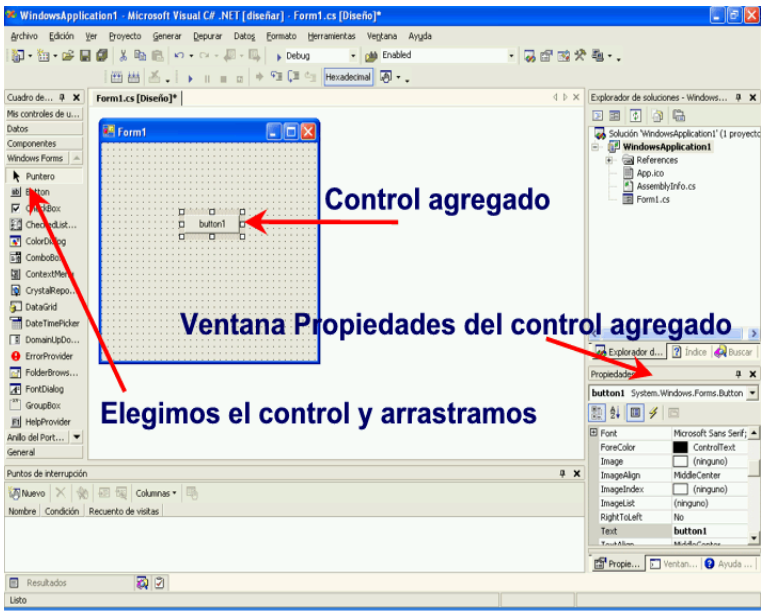


Figura 6 Agregar Controles a un Form

Los formularios de Windows contienen los siguientes controles:

Button	CheckBox	CheckedListBox	ColorDialog
ComboBox	ContextMenu	DataGrid	DateTimePicker
DomainUpDown	FontDialog	GroupBox	HelpProvider
HScrollBar	ImageList	Label	LinkLabel
ListBox	ListView	MainMenu	MonthCalendar
NumericUpDown	OpenFileDialog	PageSetupDialog	Panel
PictureBox	PrintDialog	PrintPreviewControl	PrintPreviewDialog
ProgressBar	PropertyGrid	RadioButton	RichTextBox
SaveFileDialog	Splitter	StatusBar	TabControl
TextBox	Timer	ToolBar	ToolTip
TrackBar	TrayIcon	TreeView	VScrollBar
ErrorProvider	DateTimeFormat	NumericFormat	

Tabla 3 Controles

No nos detendremos en explicar para que sirve cada uno de estos controles ya que Visual Studio proporciona información suficiente, y siempre nos queda la opción de consultar la *Ayuda Dinámica* que nos proporcionará todo tipo de información y ejemplos de uso.

- ## Eventos

El modelo de programación de los formularios de Windows está basado en eventos. Cuando un control cambia de estado, así como cuando un usuario hace clic en un botón, se provoca un evento. Para controlar un evento, la aplicación en cuestión registra un método de control de eventos para ese evento.

Se llama a un método de control de eventos sólo cuando tiene lugar un evento específico de un control determinado. Esto permite evitar tener un solo método en el formulario que controle todos los eventos de todos los controles. Esta característica también hace que el código sea más fácil de entender y mantener. Además, debido a que la arquitectura de eventos de los formularios de Windows está basada en delegados, los métodos de control de eventos tienen seguridad de tipos y se pueden declarar como privados. Esta capacidad hace posible que el compilador pueda detectar la falta de coincidencia en la firma del método durante la compilación. También mantiene la interfaz pública de la clase **Form** despejada de métodos de control de eventos públicos.
Clases de eventos
Cada evento tiene dos clases compatibles:

- Clase de delegado **EventHandler** utilizada para registrar el método de control de eventos. Firma de **EventHandler** que dicta la firma del método de control de eventos.

- Clase **EventArgs** que contiene datos acerca del evento provocado.

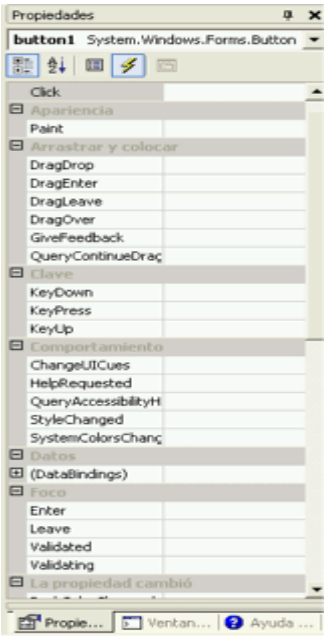
La firma de un **EventHandler** consiste en que el primer argumento contiene una referencia al objeto que provocó el evento (emisor) y en que el segundo argumento contiene datos acerca del evento (instancia de **EventArgs**). Por ejemplo, el evento **Click** de un **Button** utiliza el siguiente controlador de eventos.

```
public delegate void EventHandler(object sender, EventArgs e);
```

Como resultado, cualquier método de control de eventos del evento **Click** debe tener la siguiente firma.

```
<access> void <name>(object sender, EventArgs evArgs)
```

Existen varios eventos que utilizan clases **EventHandler** e **EventArgs** genéricas. Sin embargo, algunos eventos necesitan información adicional que es específica del tipo del evento provocado. Por ejemplo, los eventos relacionados con el movimiento del mouse (ratón) incluyen información acerca de la posición del puntero del ratón y de los botones del ratón. Estos eventos definen sus propias clases que se deben heredar de las clases **EventHandler** y **EventArgs**. Por ejemplo, el evento **MouseDown** utiliza las clases **MouseEventHandler** y **MouseEventArgs**.



Podemos consultar los eventos posibles que podemos utilizar con los controles en la ventana de propiedades de cada control

Figura 7. Ventana Propiedades del Control

- **Duración determinada y eliminación**

El modelo de clase de .NET Framework proporciona el método **Dispose** de la clase **Component**. Se llama al método **Dispose** cuando un componente determinado deja de ser necesario. Por ejemplo, los formularios de Windows llaman al método **Dispose** que se encuentra en un formulario y a todos los controles que se encuentran en ese formulario, cuando se cierra el formulario en cuestión. Normalmente, se utiliza **Dispose** para liberar grandes recursos de manera puntual y para quitar referencias a otros objetos, de manera que se puedan recuperar gracias al recolector de elementos no utilizados. También se le suele llamar para detener cualquier lógica de programa en ejecución asociada al formulario. Debería mantenerse el código del método **Dispose** tan sencillo y estable como fuera

posible. Si se produce un error en el método **Dispose**, es probable que no se puedan liberar de la memoria los recursos más grandes.

▪ Formularios de Windows y gráficos

Common Language Runtime aprovecha la versión avanzada de la interfaz de dispositivo gráfico (GDI) de Windows denominada GDI+. GDI+ está diseñada para proporcionar un alto rendimiento y facilidad de uso. Admite gráficos en 2-D, tipografía e imágenes. Las clases de GDI+ residen en los espacios de nombres **System.Drawing**, **System.Drawing.Drawing2D**, **System.Drawing.Imaging** y **System.Drawing.Text**. Los espacios de nombres se encuentran en el **System.Drawing.DLL** de ensamblado.
La clase **Graphics** representa una superficie de dibujo de GDI+. Para utilizar GDI+, primero es necesaria una referencia a un objeto Graphics. Normalmente, se obtiene una referencia a un objeto Graphics en el evento **Paint** de un control o formulario, o en el evento **PrintPage** de un **PrintDocument.**
Tras crear un objeto **Graphics**, se puede utilizar para dibujar líneas, rellenar formas y dibujar texto, entre otras cosas. Los objetos mayores que se utilizan junto con el objeto **Graphics** son los siguientes.

Pincel	Se utiliza para rellenar superficies delimitadas con esquemas, colores o mapas de bits.
Lápiz	Se utiliza para dibujar líneas y polígonos, junto con rectángulos, arcos y gráficos.
Fuente	Se utiliza para describir la fuente que se usa para procesar textos.
Color	Se utiliza con el fin de describir el color que se usa para procesar un objeto determinado. En GDI+, el color puede tener una mezcla alfa.

GDI+ es totalmente compatible con una amplia gama de formatos de imagen, como archivos .jpeg, .png, .gif, .bmp, .tiff, .exif y .icon.

Las clases **Pen** y **Brush** incluyen un conjunto de pinceles y lápices sólidos estándar de todos los colores

▪ Imprimir con formularios de Windows

PrintDocument	Se utiliza **PrintDocument** para enviar resultados a una impresora. Hay que crear una instancia de **PrintDocument**, establecer algunas propiedades que describan lo que hay que imprimir y llamar al método **Print**. **PrintDocument** provoca un evento **PrintPage** para cada página que se va a imprimir. Hay que agregar una determinada lógica de impresión a un controlador de eventos para este evento.
PrinterSettings	Información sobre cómo se debe imprimir un documento. Incluye la impresora donde se debe imprimir.
PageSettings	Información sobre cómo se debe imprimir una página
PrintPageEventArgs	Datos del evento **PrintPage** en **PrintDocument**. Proporciona un rectángulo de recorte y un objeto **Graphics** para la superficie de impresión.
PrintEventArgs	Datos de los eventos **BeginPrint** y **EndPrint** en **PrintDocument**. Permite cancelar el trabajo de impresión.
PrintDialog	Cuadro de diálogo de selección de impresora. Engloba a la API de *PrintDlg* de Win32.
PageSetupDialog	Cuadro de diálogo de propiedades de página. Engloba a la API de *PageSetupDlg* de Win32.
PrintPreviewControl	Control que muestra **PrintDocument**. Permite la creación de un cuadro de diálogo de vista preliminar.
PrintPreviewDialog	Cuadro de diálogo que muestra **PrintDocument** utilizando **PrintPreviewControl**.
PrintController	**PrintController** controla cómo imprimir **PrintDocument**. **PrintDocument.Print** utiliza un control de impresión para procesar el documento. .NET Framework proporciona dos controladores de impresión: ● **DefaultPrintController** envía a una impresora. ● **PreviewPrintController** envía a **PrintPreviewControl**. Normalmente, nunca es necesario implementar **PrintController**. Sólo es necesario implementar **PrintController** si se desea enviar a otro destino.

la lógica del controlador de eventos **PrintPage** será la siguiente:

· Hay que imprimir el contenido de la página utilizando la información de los argumentos de evento. Los argumentos de evento contienen**Graphics** para la impresora, **PageSettings** para la página, los bordes de la página y el tamaño de los márgenes.

· Hay que determinar si existen más páginas para imprimir.

· En caso de que haya más, hay que establecer **HasMorePages** en **true**.

· En caso de que no haya más, hay que establecer **HasMorePages** en **false**.

Mostrar una ventana de vista preliminar
La ventana Vista preliminar permite al usuario obtener una vista previa del documento antes de imprimirlo. Es posible agregar una ventana de vista preliminar a la aplicación en cuestión creando un **PrintDocument** y pasándolo al cuadro de diálogo **PrintPreview**.

4. Un poco sobre XML

XML (Lenguaje de marcado extensible), es un lenguaje utilizado para escribir datos. El objetivo de XML consiste en proporcionar un formato estándar que puedan leer, procesar y escribir diferentes aplicaciones que se ejecutan en hardware diferente. XML ha adquirido importancia creciente como formato estándar para el intercambio de datos.

▪ Objetivos de XML

En 1996, el Word Wide Web Consortium (conocido como W3C) emprendió la tarea de diseñar un formato de datos estándar. Perseguía varios objetivos. Uno de ellos era que este formato debía ser capaz de representar cualquier forma de datos estructurados. Esto significa que, además de especificar los propios datos, XML debía indicar también cómo se organizan de forma clara y sin ambigüedades. Otro de los objetivos consistía en garantizar que el formato fuera portable y utilizable en Internet. Esto era importante debido a que podrían existir multitud de aplicaciones con la necesidad de procesar datos, ubicadas en cualquier parte. Además el formato debía hacer que los desarrolladores les resultara tan sencillo como fuera posible escribir programas que consumieran y produjeran XML, ya que si fuera complejo nadie desearía usarlo.

▪ Estructura de XML

Para cumplir los requisitos de portabilidad y apertura, XML utiliza texto normal para representar datos incrustados en etiquetas que describen la estructura de los datos.
Las **etiquetas** XML son los elementos encerrados entre paréntesis angulares (<, >).
El primer elemento (<?xml version=”1.0”> que aparecerá en nuestro fichero XML indica la versión de XML a la que es conforme el documento, el resto del documento contiene los datos, como vemos en este recorte de código

```
<?xml version="1.0" encoding="utf-8" ?>
<menu xmlns="./config_palabras.xsd">
<palabras imagen="./imagenes/armarioropa.png" nombre="ROPA"
label="Escribe aquí la prenda de ropa">
 <palabra nombre="bufanda" imagen="./imagenes/ropa/bufanda.wmf"
sonido="./sonidos/ropa/bufanda.wav" />
```

Un documento XML válido y bien formado debe ajustarse a una serie de criterios. Sólo debe existir un único elemento raíz actuando de contenedor para todos los datos , el cual contiene los elementos individuales como subelementos anidados. Para cada elemento “inicio” *<identificador>*, debe existir el correspondiente elemento fin denotado como *</identificador>* indica el final de los datos

correspondientes a ese elemento.
El sangrado y los saltos de línea no son significativos en XML pero se pueden utilizar para leer el documento con mayor facilidad.

- ## Esquemas XML

Un esquema describe la estructura de un documento XML y puede incluir información adicional como reglas y comprobaciones de validación. Por ejemplo una regla que prohíba valores negativos en un determinado item.
Para que dos aplicaciones puedan utilizar el mismo archivo XML debe utilizarse el mismo esquema. Muchas organizaciones están desarrollando esquemas estándar para describir requisitos de datos comunes para diversas industrias, permitiendo la libre interoperatividad de las aplicaciones que se basan en dichos esquemas.

- ## API de XML y .NET Framework

Anteriormente se ha indicado que uno de los objetivos del W3C (Word Wide Web Consortium) consistía en diseñar un formato, XML, que fuera fácil de programar y utilizar. Se han desarrollado diversas API para manejar XML; las dos mas comunes son el Modelo de objetos de documentos (DOM, Document Object Model) y la API simple para XML (SAX, Simple API for XML). DOM es el resultado del trabajo del W3C (SAX no lo es); ha sufrido varias revisiones y sigue actualizándose a medida que la tecnología y los requisitos evolucionan.
Microsoft ha adoptado muchas de las características de programación del DOM y las ha expuesto a través de la clase *XmlDocument* del espacio de nombres *System.XML* en la biblioteca de clases .NET Framework, también ha agregado algunas extensiones a la funcionalidad de la implementación.

- ## XML y Visual Studio

La filosofía es similar al la creación de formularios Windows, si ya hemos creado un proyecto como ya hemos mencionado, debemos *agregar un nuevo elemento* o uno ya existente en su caso en la opción de menú *Proyecto*, como muestra la figura

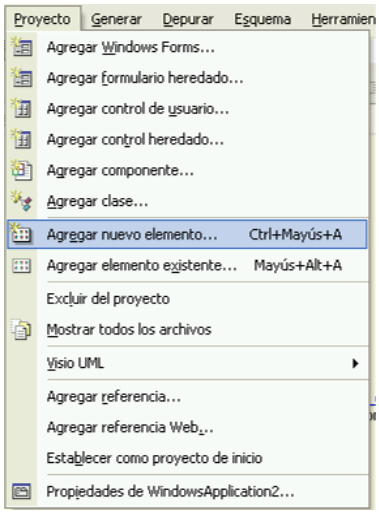


Figura 8 Agregar un nuevo elemento

Se presenta una pantalla como la mostrada en la figura elegiremos *Esquema XML* y el nombre que deseemos darle a dicho esquema

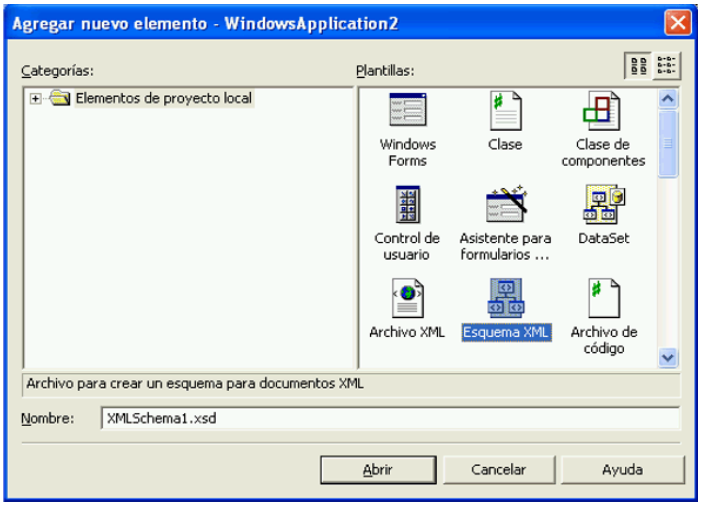


Figura 9 Agregar esquema XML

Al abrir el fichero creado podremos empezar a añadir elementos al esquema como se muestra en la figura siguiente arrastrándo del cuadro de herramientas el elemento que queramos incluir en el esquema.

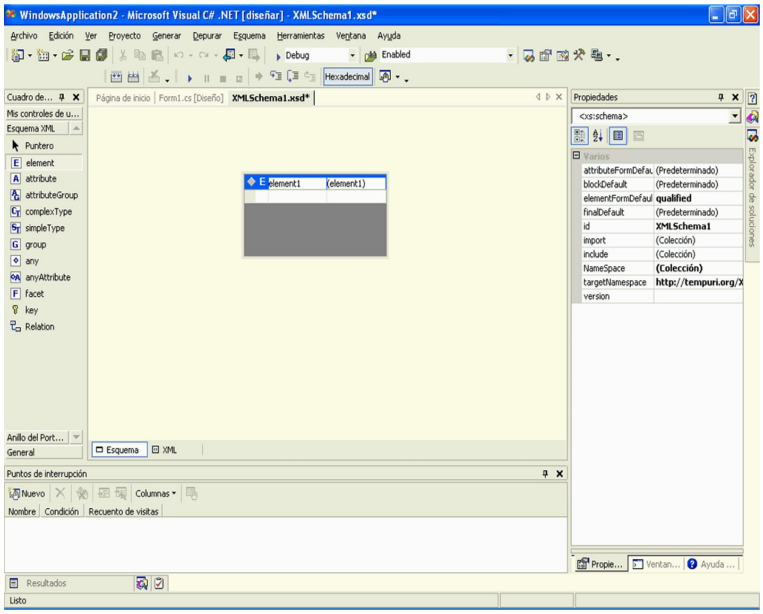


Figura 10 Agregar elementos al esquema XML

De forma análogo se agrega al proyecto un archivo XML repitiendo los pasos anteriormente descritos pero para el elemento archivo XML.

Apéndice B. Contenidos del CD-Rom

Los contenidos del CD-ROM que acompaña esta documentación son:

- **Directorio de Instalación:**
  - **Subdirectorio Instalación Requisitos previos**
    Contiene los archivos necesarios para la instalación del Tablet PC (SDK) v. 1.7 y del Módulo de reconocimiento de escritura para Microsoft Windows XP Tablet PC Edition **TRSetup.exe**
  - **Subdirectorio Instalación Aplicación**
    Contiene la carpeta Tableta Gráfica dentro de la cual se encuentra la carpeta Escribe, yo leo_Setup, ésta contiene la carpeta Debug y dentro de la misma se encuentra el archivo de **Setup.exe** de instalación de la aplicación.
- **Directorio Memoria:** Contiene la memoria en formato PDF.
- **Directorio Manual de Usuario:** Contiene el manual de usuario en formato PDF.

Apéndice C. Glosario de Términos

API	Interfaz de Programación de Aplicaciones
SDK	Kit del Desarrollo del Software
GAC	Caché de Ensamblado Global
DLL	Librería de enlace Dinámico (dynamic-link library)
PDD	Trastorno Generalizado del Desarrollo
AD/HD	Desorden deficitario de la atención / Hiperactividad
AENOR	Asociación Española de Normalización y Certificación
GIRO	Grupo de Investigación en Reutilización y Orientación a Objeto
PDA	Asistente Personal Digital
UML	Lenguaje de modelado unificado
MUI	Interfaz de Usuario Multilingüe
ISF	Formato serializado de tinta
SAPI	Interfaz de programación de aplicaciones de voz (Speech)