UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

APLICACION DE RECONOCIMIENTO DE VOZ A COMANDOS DE WINDOWS Y WORD

T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACION

P R E S E N T A :

JAIME ALFONSO REYES CORTES

DIRECTOR DE TESIS

DR. JOSE ABEL HERRERA CAMACHO

CIUDAD UNIVERSITARIA MEXICO, D. F. 2004

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

APLICACIÓN DE RECONOCIMIENTO DE VOZ A COMANDOS DE WINDOWS Y WORD

T E S I S

QUE PARA OBTENER EL TÍTULO DE

INGENIERO EN COMPUTACIÓN

P R E S E N T A :

JAIME ALFONSO REYES CORTÉS

DIRECTOR DE TESIS: DR. JOSÉ ABEL HERRERA CAMACHO

CIUDAD UNIVERSITARIA **MÉXICO D. F. 2004**

Dedicatoria

A mi madre Lic. María M. Cortés Cedillo, por su sabiduría, su fortaleza, sus enseñanzas y su constante afán de superación

A mi padre Lic. Crescencio Reyes Soto, por su elocuencia, su cultura y por enseñarme a discernir entre el arte y la vida

A mis Q.·. H.·. Héctor Omar y Aldo Reyes Cortés por su particular forma de apoyarme siempre

Al Director de la Facultad de Ingeniería M.C. Gerardo José Ferrando Bravo por su desinteresada e incasable lucha en pro del engrandecimiento de nuestra alma mater

A todos mis maestros por otorgarme un poco de su sabiduría y motivarme para seguir aprendiendo

En especial al Dr. José Abel Herrera Camacho, por su paciencia, todo su apoyo y sus enseñanzas

A la Universidad Nacional Autónoma de México y a la Facultad de Ingeniería por abrirme sus puertas, enseñarme a ser, a crecer y a pensar por mí mismo

Paralelamente en una alegoría específica de mi vivencia personal, a la danza que, con su expresión, creación y movimiento, me ha otorgado un camino más de crecimiento y realización

A aquellos que han hecho de la autosuperación una forma de vida y que no se doblegan ante las vicisitudes de la vida

J.A.R.C.

Agradecimientos

A Inés Jiménez, por abrirme su corazón y brindarme su apoyo incondicional

A mis amigos Ángeles Ciprés, Alba Flores, Ángeles Hernández, Sergio Crescencio y Rodrigo Domínguez por su apoyo aún en la distancia y por el don de su amistad

A Omar Nieto, por sus consejos y su apoyo en la realización de este trabajo

A la DEPFI y al laboratorio de Procesamiento Digital de Voz por permitirme la utilización de sus instalaciones para la realización de esta tesis

A todos mis compañeros de facultad que tuvieron la oportunidad y el deseo de concluir sus estudios y continuar con sus vidas

A Javier Juárez y Lucina Wong y a los muchachos de la Escuela Mexicana de Baile Fino de Salón por todo su apoyo y por enseñarme que la danza brinda una posibilidad más de crecimiento personal

A la casa de cultura Profa. Virginia Poulat y a mis alumnos del Taller de baile de salón por permitirme realizarme en mi segunda alternativa de vida

Índice general

Índice de Figuras

Índice de tablas

INTRODUCCIÓN

De entre la diversidad de seres vivos que existen en la naturaleza, destaca el ser humano por poseer características, en cierta manera, especiales y bastante complejas, ya que es capaz de realizar actividades comunes a los otros seres vivos, como comer, dormir, respirar, etcétera y otras que no lo son, como utilizar el intelecto o la imaginación para resolver problemas complejos o crear nuevas ideas. Todo esto lo llevamos a cabo, a veces, sin siquiera ponernos a pensar en cómo es que llegamos a lograrlo. Se puede decir que son actividades inherentes a los seres humanos. Sin embargo, si tratamos de analizar cada una de ellas nos topamos con ciertas dificultades, ya que, aunque aparentemente sean procesos comunes y sencillos para nosotros, poseen todo un conjunto de etapas que a veces resultan sorprendentes por ser acciones bastante laboriosas y por llevarse a cabo en tan poco tiempo. Por ejemplo, cuando escuchamos se realiza toda una serie de eventos que se inician cuando las ondas sonoras entran por el oído y son conducidas a su interior; una vez ahí son procesadas por un conjunto de órganos especializados y posteriormente se envían al cerebro donde, finalmente, se les da sentido. Así es, en pocas palabras, como el ser humano es capaz de llevar a cabo el proceso de la audición. ¿Y qué decir del proceso del habla? Pues es también un proceso bastante complejo y además de suma importancia para los seres humanos ya que, en su mayoría, es nuestra forma de comunicarnos y entendernos.

Entonces, nos adentramos en el campo de la tecnología de voz o del tratamiento del habla. Tecnología surgida desde hace 30 años y que ha ido en constante desarrollo debido a que nos hemos empeñado en hacer que las máquinas también puedan escucharnos y entendernos con el fin de controlarlas y utilizarlas de la manera más rápida y eficiente posible. Así la tecnología de voz nos proporciona un medio de controlar e interactuar con las máquinas y, en específico, con la computadora personal (PC) y con sus aplicaciones.

Además, pensemos en los beneficios que trae consigo la tecnología de voz:

- Simplicidad y conveniencia.
- Permite que aquellas personas que no pueden hacer uso del teclado puedan utilizar los recursos que ofrece una PC.
- Soporte de voz para aplicaciones de Internet, que se utilizan cada vez más, y en general, para cualquier aplicación de la PC.
- Asimismo, ofrece soporte para nuevas aplicaciones controladas por voz.

En el campo de la computación, la tecnología de voz ofrece al usuario de la PC tres funciones primarias.

1. *Control y comandos*. Utiliza comandos de voz para navegar a través de los menús, barras de herramientas y cuadros de diálogo en las aplicaciones.
2. *Dictado*: Dictarle directamente a las aplicaciones por medio de un micrófono conectado a la PC, sin necesidad de utilizar el teclado o algún otro dispositivo de entrada.
3. *Texto a voz o TTS (Text to Speech),* por sus siglas en inglés. La computadora, literalmente lee el texto tecleado y lo sintetiza en una voz por medio de la tarjeta de sonido y lo dirige a un auricular o a los parlantes que usan una PC.

Los primeros dos puntos se engloban en la categoría de reconocimiento de voz y el último en la categoría de síntesis o generación de voz. El presente trabajo se encuentra acotado al control y comandos, y aunque ya existen bastantes software que realizan reconocimiento de voz en inglés, la meta de este sistema es que se realice uno que sea en español, y el objetivo es que permita controlar una parte de toda la gama de funciones que nos ofrece el sistema operativo Microsoft Windows y una de sus aplicaciones de uso más frecuente, el procesador de textos Microsoft Word.

De manera general, en el capítulo 1, *Sistema de reconocimiento de palabras aisladas,* se tratan los principios teóricos en los que se basa el sistema de reconocimiento de voz desarrollado. Se da una breve descripción de la naturaleza de la voz y cómo se produce. Además, se presentan los fundamentos de la técnica de cuantización vectorial para reconocer patrones específicos de voz.

Por otra parte, en el capítulo 2, *Aspectos básicos del software Windows y Word,* se presenta una breve introducción a los componentes principales del sistema operativo Microsoft Windows: La API de Windows, de la que se habla someramente de sus categorías y de los servicios que proporciona para la realización del presente trabajo, y el modelo de componentes de objetos (COM), del que se hace una descripción general de su estructura. Asimismo, en este capítulo, se presenta una introducción al mecanismo de automatización y el uso que se la da para controlar Microsoft Word.

Una vez comprendidos los conceptos sobre la API de Windows, el modelo COM y el mecanismo de automatización, en el capítulo 3, *El shell de Windows,* se observa como se

aplican dichos conceptos en el shell de Windows y en su estructura. Veremos la manera en que interactúa el Explorador de Windows y cómo es que realiza las operaciones básicas con archivos y directorios, así como la ejecución de aplicaciones. Se da, también, una breve descripción de los tipos de extensiones del shell que existen y la manera de automatizar el Explorador de Windows.

En el capítulo 4, *Comandos de voz,* se hace una descripción del sistema ***Comandos de voz*** desarrollado. Se explica por medio de esquemas en qué consiste cada módulo de manera general y se utilizan ilustraciones para mostrar los campos y los parámetros de configuración con que cuenta este sistema. Asimismo se hace un esbozo de la funcionalidad que posee y se muestran los comandos de voz implantados.

En el capítulo 5, *Pruebas y resultados,* se planten las pruebas realizadas y se presentan los resultados obtenidos para el sistema en forma de tablas tanto para el entrenamiento como para el reconocimiento y para los equipos involucrados. Además se hace una descripción de los resultados de la ejecución de los comandos de voz.

En el capítulo 6, *Conclusiones*, además de hacer un resumen de las conclusiones se presentan los requerimientos mínimos que necesita el sistema ***Comandos de voz*** para ejecutarse. Asimismo se proponen mejoras y posibles aplicaciones de este sistema.

Por último, en los apéndices *A* y *B*, se incluyen someramente los entornos de usuario de Windows y de Word, respectivamente, para poder entender cómo son manejados desde el punto de vista del usuario.

Capítulo 1

Sistema de reconocimiento de palabras aisladas

1.1 El reconocimiento automático de voz

En el reconocimiento de voz existen varios campos de estudio, como son los sistemas de reconocimiento de palabras aisladas independientes del locutor, los de reconocimiento de voz dependientes del locutor, verificación de locutor, reconocimiento de palabras continuas, traducción, análisis sintáctico y semántico, algoritmos de codificación y análisis de voz.

El presente trabajo se encuentra acotado al reconocimiento de palabras aisladas dependiente del locutor y también al campo del análisis de voz. Es de palabras aisladas ya que en la pronunciación de las mismas se llevan a cabo pequeñas pausas entre una y otra de tal forma que el procesamiento se realiza basado en un vocabulario específico. Es dependiente del locutor porque basta con "alimentar", por así decirlo, al sistema con las grabaciones de las palabras de la persona cuya voz se desea reconocer, en este caso la de un servidor, para que dicho sistema las procese y lleve a cabo el reconocimiento de las mismas. Y abarca el análisis de voz, específicamente la parametrización, porque después de aplicarles ciertos algoritmos a las señales de voz se obtienen datos en forma paramétrica.

Al proceso de 'alimentar' al sistema, junto con el proceso de obtención de parámetros, los cuáles se almacenan para su posterior utilización, se denomina entrenamiento.

Ahora bien dentro de los algoritmos que se utilizan para procesar las señales de voz se encuentra la cuantización vectorial, que es el pilar de este sistema de reconocimiento, ya que es través de ésta de donde se obtienen los patrones que sirven de referencia para realizar tal reconocimiento.

Posterior al entrenamiento, se pasa a la siguiente fase que es el reconocimiento en sí o la clasificación de los parámetros. Dada una señal de voz que se desea reconocer, se le procesa con los mismos algoritmos y se obtienen sus respectivos patrones, a estos últimos se les compara con los ya almacenados y aquel conjunto de patrones que tenga mayor similitud con los obtenidos será la palabra reconocida.

1.1.1 Problemas que presenta un sistema de reconocimiento de voz

Las señales de voz tienden a concatenarse, es decir, cuando nos comunicamos es difícil encontrar dónde se inicia una palabra y dónde termina dicha palabra y es difícil delimitar los sonidos que componen dicha palabra, aunque la experiencia ha demostrado que basta con una separación aproximada.

Las señales de voz son variables, es decir, dos señales de voz de la misma frase e incluso de la misma palabra resultan ser distintas aún cuando hayan sido pronunciadas por el mismo locutor.

Existe ambigüedad en el habla, esto es, existen palabras cuya pronunciación es igual, pero su significado es diferente (homófonas), además, una misma palabra puede tener varios significados dependiendo del contexto en el que se hable.

Presencia de ruido en las señales de voz, debido a que en el ambiente en que se trabaja existen ciertas perturbaciones que tienden a modificar las señales, haciéndolas difíciles de reconocer.

Complejidad en el habla, ésta varia según el acuerdo entre los individuos, la región en la que se habite, con quién se esté hablando e incluso el estado de ánimo en el que se encuentre el parlante, además de que solo es una parte del proceso de comunicación.

Un sistema de reconocimiento de voz tendrá éxito en la medida que resuelva los problemas planteados anteriormente.

1.2 Naturaleza de la señal vocal

La voz se obtiene por la acción conjunta de varias regiones de órganos:

Por una parte se encuentra el tracto pulmonar o respiratorio, formado por los pulmones y la tráquea; los pulmones generan aire comprimido que es conducido por la tráquea. Dichos órganos controlan la amplitud de los sonidos, y la única contribución audible del tracto pulmonar son los silencios inter y entre palabras.

Por otra, se encuentra la laringe, situada por encima de la tráquea y por debajo de la faringe. En la laringe se encuentran las cuerdas vocales en donde se producen los sonidos y éstas al cerrarse protegen al tracto pulmonar de la entrada de objetos y permiten la formación de presión dentro del tórax y el abdomen.

Por último, el tracto vocal, ubicado por encima de las cuerdas vocales, se encarga de la coloración y la articulación de la voz, principalmente por la lengua, los labios y la mandíbula baja. Asimismo contiene los puntos principales desde los cuáles los sonidos son radiados. A esta función se le llama modulación, ya que el sonido se modula en el tracto vocal con el fin de modificar la calidad del sonido e interponer sonidos adicionales e interrupciones a los sonidos.

1.2.1 Modelo digital para la voz

El análisis de la señal vocal se lleva a cabo mediante un modelo que describe el proceso del habla clasificando las señales en dos tipos:

Señales sonoras: Aquellas que se caracterizan por tener alta energía y su rango de frecuencias está entre 300 y 4000 Hz[1]. Son generadas por las cuerdas vocales y además presentan cierta periodicidad, como se muestra en la figura 1.1.

Fig. 1.1 Señal sonora.

Señales sordas: Aquellas que se caracterizan por tener baja energía y componente frecuencial uniforme presentando aleatoriedad en forma de ruido[2], como se muestra en la figura 1.2.

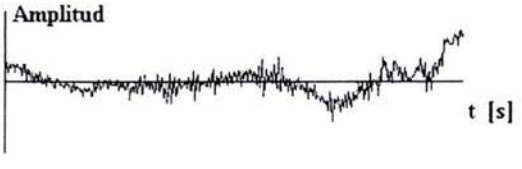


Fig. 1.2 Señal sorda.

[1] Andrés Flores Espinosa. Reconocimiento de palabras aisladas. http://www.alek.pucp.edu.pe/~dflores/tesis/naturaleza.html
[2] Idem.

Es así como la señal dividida en dos partes se modela, y dicho modelo incluye los cambios de la señal de excitación, la respuesta del tracto vocal y los efectos de los labios en la radiación como se ve en la figura 1.3.

El modelo se presenta entonces como un filtro variable en el tiempo que ha sido utilizado por casi todos los sistemas de procesamiento de voz. En este modelo los articuladores son modelados por filtros LTI usando el hecho de la independencia relativa entre la fuente y el tracto vocal, y de que los sonidos son estacionarios en intervalos cortos, aproximadamente de 10 a 20 milisegundos.

El filtro variable en el tiempo tiene dos posibles señales de entrada que dependerán del tipo de señal, sonora o sorda. Para señales sonoras la excitación será un tren de impulsos espaciado a frecuencias que sean iguales a la frecuencia del tono fundamental, mientras que para las señales sordas la excitación será ruido aleatorio. La combinación de estas señales modelan el funcionamiento de la glotis.

El espectro de frecuencias de la señal vocal puede obtenerse a partir del producto del espectro de la excitación por la repuesta en frecuencia del filtro. El tracto vocal manifiesta un número muy grande de resonancias, que toman el nombre de formantes, sin embargo, se consideran solo las tres o cuatro primeras y cubren un rango de frecuencias entre 100 y 3500 Hz. Esto debido a que las resonancias de alta frecuencia son atenuadas por el comportamiento del tracto que tiende a actuar como un filtro pasobajas con una caída de aproximadamente -12 dB por octava alrededor de 0.8 a 1 kHz.

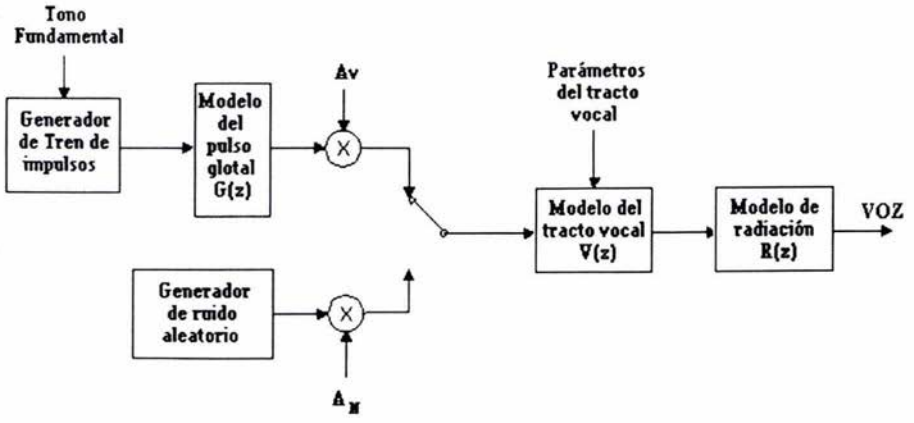


Fig. 1.3 Modelo digital para la voz.

1.3 Codificación de voz utilizando VQ-LPC-cepstral

En la figura 1.4 se presenta el esquema del sistema de reconocimiento de voz que utiliza coeficientes cepstral[3] y que se describe de manera sómera en las siguientes páginas. Este sistema es en el que me apoyo para realizar el reconocimiento de voz.

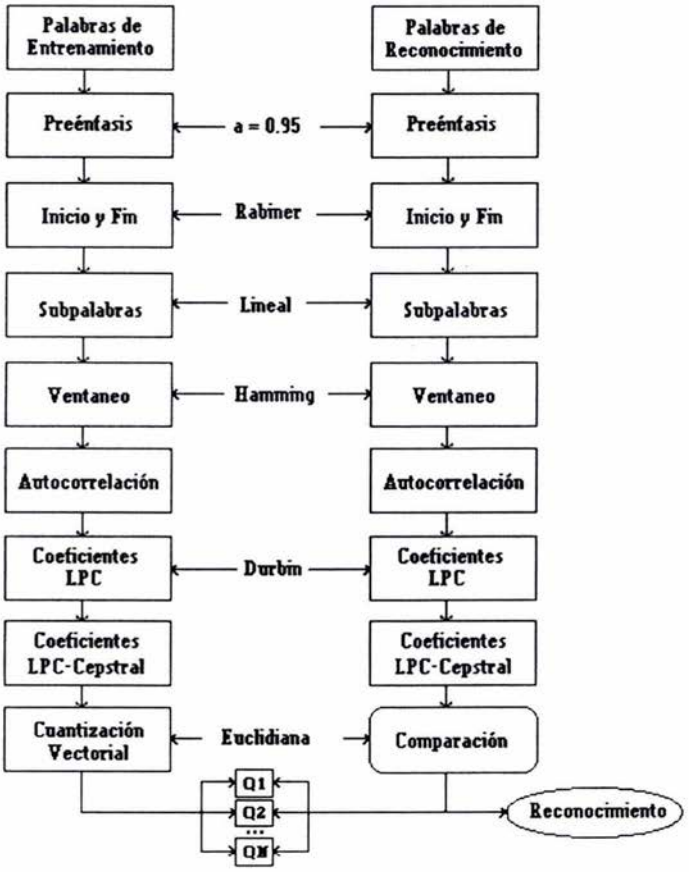


Fig. 1.4 Sistema de reconocimiento LPC-Cepstral.

[3] Tomado de Arturo Gardida Degollado. Reconocimiento de palabras aisladas utilizando cuantización vectorial. Tesis de Licenciatura. UNAM. FI, pág. 38.

1.3.1 Filtrado de preénfasis.

Del modelo digital para la voz se desprende que en el espectro de la voz existe una caída de -6 dB/octava, conforme la frecuencia aumenta. Esto se debe a la combinación de una caída de -12 dB/octava ocasionada por la fuente de excitación de la voz y un incremento de +6 dB/octava ocasionado por la radiación de la boca. Y significa que, cada vez que la frecuencia aumenta al doble, la amplitud de la señal se reduce en un factor de 16. Por lo que se desea compensar esta caída de -6 dB/octava realizando un preprocesamiento de la señal de voz que de un incremento de +6 dB/octava en el rango apropiado, de manera que la medición del espectro tenga un rango dinámico similar a lo largo de todo su ancho de banda. A éste proceso se le denomina preénfasis y para implementarlo se utiliza un filtro digital pasoaltas de primer orden, como se indica a continuación:

$$y[n] = x[n] - a\,x[n-1]$$

donde *y[n]* denota la salida del filtro de preénfasis de la muestra actual, *x[n]* es el valor de la muestra actual a la entrada del filtro, *x[n-1]* es la muestra de la entrada anterior y a es una constante que generalmente se elige entre 0.9 y 1. En este caso el valor de *a* es de 0.95.

1.3.2 Detección de inicio y fin de palabras aisladas

En la mayoría de los esquemas de procesamiento de voz se asume que las propiedades de la señal de voz varían relativamente poco con respecto al tiempo. Esta suposición conduce a una variedad de métodos de procesamiento en "tiempo corto" en los que se aislan y procesan pequeños segmentos de la señal de voz como si éstos fueran segmentos de un sonido significativo con propiedades fijas. Esto último se repite de manera periódica tanto como se desee. Frecuentemente estos pequeños segmentos, algunas veces llamados tramas, se traslapan unos sobre otros. El resultado del procesamiento de cada trama puede ser un escalar o un vector. Por tanto, este procesamiento produce una nueva secuencia dependiente del tiempo que puede servir como una representación de la señal de voz.

Uno de los métodos que cae en la descripción mencionada anteriormente se utiliza para la detección de inicio y fin de la palabra. Esto es, una vez realizado el preénfasis se procede a realizar la detección del inicio y el fin de cada palabra para eliminar el ruido de fondo que se produce durante la grabación de la señal y trabajar únicamente con aquellas muestras que representen a la señal de voz en si. Para realizar tal detección se trabaja con el algoritmo de

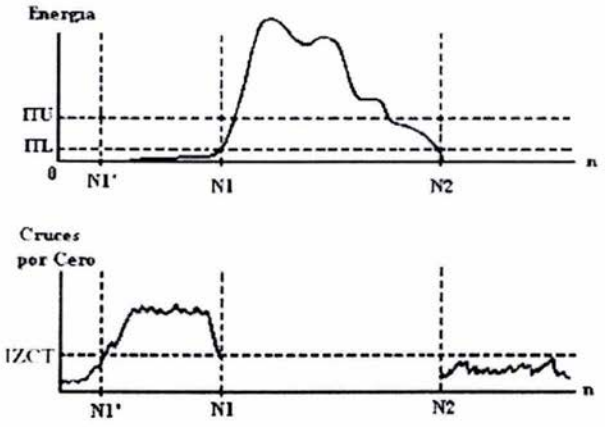


Fig. 1.5 Detección de inicio y fin de una señal de voz.

Rabiner-Sambur, que se basa en la combinación de dos mediciones realizadas en el dominio del tiempo: la magnitud promedio y la tasa de cruces por cero. La magnitud promedio refleja la variación de la amplitud de la señal y la tasa de cruces por cero mide el contenido frecuencial de la señal. Este algoritmo se puede resumir en la figura 1.5.

Como primer paso se obtienen, por cada trama de *n* muestras, la magnitud promedio de la señal y su tasa de cruces por cero. En este trabajo se toma *n* igual a 128 muestras. Se asume entonces que las 10 primeras y las 10 últimas tramas son de ruido para obtener una caracterización estadística del ruido de fondo. Utilizando esta estadística se calculan los umbrales que nos servirán para detectar el inicio y el fin. Se trabaja primero con el inicio de la señal. Se define un primer umbral **ITU** (umbral superior de energía) para obtener el intervalo en el cual la energía promedio de la señal siempre es excedido y se asume que el inicio y el final se encuentran fuera de este intervalo. Luego se verifican las tramas desde el punto donde se rebasó por primera vez el umbral **ITU** hacia el principio de la señal de voz, hasta un punto N1 donde la magnitud promedio cae por debajo de un umbral menor de energía, **ITL**, y que se propone tentativamente como el inicio de la palabra. Se sigue un procedimiento similar, partiendo desde el fin de la señal, para encontrar el punto tentativo N2. En esta etapa es razonablemente válido suponer que el inicio y fin no se encuentran dentro del intervalo de N1 a N2, por lo que el siguiente paso es regresar desde N1 (adelantar desde N2) hacia el inicio (final) de la señal y comparar la tasa de cruces por cero

con un umbral **IZCT** (umbral de cruces por cero). Esto se limita a las 25 tramas que preceden a N1 (siguen a N2). Y si la tasa de cruces por cero excede el umbral IZCT 3 o más veces, el punto de inicio N1 se recorre hasta el punto en donde el umbral se excedió por primera vez. De otra manera el inicio se escoge en el primer punto N1. Se sigue un procedimiento similar para determinar el final de la palabra[4].

1.3.3 Segmentación en subpalabras

Para poder llevar a cabo la cuantización vectorial de una forma eficiente y que todos los parámetros de las señales de voz sean comparados con los patrones de referencia el mismo número de veces, se requiere que las palabras sean divididas en subpalabras. Existen tres tipos de segmentación: *lineal, no lineal (KM)* y *acústica (MLR)*[5].

La *segmentación lineal* es la más sencilla y consiste en dividir a las señales de voz en N segmentos de igual longitud. La *segmentación KM* es un tipo de segmentación experimental basada completamente en el algoritmo de agrupamiento **K-medias**, éste algoritmo divide a la señal en segmentos de voz que tienen características similares. La *segmentación acústica* se realiza aplicando una metodología de clasificación de la razón de máxima verosimilitud (**MLR**) que detecta cambios espectrales.

En este trabajo se hace uso de la segmentación lineal con N igual a cuatro segmentos.

1.3.4 Segmentación de subpalabras en tramas y ventaneo

A la señal que se le realizó *el filtrado de preénfasis* y la *segmentación lineal* se le separa en bloques o tramas, como ya se mencionó, para su procesamiento.

A cada trama se la somete a una *ventana de Hamming*, que es la ventana que se usa comúnmente en el análisis de la señal de voz, con el objeto de suavizar la señal en los bordes de dicha ventana y así reducir la alteración de la respuesta en frecuencia que es provocada por la separación de la señal en tramas.

La ventana se define como

$$w(n) = 0.54 - 0.46\cos\left(2\pi n / (N-1)\right) \qquad 0 \le n \le N-1$$

y se presenta en la figura 1.6 para una ventana de 1200 muestras

[4] Tomado de Dr. Abel Herrera Camacho. Apuntes de Procesamiento Digital de Voz.
[5] Arturo Gardida Degollado. Opus citatum. Pág.29

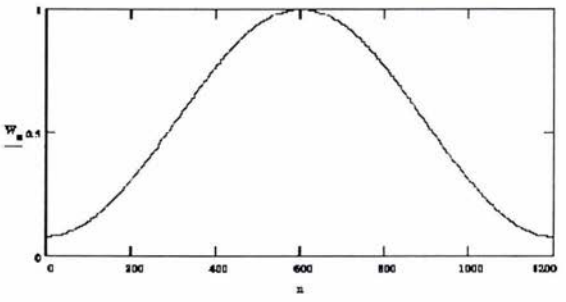


Fig. 1.6 Ventana de Hamming para 1200 muestras.

1.3.5 Coeficientes de autocorrelación

A cada trama de la señal se le calcula su *autocorrelación* mediante la siguiente expresión:

$$r(m) = \sum_{n=0}^{N-1-m} x(n)x(n+m), \qquad m = 0,1,2,...p$$

Donde el valor de autocorrelación más alto p, es el orden del análisis LPC. El valor de p en procesamiento de voz comúnmente es de 8, que es el que se emplea en este sistema.

La importancia de la *función de autocorrelación* es que resulta una forma muy conveniente de desplegar ciertas propiedades de la señal, tal como su periodicidad.

1.3.6 Coeficientes de predicción lineal, LPC

La idea básica detrás del modelo LPC es que dada una muestra de voz en un tiempo ***n***, ***s(n)***, puede ser aproximada como una combinación lineal de p muestras de voz precedentes[6]. Al minimizar la suma de las diferencias de los cuadrados (sobre un intervalo finito) entre las muestras de voz actuales y las predichas linealmente puede ser determinado un conjunto único de coeficientes de predicción, esto es,

$$s(n) \approx a_1 s(n-1) + a_2 s(n-2) + ... + a_p s(n-p)$$

Utilizando el método de autocorrelación para determinar este conjunto de coeficientes de predicción se llega a una matriz de la forma

[6] Rabiner y Juang. Fundamentals of speech recognition. Pág. 100

$$\begin{bmatrix} r_n(0) & r_n(1) & r_n(2) & \Lambda & r_n(p-1) \\ r_n(1) & r_n(0) & r_n(1) & \Lambda & r_n(p-2) \\ r_n(2) & r_n(1) & r_n(0) & \Lambda & r_n(p-3) \\ \text{M} & \text{M} & \text{M} & & \text{M} \\ r_n(p-1) & r_n(p-2) & r_n(p-3) & \Lambda & r_n(0) \end{bmatrix} \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \hat{a}_3 \\ \text{M} \\ \hat{a}_p \end{bmatrix} = \begin{bmatrix} r_n(1) \\ r_n(2) \\ r_n(3) \\ \text{M} \\ r_n(p) \end{bmatrix}$$

A esta matriz se le denomina Toeplitz y lo que se debe hacer para obtener los coeficientes de predicción lineal es resolver el sistema de p x p ecuaciones simultáneas, o bien, se pueden obtener con el empleo de métodos numéricos, como el algoritmo Levinson-Durbin.

Algoritmo de Levinson-Durbin

El método formal para convertir los coeficientes de autocorrelación a un conjunto de parámetros LPC se conoce como método de Durbin y puede ser dado formalmente como el siguiente algoritmo:

Inicialización.

$$E_{LP}^{(0)} = r(0)$$

$$Para \quad 1 \le i \le N_{LP}$$

$$\{$$

$$k_i = \frac{\left[r(i) - \sum_{j=1}^{L-1} \alpha_j^{(i-1)} r(|i-j|) \right]}{E^{(i-1)}} \qquad 1 \le i \le p \quad ; \quad 1 \le j \le i-1$$

$$\alpha_i^{(i)} = k_i$$

$$\alpha_j^{(i)} = \alpha_j^{(i-1)} - k_i \alpha_{i-j}^{(i-1)}$$

$$E^{(i)} = \left(1 - k_i^2\right) E^{(i-1)}$$

$$\}$$

Las iteraciones se realizan para i = 1, 2, 3, ..., p, y la solución final está dada por

$$a_m = coeficientes\ LPC = \alpha_m^{(p)}$$

1.3.7 Coeficientes LPC cepstral

El *cepstrum* se define como la transformada inversa de Fourier del logaritmo del espectro de potencia de la señal. El cálculo de los *coeficientes cepstral* se realiza a partir de los coeficientes LPC.

$$c_m = a_m + \sum_{k=1}^{m-1} \left(\frac{k}{m} \right) c_k \, a_{m-k} \qquad 1 \le k \le m$$

donde los a_m son los coeficientes LPC y los c_m son llamados *coeficientes LPC-Cepstral*, que son los coeficientes Cepstral derivados del análisis LPC. No son iguales a los coeficientes Cepstral obtenidos directamente a partir de la señal de voz real, sin embargo son una buena aproximación y han mostrado generar resultados semejantes, además de ser fácilmente obtenidos.

1.3.8 Cuantización vectorial

La idea detrás de la *cuantización vectorial* o *VQ*, por sus siglas en inglés, es que dado un conjunto **N** de vectores en un espacio de dimensión **K**, éste se puede dividir en un número **L** de regiones no uniformes (**L** < **N**), en el que cada región es representada por un solo vector, también de dimensión **K**, denominado *centroide* o *cuantizador*, el grupo de vectores representados por un centroide se le denomina *clúster* y el conjunto de **L** centroides es referido en la bibliografía como alfabeto (codebook).

Este sistema utiliza *VQ*, dado que se tiene un número muy grande de datos de entrenamiento, y se desea obtener un número reducido pero muy representativo de las señales de voz. Todo esto, dado que la información debe ser almacenada en una computadora que tiene un cierto límite de memoria.

La forma en que se realiza la *VQ* es a través de algoritmos de agrupamiento los cuales acomodan los grupos de vectores de acuerdo a los que presentan características similares. En este caso se utiliza el algoritmo de agrupamiento ***K-medias*** porque genera buenos resultados y por que nos permite fijar en forma predeterminada el número de centroides.

El conjunto de vectores a los que se les aplica el algoritmo se obtiene reuniendo todos los vectores de coeficientes (coeficientes LPC o coeficientes cepstral), que corresponden a un determinado segmento de cada una de las señales de entrenamiento que representan a la misma palabra.

Algoritmo K-medias

Se describe a continuación el algoritmo de agrupamiento K-medias también llamado algoritmo de Lloyd.

1) Escoger K centros de grupo iniciales $z_1(1)$, $z_2(1)$, ..., $z_k(1)$.
2) En la iteración l, asignar las muestras a los grupos:

$$Asignar \quad x \quad a \quad \chi_i(l) \quad si \quad |x - z_i(l)| \quad \leq \quad |x - z_j(l)| \qquad j = 1,2,...,K \quad j \neq i$$

3) Calcular los nuevos centros de grupo:

$$z_i(l+1) = \frac{1}{N_i} \sum_{x \, en \, \chi_i(l)} x \qquad i = 1,2,...,K$$

donde N_i es el numero de muestras asignadas al cluster j(l).
4) Si $z_i(l+1) = z_i(l)$ para i = 1, 2, ..., K el algoritmo ha convergido y debe terminarse. En caso contrario, regresar al paso 2.

1.4 Clasificación de voz

La comparación se realiza de la siguiente forma: por cada segmento de la señal que se desea reconocer, se tienen un determinado número de vectores de coeficientes (LPC o cepstral), cada vector es comparado contra los centroides que corresponden al mismo segmento de una palabra, se toman la distancias menores entre vectores y centroides, y se suman (se genera resultado parcial d_{siwj}, si:segmento i-ésimo, wj:palabra j-ésima); a continuación los resultados parciales por cada segmento y por cada palabra son sumados (se genera resultado D_{wj}); finalmente los resultados D_{wj} son comparados, se escoge la menor y la palabra es reconocida por aquel conjunto de centroides contra los cuales se tenga la menor distancia.

1.4.1 Distancia euclidiana

También denominada como error cuadrático o distancia euclidiana entre dos vectores. Es la medida más conveniente y comúnmente utilizada para calcular la distancia entre dos vectores y se define como:

$$d(X_1, X_2) = |X_1 - X_2|^2 = \sum_{j=1}^{n} (X_{1j} - X_{2j})^2$$

1.4.2 Distancia de Itakura-Saito

La distancia de Itakura-Saito, es una distancia de máxima similitud entre un vector de coeficientes de predicción X_1 y otro vector de coeficientes X_2 dada por,

$$d(X_1, X_2) = (X_1 \bullet X_2)^T \Phi x (X_1 \bullet X_2)$$

donde

$$\Phi x = \{\phi(i-k)/\phi(0), \qquad 0 \le i,k \le N-1\}$$

es la matriz de autocorrelación normalizada cuyos coeficientes Φ(i-k) son usados para el cálculo del vector coeficientes de predicción X_1. Se puede notar además que Φx es una matriz de ponderación que cambia de valor al cambiar el vector de coeficientes de predicción X_1[7].

[7] Arturo Gardida Degollado. Opus citatum. Pág.37.

Capítulo 2

Aspectos básicos del software Windows y Word

2.1 La interfaz de programación de aplicaciones de Windows (Windows API)

La *interfaz de programación de aplicaciones de Windows* ***(Win32 API)*** es la *API* fundamental de Windows. *API* son las siglas de *Application Programming Interface*. El término *API* suele utilizarse para describir un grupo de funciones que son parte de una aplicación pero que otra aplicación puede utilizar, por tanto, la *API* de Windows son un grupo de funciones que son parte Windows y que están disponibles para cualquier aplicación Windows.

Dentro de Win32 se engloban como extensiones el resto de las *API* que se pueden añadir a Windows: Windows 3.1, Windows 95, Windows 98, Windows NT, Windows 2000 y Windows XP.

La implantación de las funciones *API* no es completa en todas las plataformas. Las únicas plataformas que poseen un desarrollo completo de la *API* son Windows NT/2000 y XP.

2.1.1 Resumen de la API Win32 de Windows

La *interfaz de programación de aplicaciones de Windows* de Microsoft permite a las aplicaciones explotar la potencia de la familia de los sistemas operativos Windows. Usando esta *API* se pueden desarrollar aplicaciones que corren exitosamente en todas las versiones de Windows mientras toman ventaja de todas las características y capacidades únicas de cada versión. (Nótese que ésta es llamada formalmente *Win32 API*. El nombre de Windows API hace ver mucho mejor sus raíces en Windows de 16 bit y su soporte en Windows de 64 bit).

Desde el punto de vista funcional la API de Windows se puede dividir en las siguientes categorías:

Servicios del sistema. Las funciones de los servicios base dan a las aplicaciones acceso a los recursos de la computadora y las características del sistema operativo subyacente. Dentro de las funciones proporcionadas por los servicios del sistema se encuentran aquellas que administran la memoria, los sistemas de archivos, dispositivos, procesos y subprocesos.

Una aplicación utiliza estas funciones para administrar y monitorear los recursos que necesita para completar su trabajo.

Dentro de este mismo grupo se ubican las funciones de trabajo en red. Las funciones de red permiten la comunicación entre aplicaciones de diferentes máquinas que se encuentren en una red. Se pueden utilizar estas funciones para crear y administrar conexiones a recursos compartidos, como directorios o impresoras, asimismo se incluyen funciones dedicadas a que la aplicación pueda compartir recursos con otras aplicaciones.

Otro tipo de funciones que se encuentran en esta categoría son las de información sobre el sistema. Se pueden utilizar estas funciones para determinar características específicas de la computadora. Una aplicación puede conocer la presencia de un ratón o las características del disco duro sin más que usando funciones de información del sistema.

Librería de controles comunes. El shell incorpora un número de controles que ayudan a darle a Windows su aspecto y funcionalidad. Estas librerías se encuentran disponibles para todas las aplicaciones debido a que estos controles son soportados por DLLs, que son parte del sistema operativo. El uso de estos controles comunes ayuda a que la interfaz de usuario de una aplicación sea consistente con el shell y con otras aplicaciones.

Los controles comunes son un conjunto de ventanas de control soportadas por la librería de controles comunes Comctl32.dll. Como otros controles, un control común es una ventana hija que una aplicación utiliza en conjunción con otra ventana para llevar a cabo tareas de entrada/salida.

Interfaz de Dispositivos Gráficos. La interfaz de dispositivos gráficos proporciona funciones y estructuras relacionadas que una aplicación puede utilizar para generar salidas gráficas para monitores, impresoras y otros dispositivos. Con las funciones GDI se pueden dibujar líneas rectas, curvas, rutas, texto e imágenes de mapas de bits.

Interfaz de usuario. Desde el punto de vista técnico las funciones de interfaz de usuario dan a las aplicaciones los medios para crear y administrar una interfaz de usuario. Estas funciones se utilizan para crear y utilizar ventanas para desplegar salidas, hacer solicitudes al usuario y llevar a cabo las demás tareas necesarias para soportar la interacción con el usuario.

Shell de Windows. La interfaz de usuario de Microsoft da al mismo el acceso a una amplia variedad de objetos necesarios para correr aplicaciones y administrar el sistema operativo.

Los más conocidos y numerosos de estos objetos son los fólders y los archivos que se encuentran en las unidades de disco de la computadora. Existen también un número de objetos virtuales que permiten al usuario realizar tareas tales como enviar archivos a impresoras remotas o acceder a la papelera de reciclaje. El shell organiza estos objetos en un espacio de símbolos jerárquico y provee a los usuarios y a las aplicaciones de una manera eficiente y consistente de administrar objetos. El shell como tal se tratará con profundidad en el capítulo siguiente ya que es una parte importante de la realización de este trabajo.

2.1.2 Servicios del sistema

De entre los servicios del sistema que destacaremos se encuentran los de información del sistema y quedan resumidos en la tabla 2.1

Los puntos que nos interesan son el cierre del sistema, que es el apartado donde se engloba al apagado y el reinicio del sistema y el registro de Windows que juega un papel muy importante en Windows y del que se hablará posteriormente.

Concepto	*Descripción*
Registro	El registro de Windows es la base de datos del sistema para todos los sistemas operativos Windows. Contiene información acerca de la configuración del hardware y el software así como de los usuarios del mismo. Cualquier programa basado en Windows puede añadir información al registro y leer de nuevo esa información desde el registro. Los clientes buscan en el registro para obtener los componentes que puedan usar
Información del sistema	Recupera o establece las configuraciones, versión y métricas del sistema
Cierre del sistema (System shutdown)	Cierra la sesión del usuario actual, apaga el sistema o bloquea la estación de trabajo
Tiempo	Recupera y establece la fecha y la hora del sistema
Proveedor de Tiempo	Proporciona la duración exacta del uso del hardware o de una red, así como la duración de otros clientes en la red

Tabla 2.1 Servicios de información del sistema.

Cierre del sistema (System shutdown)

Los mensajes y las funciones de cierre del sistema permiten a las aplicaciones cerrar la sesión del usuario actual, apagar el sistema o bloquear la estación de trabajo.

Apagado del sistema. El cierre del sistema lleva al mismo a la condición en la cuál resulta seguro apagar la computadora. Durante esta etapa todos los buffers de los archivos de sistema son vaciados al disco, luego un mensaje de diálogo se despliega informando al usuario que la computadora puede ser apagada. Asimismo existe la opción de reinicializar el sistema que permitirá a la computadora encender e iniciar el sistema de nuevo en vez de desplegar el mensaje de diálogo de apagado del sistema.

Cierre de sesión. El cierre de sesión detiene todos los procesos asociados con el contexto de seguridad del proceso que llamó a la función de salida, saca al usuario actual de la sesión en la que está trabajando y despliega un cuadro de diálogo de inicio de nueva sesión.

Bloqueo de la estación de trabajo. Protege el desplegado del monitor de algún uso no autorizado cuando se deja encendida la pc.

Para apagar el sistema se utiliza la función *ExitWindowsEx* que pertenece a la API de Windows. Esta función cierra la sesión actual, apaga el sistema o lo reinicia de acuerdo a los parámetros que se le den.

2.2 Introducción al modelo de componentes de objetos (COM)

Windows es un producto formado en su mayor parte de componentes, piezas que conforman un gran puzzle que es el propio sistema operativo. Usando estos componentes, una aplicación puede acceder a los servicios del directorio activo, añadir elementos al escritorio activo, gestionar accesos directos, recorrer el contenido del espacio de símbolos, por mencionar algunos ejemplos. También existe la posibilidad de crear nuevos componentes que encajen en este puzzle y cuya finalidad sería añadir nuevas capacidades al sistema.

Todos los componentes que forman Windows, tanto los propios como los desarrollados para incrementar su funcionalidad, siguen normas de una especificación: el modelo de componentes *COM*. El término *COM* (*Component Object Model*, por sus siglas en inglés), *Modelo de componentes de objetos*, es parte inseparable de Windows y base de tecnologías

como el enlace e inserción *OLE*, la *automatización* o los *controles ActiveX*. Se trata por consiguiente de un elemento fundamental que se utiliza para el desarrollo de este sistema.

2.2.1 Reutilización de objetos

En los últimos años, el desarrollo de software ha pasado de estar en una etapa artesanal, casi primitiva, a otra que se podría denominar de "industrialización del software". La mayor influencia en esta evolución la han tenido las técnicas orientadas a objetos y el desarrollo basado en componentes, conceptos que aportan un factor primordial: la reutilización. Aunque se trate de un término novedoso en el campo de los lenguajes de programación, lo cierto es que no se trata de una idea en absoluto nueva. Tomemos como ejemplo la fabricación de automóviles. Actualmente a ningún fabricante de autos se le ocurre fabricar todos los elementos del producto que vende, desde la más mínima pieza del motor hasta las lámparas o las llantas. Existen productores especializados en cada campo, de tal forma que el fabricante de autos puede centrarse en lo que diferencia a su automóvil de los demás: diseño, ingeniería, tecnología, etc. Los vehículos se producen en gran cantidad gracias a la fabricación en cadena de todos sus componentes, obteniéndose como resultado un producto mejor a un menor costo.

En términos de desarrollo de software hablamos de que sería absurdo pretender crear una aplicación interviniendo y diseñando todas sus partes. Sería preciso tener conocimientos en campos tan dispares como el acceso a datos, el tratamiento de imágenes, gestión de dispositivos de impresión, comunicaciones, etc. Entonces resulta mucho más fácil delegar las tareas comunes en componentes especializados. El resultado será satisfactorio desde todos los puntos de vista: menor tiempo de desarrollo, menores costos y mayor fiabilidad por mencionar algunos.

Sin embargo, reutilizar piezas de software es una tarea más simple de plantear que fácil de llevar a la práctica. Múltiples lenguajes y sistemas, incompatibilidades diversas entre ellos, recelo por parte de los programadores a usar un código no escrito y mantenido personalmente y, en definitiva, la falta de un estándar que facilite la reutilización son causas que han pesado durante mucho tiempo.

Lenguajes orientados a objetos

La primera tentativa de facilitar la reutilización de un código se planteó en los lenguajes orientados a objetos, cuyo exponente más representativo, en cuanto su uso, es sin duda

C++. Este tipo de lenguajes cuenta con características fundamentales como la abstracción, encapsulación y herencia, cuyo objetivo es simplificar el desarrollo y permitir aprovechar un código de una forma más sencilla que en los lenguajes tradicionales.

Ahora bien, las técnicas mencionadas anteriormente facilitan la reutilización de una forma limitada. Los elementos que es posible usar múltiples veces o bien extender son clases, moldes, a partir de los cuáles se crean objetos. Dichos moldes, creados usando una sintaxis específica del lenguaje que proceda, no pueden ser usados desde otros lenguajes. Por ejemplo, una clase definida en C++ no sirve para crear un objeto con COBOL o Pascal. Aunado a esto, en ocasiones existen incompatibilidades entre distintas versiones del mismo lenguaje. Las técnicas de orientación a objetos se enfocan en facilitar la reutilización del código siempre con el mismo lenguaje y, en la mayoría de las ocasiones, al nivel de código fuente. No existe una especificación que establezca cómo se ha de realizar la comunicación con esos objetos en formato binario, lo cuál es un impedimento para su eficiente aprovechamiento.

2.2.2 Modelos de componentes

La solución a los problemas planteados anteriormente surge con los conocidos como *modelos de componentes*. Un *modelo de componentes* es una especificación en la que se definen, entre otros datos, la estructura de los componentes, la comunicación entre ellos y los medios disponibles para que el desarrollador los manipule desde sus programas o desde entornos de desarrollo creados con este fin.

Actualmente, existen tres modelos de componentes especialmente importantes: *COM*, *JavaBeans* y *CORBA* (*Common Object Request Broker Architecture*, Arquitectura común para gestores de solicitudes a objetos). Este último no es en sí un modelo de componentes, sino una tecnología que permite que componentes escritos en diferentes lenguajes y que se ejecutan en distintos sistemas, se comuniquen entre sí. Por su parte *JavaBeans* es un modelo ligado íntimamente a la plataforma Java.

COM, que es el modelo que nos interesa, fue creado por Microsoft para el sistema operativo Windows, aunque posteriormente tanto Microsoft como terceras empresas lo han transportado a otros sistemas. Se trata de una especificación binaria relativamente sencilla. En ella se establece la estructura que debe tener un objeto COM y la forma de acceder a los

servicios que implanta. No se centra, por el contrario, en cómo deben definirse o implantarse esos objetos a nivel de código fuente.

En consecuencia, un objeto *COM* puede crearse usando cualquier lenguaje, ya sea orientado a objetos o no, y desarrollado en cualquier plataforma, aunque lo habitual es que ésta sea Windows. Lo único importante es que la imagen binaria de ese componente cumpla con el modelo *COM*.

La ventaja de un modelo de componentes como *COM* con respecto a los objetos generados con un lenguaje como C++ es clara. Un objeto *COM* podrá ser usado desde cualquier lenguaje y en cualquier sistema que cumpla con las especificaciones *COM*, ya que se trata de un estándar binario. Los detalles de codificación e implantación, que son la base de los lenguajes orientados a objetos, no tienen importancia alguna.

2.2.3 Definición de COM

COM es una especificación que define un modelo de componentes binarios. COM es una parte fundamental de Windows. En sí, *COM* es una arquitectura estándar de software basado en interfaces que está diseñado para separar código en componentes y en donde cada componente expone un conjunto de interfaces a través de la cuáles toda la comunicación con ese componente se maneja. De acuerdo con Microsoft, *COM* es un sistema distribuido, orientado a objetos e independiente de la plataforma que se utiliza para crear componentes binarios de software que pueden interactuar entre sí[1].

COM es un conjunto de servicios que el propio sistema pone a nuestra disposición. Dichos servicios se encuentran alojados en lo que se conoce como *librería COM* y en el *administrador de servicios de control* o *SCM* (*Service Control Manager*), por sus siglas en inglés. Se trata, básicamente, de servicios de bajo nivel para los servidores y clientes COM. La *librería COM* contiene funciones básicas para el funcionamiento de COM en una aplicación. El *SCM* es un ejecutable que facilita la comunicación entre servidores y clientes interactuando con la librería COM.

En cierto modo COM ha conseguido lo que los lenguajes orientados a objetos venían prometiendo desde hace mucho tiempo: la posibilidad de reutilizar código, de no tener que desarrollar repetitivamente los mismos elementos. La manera en que COM realiza esto es, básicamente, separando la *interface* de la *implantación*.

[1] Traducción de la definición de COM que se da en el sitio de Microsoft (www.msdn.microsoft.com)

COM es una tecnología relativamente simple en su concepción, si bien ha sido la base que ha servido para desarrollar otras arquitecturas más complejas. Entre éstas se encuentran las capacidades para crear documentos compuestos, los conocidos *controles ActiveX* y el poder controlar unas aplicaciones desde otras mediante la automatización, que es en la que se basa el presente trabajo. Para comprender y aprovechar las posibilidades de dichas técnicas es indispensable entender qué es y cómo funciona COM.

2.2.4 Estructura de COM

El concepto fundamental que se necesita tener claro para comprender el modelo de componentes de objetos es el de la separación entre la interfaz pública de un componente y su implantación. Este concepto es el que da lugar a elementos como las interfaces COM, las tablas de métodos o las clases de objetos.

Un *componente COM* es un objeto que implanta una o más interfaces, entendiendo como tales simples zonas de intercomunicación que envían fuera un objeto COM. El objeto tiene que asociar a cada una de éstas el código apropiado para conseguir una determinada función que se espera de ellos, realizando lo que se conoce como *implantación*. Desde este punto de vista, un objeto COM se puede representar desde fuera como una caja negra, como se ve en la figura 2.1. Lo que ocurre en ella, únicamente se puede saber por medio de lo que entra y sale de sus interfaces. Y de la misma manera, el mundo exterior es para el objeto como un agujero negro, está oculto para él, porque su pequeño universo acaba en sus fronteras exteriores. Esto es, que sólo le es posible comunicarse a través de sus interfaces.

Desde un punto de vista más formal se denomina *interface* a un conjunto de servicios expuesto por un componente COM. Desde el punto de vista lógico se podría decir que una *interface* es un contrato público en el que el componente se compromete a cumplir, implantando los servicios descritos.

Para poder tener una referencia vamos a hablar de las interfaces haciendo una analogía con el lenguaje C++. Físicamente, la *interface*, es una tabla formada por punteros a métodos que la misma interfaz soporta y que están implantados por un cierto objeto, a dicha tabla se le denomina *tabla virtual* o *vtable*. Así, para utilizar un cierto objeto, desde cualquier aplicación Windows, tan sólo es preciso saber cómo recuperar un puntero a cualquiera de sus interfaces. A partir de ahí bastará con llamar a sus métodos como si se tratara de una llamada a cualquier método desde un apuntador de una clase C++, cómo se puede ver en la

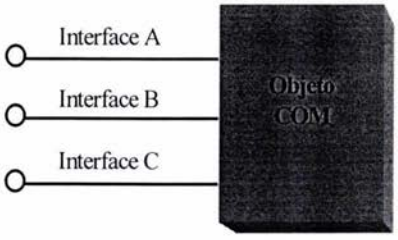


Figura 2.1 Estructura de un objeto COM

figura 2.2. De esta manera, un objeto COM puede exponer sus funciones en su tabla virtual para proporcionar al cliente acceso directo a tales funciones.

El usuario de un componente COM no necesita conocer detalle alguno de la implantación o funcionamiento interno de una interfaz. Sólo es necesario que conozca las interfaces que se encuentran disponibles en el objeto que, como ya se mencionó, son los únicos elementos a los que puede tener acceso.

Al crear un componente COM hay que decidir cuáles interfaces han de ser las que se implantarán, estás pueden ser interfaces estándar predefinidas o bien interfaces creadas por nosotros mismos. Se puede decir que una interface es como una clase abstracta de C++, ya que implantar una interfaz significa, que el objeto que se cree tendrá que codificar todos los métodos indicados en ella y los detalles de la implantación no interesan a nadie excepto al propio desarrollador del objeto.

Ahora bien, las interfaces no representan por si mismas ningún elemento que pueda utilizarse sin más, puesto que es necesario que estén asociadas con algún componente que ejecute las acciones que el usuario espera. Un objeto COM es el motor que ejecuta los métodos definidos en las interfaces, la ingeniería que se encargará de que cuando un cliente llame a un método de una interfaz se produzca el efecto esperado y es el desarrollador del objeto el responsable de unir todos los componentes básicos creando la lógica que llamamos implantación.

Retomando la analogía que hacemos de COM con C++ tenemos que un componente COM es similar a un objeto de C++, no obstante, existe una diferencia fundamental: un objeto C++ puede ser creado y usado directamente, ya que cuenta con un archivo de cabecera que

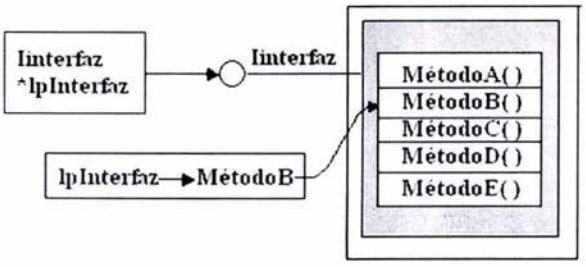


Fig. 2.2 Representación de una interfaz, su tabla virtual y un apuntador al MétodoB( )

contiene una definición de clase para ese objeto; mientras que un componente COM, como puede estar implantado en otro lenguaje, únicamente se puede acceder a sus interfaces.

Además, para que un objeto pueda ser considerado como un *componente COM* es preciso que cumpla los siguientes requerimientos: Que controle por si mismo su tiempo de vida, conociendo el número de punteros a sus interfaces que están actualmente en uso. Que cuente con un mecanismo de resolución de interfaces, es decir, que proporcione a los clientes el acceso a otras interfaces que dicho objeto soporte. Asimismo, es necesario que pueda realizar el registro de los componentes y que implemente la factoría o factorías que permitan a los clientes crear dichos componentes.

Servidores y clientes COM

Los objetos C++ no existen individualmente en forma aislada, sino que forman parte de aplicaciones, librerías de enlace dinámico, componentes o servicios. De igual manera, los componentes COM no existen aisladamente sino que se alojan en el interior de librerías de enlace dinámico o de ejecutables. A estos contenedores se les denominan *servidores* ya que, a diferencia de una DLL o de un ejecutable corrientes, poseen los elementos, mencionados en la sección anterior, que los habilitan para ser utilizados por el motor COM. Por su parte, un *cliente* es cualquier código u objeto que obtiene un apuntador a un servidor COM y hace uso de sus servicios cuando llama a los métodos de sus interfaces.

Existen dos tipos de servidores, los *servidores dentro de proceso* y los *servidores fuera de proceso*. Los servidores dentro de proceso están implantados en librerías de enlace dinámico y se ejecutan dentro del espacio de proceso del cliente y, los servidores fuera de

proceso se ejecutan en su propio espacio de proceso y se implantan a manera de archivos ejecutables. Dentro de los servidores fuera de proceso tenemos a aquellos que residen en máquinas locales y a aquellos que residen en máquinas remotas, a estos últimos se les denomina como *servidores remotos*.

Como ejemplos de servidores dentro de proceso, ejecutables, tenemos aplicaciones como Microsoft Excel o Microsoft Word, por mencionar dos de las más conocidas, siendo la automatización de ésta última uno de los pilares para la realización del presente trabajo. Se trata de aplicaciones cuya funcionalidad está construida en forma de componentes que, gracias a COM, pueden ser usados desde cualquier otra aplicación. Por otra parte, existen numerosos ejemplos de servidores dentro de proceso, librerías de enlace dinámico, ya que el propio sistema Windows está repleto de librerías de enlace dinámico que alojan multitud de componentes diversos, entre ellos, los que controlan al shell de Windows, que es resulta ser la otra parte fundamental del desarrollo de este sistema.

La interfaz IUnknown

Independientemente de su cometido, cualquier objeto COM debe ofrecer al menos una interfaz predefinida: la interfaz **`IUnknown`**. Ésta le sirve a un cliente potencial como primer punto de acceso y le ayuda a averiguar si el objeto en cuestión ofrece alguna interfaz adicional. De cierta manera esta interfaz es el pilar central de COM ya que se encarga de dos tareas fundamentales: el control de la vida de los objetos y de proporcionarle al cliente el acceso a otras interfaces que el objeto posea. De hecho todos los objetos COM cuentan con esta interfaz, puesto que las interfaces COM derivan o tienen como base, directa o indirectamente, a **`IUnknown`**.

La interfaz **`IUnknown`** posee solamente tres métodos, que son los que debe soportar cualquier componente COM: *QueryInterface*, que se utiliza para identificar y navegar por las interfaces que un objeto soporta, es decir, es el método que realiza la resolución de interfaces; *AddRef*, que es llamado cada vez que un cliente solicita una interface y *Release*, que es llamado cada vez que un cliente libera una interface. Ambos métodos son complementarios, ya que manejan un contador interno que posee el objeto COM, *el contador de referencias*. El contador de referencias indica el número de apuntadores a interfaces que están siendo manejados por los clientes. Cuando el contador de referencias se

hace cero, este valor le indica al objeto de que ya no existe ningún cliente que esté usando sus interfaces y que puede removerse a sí mismo de la memoria de manera segura.

El registro de Windows y la identificación de objetos e interfaces

Diferentes componentes pueden o no pueden soportar diferentes interfaces y los clientes pueden decidir en tiempo de ejecución que componentes utilizar. Para ayudar con estas tareas necesitamos identificar de alguna manera todos los tipos de interfaces y a aquellos componentes que son referenciados por otros componentes o por otros clientes. Dicho de otra manera, para que se consiga la separación entre interfaces e implantación, es necesario que las interfaces y los componentes puedan ser identificados independientemente del lenguaje en el que estén implantados, sin embargo, dichos elementos poseen un ámbito reducido, es decir, sólo son válidos en una aplicación o en un entorno de desarrollo específicos. Es por esta razón que el sistema los reconoce mediante otros identificadores que son mucho menos descriptivos y que son únicos en el tiempo y en el espacio, ya que así lo garantiza el algoritmo generado por Microsoft para su generación. Dichos identificadores se denominan *identificadores globales únicos* o *GUID (Global Unique IDentifier)*, por sus siglas en inglés y son estructuras de datos que contienen 128 bits que se generan a partir de datos tomados del hardware, la fecha y la hora y un contador secuencial[2].

Cada una de las interfaces estándar tiene asignado un GUID y todos y cada uno de los objetos COM que forman el sistema se identifican mediante un GUID.

Los GUID que se utilizan para identificar componentes COM se denominan *CLSID (Class Identifier)* o *identificadores de clase*, éstos siguen siendo un GUID, como cualquier otro, sólo se trata de una diferencia de notación con el fin de poseer un claro manejo de dichos identificadores. De la misma manera los GUID que se utilizan para identificar interfaces se denominan *IID (Interface IDentifier)* o *identificador de interface*.

Como un ejemplo de un GUID tenemos {0xC1E6265C, 0x017A, 0x11D3, {0x92, 0x46, 0x00, 0xC0, 0xDF, 0x80, 0x3B, 0x5F}}.

Como se puede observar es difícil manejar un número de esta manera, por lo que para facilitar su utilización a cada identificador se le asocia con una constante cuya notación está formada de la siguiente manera. En el caso de componentes, se utiliza el prefijo `CLSID_` aunado al nombre del objeto, por ejemplo la constante `CLSID_InternetExplorer`

[2] Jorge Pascual et al. Programación Avanzada en Windows 2000, pag. 726.

corresponde al CLSID del Explorador de Internet y, en el caso de interfaces, se utiliza el prefijo `IID_` aunado al nombre de la interfaz, por ejemplo, la interfaz **`IUnknown`** tiene asociada la constante `IID_IUnknown`. Así si se requiere desarrollar un nuevo componente con sus respectivas interfaces por nuestra parte es necesario asignarle un GUID que represente al componente y un GUID para cada una de sus interfaces.

Por otra parte, para que un cliente pueda acceder a los CLSID, éstos junto con los IID son almacenados en el *registro de Windows*, en donde cada identificador de clase es mapeado con un identificador programático o PROGID y con su aplicación, como se muestra en la figura 2.3 para Excel. Además de contener la información de la configuración del hardware, el software y de los usuarios del sistema, el registro mantiene información acerca de todos los objetos COM que se encuentran instalados en el sistema.

De manera general, cuando una aplicación crea una instancia de un componente COM consulta el registro para mapear el CLSID o el PROGID del componente con sus respectivos nombres de las rutas de sus archivos dll o ejecutables, que son en realidad donde se encuentran implantados dichos componentes. Esto hace que la localización física de los componentes sea transparente al cliente, ya que éste no necesita saber dónde se localiza el archivo dll o el archivo exe que proporciona el servicio del componente para poder utilizarlo. Después de determinar el servidor del componente, Windows carga el servidor dentro del espacio de proceso de la aplicación cliente si se trata de un servidor dentro de proceso o inicializa el servidor en su propio espacio de proceso si se trata de servidores locales o remotos. El servidor crea una instancia del componente y regresa al cliente una de las interfaces del componente.

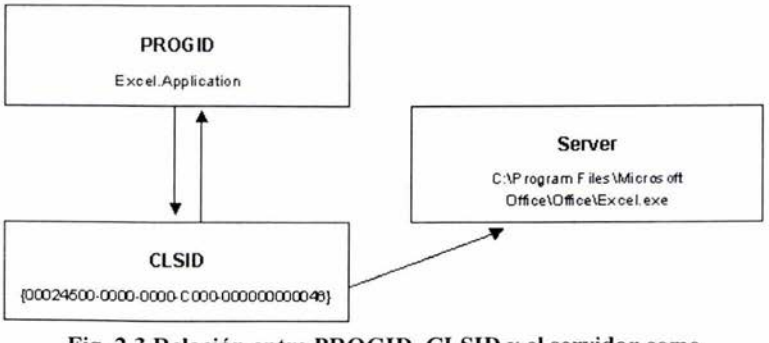


Fig. 2.3 Relación entre PROGID, CLSID y el servidor como se describe en el registro de Windows.

Aunque no se ha definido la automatización aquí se hace mención de ella porque se considera importante. Para automatizar una aplicación de Office, se puede usar su CLSID o su PROGID para crear un objeto a partir de una de las clases que expone la aplicación de Office. La tabla 2.2 lista los PROGID utilizados más comúnmente por las aplicaciones de Office 97, Office 2000 y Office XP.

Aplicación	*Identificador programático (PROGID)*
Microsoft Access	**Access.Application**
Microsoft Excel	**Excel.Application** **Excel.Worksheet** **Excel.Chart**
Microsoft Graph	**MSGraph.Chart**
Microsoft Outlook	**Outlook.Application**
Microsoft PowerPoint	**PowerPoint.Application** **PowerPoint.Presentation**
Microsoft Word	**Word.Application** **Word.Document**

Tabla 2.2 Identificadores programáticos para las aplicaciones de Office.

Los identificadores programáticos aquí listados son independientes de la versión, puesto que existen también, aquellos identificadores programáticos que son dependientes de la versión, por ejemplo, para Word 97 su PROGID es Word.Application.8 y para Word 2000 su PROGID es Word.Application.9, sin embargo, es mejor utilizar aquellos que son independientes para asegurar que el cliente sea compatible con múltiples versiones de estas aplicaciones.

La factoría del componente

Recordemos la diferencia que se estableció entre un objeto C++ y un componente COM. Como el componente COM no cuenta con un archivo de cabecera no le es posible crearse ni destruirse explícitamente, es decir, que no se pueden usar los operadores new y delete del lenguaje C++. Entonces para poder utilizar un componente COM es necesario crear, primero, instancias del objeto. Para esto existe una clase *factoría* que se encuentra asociada con cada componente COM y dicha clase "sabe", de alguna manera, cómo crear las instancias correspondientes. Así, si se desea crear una instancia de un componente determinado, se debe preguntar a su factoría.

Una factoría es un componente COM que se diferencia de los demás componentes en que siempre implanta la interfaz **`IClassFactory`** o alguna derivada de ella. A su vez esta interface contiene un método llamado *CreateInstance*, que es el que internamente crea al componente. La clase factoria sólo necesita ser implantada por aquellos componentes que han sido registrados mediante un CLSID. Los componentes que sean creados por otros componentes en el mismo servidor no necesitan estar asociados con una factoría o tener un CLSID.

Después de que la factoria crea una instancia del componente regresa un apuntador a la interface solicitada. De esta manera, el cliente ya puede comenzar a llamar a los métodos de la interface que ha obtenido.

2.2.5 Librerías de tipos

Desde el inicio de este tema se han encontrado una manera de explicar las interfaces y los componentes mostrando las similitudes y las diferencias con el lenguaje C++ para facilitar su comprensión. Imaginemos que vamos a utilizar este lenguaje para crear tanto un cliente como un servidor COM y que la descripción de las interfaces y la definición de los GUID se encuentran en archivos de cabecera, entonces el acceso del cliente al servidor se haría directamente a través de estos archivos y bastaría, únicamente, incluirlos en el proyecto del cliente. Ahora imaginemos que el cliente desea utilizar un servidor que ha sido creado con un lenguaje intérprete, como puede ser Visual Basic. Existe un problema fundamental: Visual Basic no puede usar los archivos de cabecera en los que se describen las interfaces y los GUID, por lo que difícilmente se podría obtener la información necesaria para acceder al componente.

Para resolver este problema, se crearon las librerías de tipos. Una *librería de tipos* es la interfaz pública de un servidor COM y contiene las definiciones de interfaces e identificadores de objetos, es decir, una librería de tipos contiene exactamente la misma información que se encuentra almacenada en el archivo de cabecera, aunque en un formato que es posible utilizar desde herramientas como Visual Basic y no cuenta, sin embargo, con detalle alguno acerca de la implantación de los objetos, manteniendo así el objetivo que tiene COM.

Una librería de tipos es un archivo cuyo contenido puede, opcionalmente, integrarse en forma de recurso en el propio servidor. De esta manera se le puede dotar al servidor de una librería de tipos para que pueda ser utilizado por otros lenguajes compilados, como C++.

2.3 Automatización

Un componente COM simple puede ser usado prácticamente desde cualquier entorno de desarrollo actual, siempre que se cuente con una librería de tipos. Sin este recurso, determinadas herramientas no pueden obtener la información necesaria para crear los objetos y utilizar sus interfaces. Aún así, un componente COM no puede ser utilizado desde los lenguajes de script o basados en guiones que cada vez tienen mayor presencia. Estos lenguajes le permiten automatizar tareas en ciertas aplicaciones, como Microsoft Excel, y escribir guiones que se ejecuten directamente, mediante Windows Scripting Host.

Sin embargo, existe un mecanismo, que facilita el uso de un componente COM desde cualquier lenguaje, ya sea de script o no. Este mecanismo o técnica es conocido como *automatización* y consiste, básicamente, en implantar una determinada interfaz que actúa como intérprete entre el componente y los potenciales clientes.

2.3.1 Mecanismo de automatización

La *automatización*, originalmente llamada *automatización OLE*, es una tecnología de la que se toma ventaja de los servicios que ofrece un componente COM al incorporarlo a nuestras propias aplicaciones. El mecanismo de automatización está basado completamente en COM.

La automatización, al estar basada en COM, mantiene, también, la relación cliente-servidor, ya que establece que el primero se conecte al segundo de tal manera que pueda utilizar la funcionalidad que el servidor mismo provee. En otras palabras, el cliente inicia una interacción construyendo un objeto componente, que podría tener que cargar, o asociándose a un objeto COM de un software componente que ya se está ejecutando; el cliente llama después a las funciones de interfaz del componente y libera esas interfaces cuando termina. Más adelante hablaremos de ésta interacción.

Lo anterior es posible gracias a que el componente implanta una interfaz, denominada **`IDispatch`**, bien conocida por los potenciales clientes y que actúa como intérprete entre servidor y cliente. Los métodos de ésta interfaz permiten conocer los servicios que hay

disponibles en el componente, facilitando su ejecución, es decir, esta interfaz es la que proporciona a un cliente el acceso a los servicios de un objeto.

Por otra parte, para que un *componente automatizable* sea considerado como tal debe contar con una *interfaz de despacho* o *dispinterface (dispatch interface)*, nombre con el que se le conoce abreviadamente a la interfaz **`IDispatch`**. Esta interfaz permite que el componente ejecute sus métodos sin que el cliente disponga de un puntero a cada uno de ellos, es decir, sin que disponga de la definición de interfaces. Recordemos que los lenguajes compilados como, C++, son capaces de procesar elementos como los archivos de cabecera y las librerías de tipos durante la compilación, de tal forma que al ejecutar, se cuenta con la correspondiente tabla de punteros obtenida a partir de las definiciones. Un lenguaje interpretado, por el contrario, no cuenta con esa posibilidad, por lo que difícilmente podría usar un componente simple. El fin de la automatización es dar solución a este problema, y consiste, básicamente, de que el componente cuente con un punto de entrada común y bien conocido por los clientes, la interfaz **`IDispatch`**, punto desde el que podrá acceder a cualesquiera de los métodos que estén implantados en los servicios. Por tanto, el componente automatizable puede ser usado sin necesidad de disponer de archivos de cabecera, librerías de tipos ni elementos similares, lo que implica que puede ser utilizado, por ejemplo, desde los lenguajes basados en guiones existentes en herramientas de uso común, como Word, clientes HTTP, como Internet Explorer, o el propio sistema, como Windows Scripting Host.

La automatización basa su funcionamiento en la asignación de un *identificador de despacho*, *dispid*, abreviatura de *dispatch indetifier*, a cada uno de los métodos implantados de tal forma que los clientes no necesitan ninguna tabla virtual a una interfaz, como C++. Para determinar cuál es el dispid de un cierto método, lo único que los clientes necesitan saber son los nombres de los métodos a llamar; con esta información y usando el mecanismo estándar de despacho, se resolverá cada llamada, obteniendo así dicho identificador que, a su vez, se usará para ejecutar el servicio realmente solicitado.

La interface IDispatch

El centro de todo el mecanismo de automatización es la interface **`IDispatch`**, una interface que tan sólo cuenta con cuatro métodos y que proporciona los medios para automatizar objetos COM. Esta interfaz es un estándar, lo que significa que prácticamente

todas las herramientas actuales conocen su funcionamiento. Los clientes no tienen que entendérselas directamente con esta interface, son los propios intérpretes de lenguajes como VBA los que se ocupan de ello. Al crearse un objeto automatizable, a petición de un cliente, el motor del lenguaje lo que obtiene es un puntero a la interfaz **`IDispatch`** . En el momento que el cliente decide llamar a uno de los métodos, el mencionado motor se sirve de los métodos de la interface **`IDispatch`** para realizar todo el trabajo. De esta forma, los clientes no tienen conocimiento de todo el proceso y complejidad que hay detrás, limitándose a usar el componente de una forma sencilla.

Las dispinterfaces tienen, como todo, ventajas e inconvenientes. Entre estos últimos se encuentra la pérdida de rendimiento provocada por el funcionamiento intrínseco del mecanismo de automatización. Para llamar a un cierto método es preciso obtener un identificador, a continuación hay que empaquetar apropiadamente todos los parámetros y realizar la llamada de forma indirecta, a través de un punto de entrada común.

Los métodos de la interface **`IDispatch`** son los siguientes: *GetTypeInfoCount*, que es llamado para determinar si la información de la librería de tipos se encuentra disponible; *GetTypeInfo*, llamado para determinar el tipo de información, es decir, cuando no conocemos los nombres de los métodos que pone a disposición una determinada interfaz, podemos utilizar esta función para obtener dichos nombres junto con los de sus parámetros; *GetIDsOfNames*: llamado para obtener el dispid a partir del nombre de una propiedad o

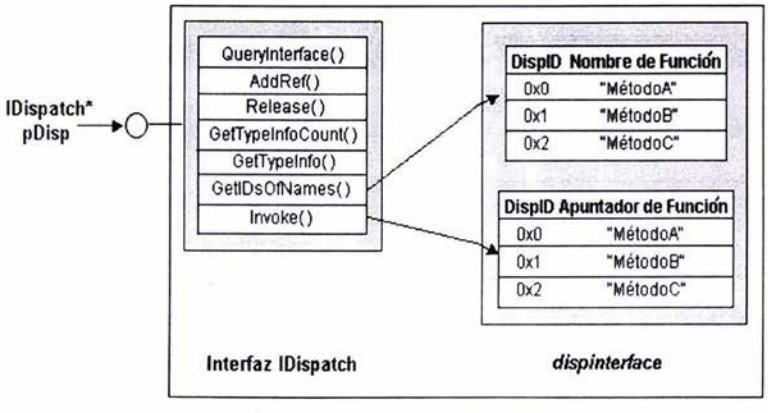


Fig. 2.4 Componente automatizable.

método e *Invoke*: llamado para invocar un método o una propiedad para un objeto.

Resumiendo, los métodos y las propiedades de un objeto forman en conjunto su interfaz de despacho o *dispinterface*. Dentro de ésta cada método y propiedad se encuentra definido por un único miembro que es el identificador de despacho de la función. Un objeto puede proporcionar a los clientes acceso a sus funciones utilizando una *dispinterface*, un arreglo de nombres de funciones y un arreglo de apuntadores a funciones que son indexados por los dispid, como se puede ver en la figura 2.4.

Mecanismo de puesta en marcha de un componente automatizable

A continuación se explicará cómo se lleva a cabo la puesta en marcha del componente, desde el punto de vista de C++, para dejar más claro cómo trabaja el mecanismo de automatización.

Para iniciar el proceso de automatización, un cliente primeramente hace una solicitud de los servicios COM con la función *CoInitialize*[3]. Después, crea una instancia del servidor de Automatización haciendo una llamada a la función *CoCreateInstance*[4]. A *CoCreateInstance* se le proporciona el CLSID del servidor[5] y con dicha función se hace una solicitud a la interface **`IUnknown`**. Una vez que el apuntador a **`IUnknown`** es retornado, el cliente hace una llamada al método *QueryInteface* de **`IUnknown`** para obtener un apuntador a la interfaz **`IDispatch`** del objeto que se desea automatizar. Una vez que el cliente obtiene este apuntador puede empezar el trabajo de llamar a los métodos y propiedades que expone el objeto. Para llamar a los métodos y propiedades necesita de sus correspondientes dispids. Para hacer esto, el cliente puede llamar a *GetIDsofNames* de la interfaz **`IDispatch`** y recuperar un dispid de una función del objeto y con este identificador, el cliente puede hacer uso del método Invoke de **`IDispatch`** para invocar el método o la propiedad correspondiente. Se continua de la misma manera, haciendo pares de llamadas a los métodos *GetIDsofNames* e *Invoke* de **`IDispatch`** hasta que la tarea de automatización haya sido concluida. Después se cierra el motor COM con la función *CoUninitialize*.

---

[3] *CoInitialize* es la función que permite inicializar el motor de COM.

[4] *CoCreateInstance* es la función que permite obtener una instancia de un objeto COM.

[5] Es posible obtener el CLSID a partir del ProgID utilizando la función *CLSIDFromProgID*

En pocas palabras para ejecutar un método o una propiedad del objeto mostrado en la figura anterior, un cliente primero debe: llamar a *GetIDsOfNames* para buscar el *dispid* de la propiedad o el método correspondiente. Y luego llamar a *Invoke* para ejecutar tal método o la propiedad usando el *Dispid* como índice del arreglo de apuntadores a funciones.

Hasta aquí hemos definido la automatización, en qué consiste y su utilización de manera general, sin embargo, falta la manera de aplicarlo a Word por lo que es necesario adentrarnos en el modelo de objetos de Office que se tratará a continuación..

2.3.2 El modelo de objetos de Office

Todas las aplicaciones de Microsoft Office tienen su propio lenguaje de script el cuál puede ser usado para llevar a cabo tareas dentro de las aplicaciones. Este lenguaje basado en guiones es Visual Basic para Aplicaciones (VBA). El conjunto de funciones que una macro de VBA puede utilizar para controlar su aplicación anfitriona es el mismo conjunto de funciones que un cliente de automatización puede utilizar para controlar la aplicación externamente, sin importar el lenguaje de programación en el que esté el controlador. Comprensiblemente las aplicaciones de Office proporcionan la documentación de sus funciones en una sintaxis que puede ser fácilmente interpretada en VBA, sin embargo, si el cliente se encuentra en otro lenguaje distinto es necesario traducir la sintaxis de las funciones de Office al lenguaje en el que se encuentre diseñado el cliente.

Por otra parte, los objetos son los bloques de construcción fundamental de las aplicaciones de Microsoft Office[6]; cualquier cosa que se haga con ellos incluye manipulación de objetos. Una aplicación de Office consiste de dos partes: *contenido y funcionalidad*. El *contenido* se refiere a los documentos que se encuentran en esa aplicación así como de las palabras, números y gráficos que están incluidos en dicho documento; también se refiere a la información de los atributos de los elementos individuales en la aplicación, tal como el tamaño de una ventana, el color de una gráfica, o el tamaño de la fuente de una palabra. La *funcionalidad* se refiere a todas las maneras en que se puede trabajar con el contenido en la aplicación, por ejemplo, abrir, cerrar, añadir, borrar, copiar, pegar, editar o dar formato a los elementos en dicha aplicación.

Las aplicaciones de Office exponen su funcionalidad como un conjunto de objetos programables. Cada unidad de contenido y funcionalidad en Office es un objeto que se

[6] versiones 97 en adelante.

puede examinar y controlar mediante programación. Un libro de trabajo, un documento, una tabla, una celda o un párrafo pueden considerarse como ejemplos de objetos que exponen las aplicaciones de Office. Al comprender cómo trabajar con estos y otros objetos, se facilita la automatización de las tareas en Office.

Antes de que se pueda manejar el contenido y la funcionalidad de una aplicación de Office es importante comprender que ambos están divididos en objetos discretos y que dichos objetos se encuentran ordenados en un modelo jerárquico, como si se tratara de un árbol familiar, al que se le denomina modelo de objetos. El objeto que se encuentra en el nivel más alto de la jerarquía en una aplicación es, generalmente, el objeto **Application**, que es la aplicación como tal. Este objeto tiene objetos hijos, y éstos a su vez tienen sus propios hijos y así sucesivamente.

Para comprender mejor esto tomemos como ejemplo una parte del modelo de objetos de Word[7] y que, como se puede ver en la figura 2.5, Microsoft Word es, en sí, el objeto **Application**. Dicho objeto tiene muchos hijos, a los que se accede solamente cuando este objeto existe, es decir, cuando la aplicación está corriendo. En este caso el objeto **Application** tiene como hijos a los objetos **Documents**, **Dialogs** y **Selection** (Documentos, Diálogos y Selección). Los dos primeros son colecciones de objetos, es decir objetos que contienen a otros objetos, y el tercero es solamente un objeto del que se puede hacer uso. Un objeto **Documents** es padre de los objetos **Document** (Documento) y un objeto **Dialogs** es padre de los objetos **Dialog** (cuadro de diálogo). A su vez el objeto **Document** es padre de las colecciones de objetos **Paragraphs** (Párrafos) y **Tables** (Tablas), los que también tienen sus propios hijos, que son los objetos de tipo **Paragraph** y los objetos de tipo **Table**, respectivamente. La lista continua pero con esto debe ser suficiente para comprender la organización jerárquica del modelo de objetos de Office, en este caso el de Word que es la aplicación que nos interesa.

Por otra parte, un objeto de Office no hace nada por si mismo a menos que un cliente pueda hacer algo con ese objeto. Para controlar o examinar un objeto se utilizan las propiedades y métodos que dicho objeto posee. Una *propiedad* es una función que establece o recupera un atributo para un objeto y un *método* es una función que ejecuta una acción sobre un objeto. En general, las propiedades se utilizan para obtener el contenido, el cuál puede ser el texto

[7] Para mayor información sobre el modelo de objetos de Word consulte la documentación en el sitio de Microsoft www.msdn.microsoft.com

que se encuentra en un objeto o el establecimiento de los atributos para ese objeto; por su parte los métodos se utilizan para obtener la funcionalidad, los cuáles establecen todo aquello que se pueda hacer con el contenido.

Una vez más tomemos el modelo de objetos de Word para darnos una idea de lo que se considera como una propiedad y lo que se considera como un método en Office. Word expone el objeto **Range** que representa una determinada región contigua del documento, donde probablemente deseemos encontrar una frase o una palabra ó, tal vez, realizar alguna modificación. Para identificar de manera única a un objeto **Range**, éste posee tres atributos, dos de ellos, *Start* y *End*, señalan los caracteres de inicio y fin de dicho objeto y *StoryType*, que señala el tipo de rango al que pertenece este objeto; estos atributos pueden ser determinados al inspeccionar las propiedades del objeto **Range**. Por otra parte hay acciones que se pueden realizar sobre ese objeto, como seleccionarlo, borrarlo o copiarlo; estas

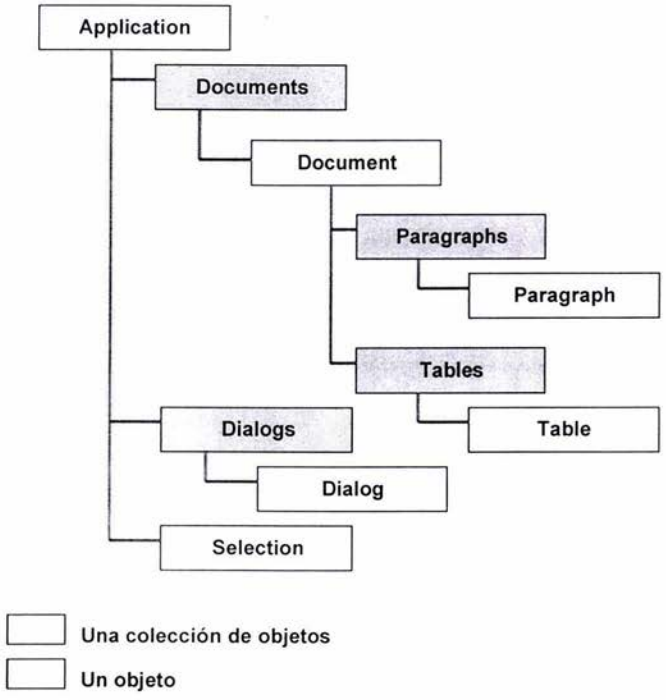


Fig. 2.5 Fragmento del modelo de objetos de Word esbozando la relación entre objetos

acciones representan los métodos *Select*, *Delete* y *Copy* del objeto **Range**.

Para acceder a un método o a una propiedad, en Visual Basic se navega hacia un objeto comenzando por el objeto que posee el nivel más alto y se continúa, descendiendo de objeto en objeto, hasta llegar hasta a aquel con el que queremos trabajar. De éste utilizamos los métodos y propiedades que necesitemos para modificar el contenido del documento.

Ahora consideremos algunos ejemplos de los métodos y propiedades que maneja Word para comprender mucho mejor la explicación anterior.

Si quisiéramos abrir un documento existente llamado "MiDocumento.doc" para trabajar con él tomaríamos la colección **Documents**, que representa a todos los documentos abiertos en la aplicación Word, y estableciéramos su propiedad *Filename* con el nombre del documento deseado.

```
Application.Documents.Open FileName:= "MiDocumento.doc"
```

Si quisiéramos insertar una tabla en este documento de la forma en la que lo hace Word accederíamos a su colección de Diálogos y con su método Item obtendríamos aquel que corresponde al diálogo que permite insertar una tabla y si obtenemos el indicador de Aceptar procedemos a ejecutar dicho comando.

```
Dim dlg As Dialog
Dlg  = Application.Dialogs.Item(wdDialogTableInsertTable)
If dlg.Display. = -1 Then dlg.Execute
```

Para guardar nuestro documento con los cambios realizados utilizamos el método *SaveAs* de **Documents** haciendo que la propiedad *FileName* tenga el nombre del documento.

```
Application.Documents.SaveAs FileName:= "MiDocumento.doc"
```

Después podríamos crear un nuevo documento utilizando el método *Add* de **Documents**.

```
Application.Documents.Add
```

De esta manera, al utilizar la sintaxis de VBA, nuestro cliente puede ser capaz de automatizar Word y otras aplicaciones de Office en forma sencilla y segura.

2.3.3 Librerías de tipos de Office

Desde el punto de vista de la automatización una librería de tipos es un archivo o parte de un archivo que proporciona información acerca del contenido y la funcionalidad de un objeto COM. En general Office y específicamente Word, requieren de una librería de tipos para ser automatizados con C++, dicha librería contiene información acerca de todas las clases o la descripción de los objetos del modelo de componentes de cada aplicación de Office.

La librería de tipos especifica toda la información que un cliente de automatización necesita para invocar un método o una propiedad de un objeto de Office. En cuanto a propiedades, la librería de tipos describe el valor que el objeto acepta o regresa y, en cuanto a métodos, la librería de tipos proporciona una lista de todos los argumentos que los métodos pueden aceptar, muestra el tipo de dato de cada argumento e indica si un argumento es opcional o no.

Las librerías de tipos suelen establecerse como un recurso en un archivo .dll, un recurso en un archivo .exe o un archivo de librería de tipos autosuficiente (.tlb)

Cada aplicación en Office proporciona recursos de librería de tipos múltiples en un solo archivo .dll. A un archivo .dll con recursos de librería de tipos múltiples se le denomina comunmente como librería objeto (.olb). La siguiente tabla muestra los nombres de archivos de las librerías de tipos para Microsoft Office 97, Office 2000 y Office XP. La librería de tipo de cada aplicación puede ser encontrada en la misma carpeta en la que se encuentra el servidor.

Aplicación	*Versión 97 (8.0)*	*Versión 2000 (9.0)*	*Versión XP (10.0)*
Microsoft Access	Msacc8.olb	Msacc9.olb	Msacc10.olb
Microsoft Excel	Excel8.olb	Excel9.olb	Excel10.olb
Microsoft Graph	Graph8.olb	Graph9.olb	Graph10.olb
Microsoft Outlook	Msoutl8.olb	Msoutl9.olb	Msoutl10.olb
Microsoft PowerPoint	Msppt8.olb	Msppt9.olb	Msppt10.olb
Microsoft Word	Msword8.olb	Msword9.olb	Msword10.olb

Tabla 2.3 Librerías de tipos de Office.

Capítulo 3

El shell de Windows

Hasta este momento hemos hablado acerca del modelo COM, de la automatización y la manera en que se puede utilizar en Microsoft Word y hemos dejado de lado los aspectos principales en los que se basa Windows. Es tiempo de retomar tales aspectos y establecer el rol que posee cada uno de ellos dentro del mismo sistema operativo. Desde el punto de vista funcional el conjunto completo del shell se puede dividir en dos áreas principales: las *funciones básicas* y las *extensiones*. La primera de ellas, la API de Windows, fue tratada sómeramete en el capítulo anterior y de la segunda, las interfaces COM, se dieron sus antecedentes por lo que en este capítulo se verá su aplicación en el shell de Windows. Es importante señalar que al agrupar las funciones y las interfaces bajo este criterio nos dará la oportunidad de entender al shell de Windows como un todo.

Debido a que Windows fue diseñado en lenguaje C puro, no fue pensado en términos de un diseño orientado a objetos y por tanto no es de sorprendernos que toda la funcionalidad básica de él haya sido expuesta por medio de llamadas a funciones API. Desde el punto de vista del shell, las interfaces de COM son una evolución de las llamadas a la API, ya que, cómo se sabe, COM permite que los componentes sean escritos y luego utilizados por medio de interfaces que exponen y no por otro medio. Es por esto, que estudiar el shell significa también profundizarnos en el conocimiento de las interfaces que éste nos proporciona para poder entenderlo.

3.1 Definición del shell de Windows

De manera general, el *shell* de un sistema operativo se define como la interfaz de usuario provista por el sistema para permitir al usuario llevar a cabo tareas comunes tales como acceder al sistema de archivos, ejecutar programas, cambiar la configuración del sistema, etcétera.

El shell de Windows tiene como fin fundamental proporcionar al usuario una forma de acceder a los recursos de la computadora. Esto implica proporcionar el acceso no sólo a los discos duros y al lector de CD-ROM, sino también al entorno de red y la configuración del sistema. El shell es un intermediario entre el usuario y la funcionalidad misma del sistema,

representada por el núcleo del sistema operativo con sus extensiones. Por ello el shell no es una parte del sistema operativo, sino que se ejecuta como un programa autónomo que, no obstante, tiene ciertos privilegios. Siempre y cuando el fabricante del sistema operativo no haya puesto obstáculos especiales, es posible en principio reemplazar el shell, como en el caso de los entornos UNIX, en los que existen variantes del shell en mayor o menor difusión[1].

El shell de Windows aplica también este concepto, y de una forma bastante avanzada puesto que si se retiran del Escritorio los componentes avanzados de la interfaz de usuario, como las imágenes, las listas y los menús contextuales permanece lo que ya se tenía en el shell de DOS y su línea de comandos: la posibilidad de crear archivos y directorios, copiarlos, moverlos, etc., y sobre todo la posibilidad de ejecutar programas. Este es el punto clave, ya que sin el shell no existiría la ejecución de programas.

Debido a que el shell ofrece una amplia gama de servicios y que en el presente trabajo no se abarca en su totalidad se tratarán brevemente los distintos servicios que ofrecen el shell mismo y el escritorio a los programas, centrándonos en las operaciones típicas de copia, reubicación, eliminación y cambio de nombre de archivos y creación de carpetas, dejando a un lado a otros servicios importantes como el mantenimiento de grupos de programas, la creación de accesos directos, el control de impresión, el almacenamiento de documentos, entre otros.

3.2 Los componentes del shell

Existen muchos componentes diferentes que forman al shell, pero comencemos con los más obvios. El *escritorio* y la *barra de tareas*, que se muestran en la figura 3.1.

Desde un punto de vista conceptual, el *escritorio* es el padre de todos los objetos que forman al shell. En términos de implantación, el *escritorio* es una ventana de una clase particular definida por el sistema y es el ancestro de todas las ventanas que son creadas y, por último, desde el punto de vista del usuario resulta ser el área de trabajo en la pantalla en la que aparecen ventanas, iconos, menús y cuadros de diálogos.

Otro componente primario del shell de Windows es la *barra de tareas*, pero ésta es en realidad sólo una ventana que pertenece al proceso del Explorador. Cada vez que se

1 Michael Tischer y Bruno Jennrich. PC Interno 5, pág. 1215

Fig. 3.1 El escritorio de Windows ...

necesita finalizar el proceso del Explorador, se hace que la barra de tareas aparezca y desaparezca. *La barra de tareas* es la ventana que también posee el menú de inicio, el área de notificación con el reloj y a aquellos botones que representan a las aplicaciones que se encuentran corriendo en ese momento.

3.2.1 El Explorador de Windows

Una concepción errónea que se da muy a menudo cuando hablamos del Explorador, es que se piensa que éste es sólo el software que corre cuando se intenta navegar por el sistema de archivos al dar doble clic en el icono del Explorador o al dar clic derecho en el botón de Inicio. Esto no es así, de hecho, el Explorador siempre está corriendo, prácticamente desde que se carga el sistema operativo (boot) hasta que se apaga la computadora. Lo que comúnmente se percibe como el Explorador, se trata únicamente de una ventana de navegación creada en un nuevo hilo que se añade al proceso del Explorador. En sí, el Explorador es el módulo ejecutable, llamado explorer.exe, que implementa el shell de Windows.

En otras palabras, el Explorador es la aplicación que juega el rol del shell del sistema. Y consiste de una colección de módulos especializados que trabajan conjuntamente para

formar a todos los objetos que componen al shell de Windows y darle también, la habilidad de llevar a cabo un conjunto de tareas especializadas. Entre éstas, se incluyen tareas como explorar una carpeta y mostrar un árbol específico de un subdirectorio, como se puede observar en la figura 3.2, o cargar un módulo externo, como una librería de enlace dinámico, y comunicarse con él.

Fig. 3.2 Ventana del Explorador de Windows.

Hablar de que el Explorador es capaz de cargar módulos y comunicarse con ellos implica que existen puntos muy bien definidos en su código donde se pueden monitorear las acciones que realice un usuario y, por tanto, posee soporte para extensiones o componentes que puedan mejorar su funcionalidad. Por ejemplo, cuando se ejecuta cierta acción, el Explorador busca extensiones registradas en el sistema, las carga y en última instancia las ejecuta. En secciones posteriores ahondaremos más en este punto, ya que es una parte importante de la realización del presente trabajo; por el momento es necesario definir otros conceptos.

3.3 La estructura del shell

Detrás del shell de Windows existe un número considerable de funciones API e interfaces COM. Esta colección heterogénea de comandos permite que se acceda y se trabaje con él de diferentes maneras. Las funciones y las interfaces no son dos aproximaciones equivalentes que proporcionan la misma funcionalidad. En lugar de eso, proveen diferente funcionalidad a diferentes niveles lógicos. Las funciones API cubren las operaciones básicas que un usuario quiera llevar a cabo sobre los objetos que forman el shell: fólders y archivos. Las interfaces COM dan la oportunidad de extender, mejorar y aún más, personalizar el comportamiento estándar de los diversos objetos del sistema que existen, incluyendo los del shell en sí mismos.

Imaginemos por un momento que el shell de Windows no fuera orientado a objetos en la práctica, es decir que no se utilizaran las interfaces COM. Entonces, ¿cómo podríamos justificar que existen algunos "objetos" organizados de una cierta manera y que identificamos claramente cuando navegamos en su estructura? Y que tales "objetos" poseen determinados atributos, que pueden considerarse como "propiedades", y también, que los objetos llevan a cabo acciones, que pueden considerarse como "métodos". Si el shell no

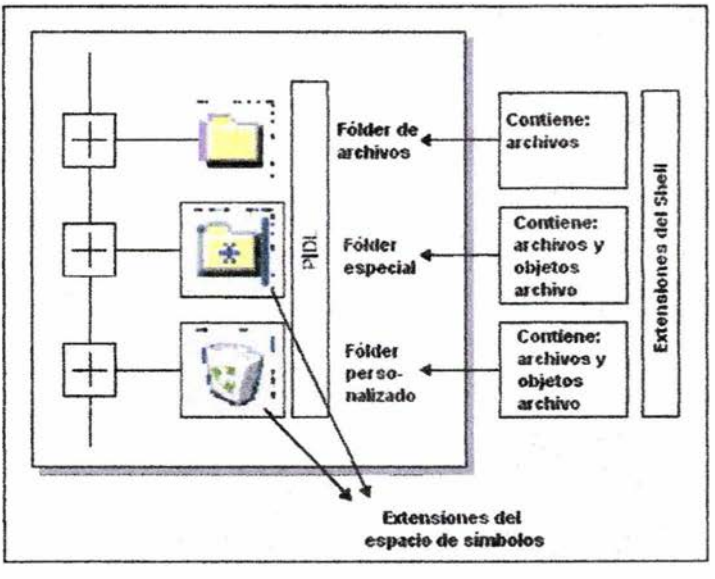


Fig. 3.3 Jerarquía de objetos del shell.

fuera orientado a objetos, entonces no podríamos hablar de todo un modelo de objetos, como COM, que encajara perfectamente y que se pudiera organizar para darle funcionalidad, sin embargo, podemos realizar una analogía para probar esto. Imaginemos una cierta infraestructura del shell, que puede ser vista como una jerarquía de objetos, es decir, tenemos una colección de objetos que trabajan conjuntamente de la forma que se representa en la figura 3.3.

De acuerdo con esta figura, básicamente el shell se compone de *fólders*. Un *fólder* es un contenedor de elementos hijos. Incluyendo como éstos a los subfólders y a los archivos, a estos elementos se les denomina *fólder ítems* o *elementos de fólder*. A la colección de fólders se le denomina *espacio de símbolos*[2]. En este último existe un fólder raíz, *el escritorio*, que es el padre de todos los fólders y entre sus hijos se incluyen elementos como las Impresoras y faxes, Mis documentos, la Papelera de reciclaje y posiblemente otros dependiendo de la configuración del sistema operativo. Ahora bien, lo que nos llama la atención de esta estructura es la forma única en que se manejan los distintos elementos, que son completamente diferentes entre sí pero que conviven en el mismo entorno. Todo esto gracias al modelo COM.

3.3.1 El espacio de símbolos

Basado en lo anterior, Windows contiene un mecanismo que permite que la información sea integrada dentro de estructuras de datos jerárquicas que el Explorador de Windows maneja internamente. Estos datos jerárquicos contienen información acerca de los objetos que se despliegan en los paneles del Explorador. A dichos datos jerárquicos se les denomina *espacio de símbolos*. Un espacio de símbolos es una colección de símbolos, tales como las claves de una base de datos o los nombres de archivos y carpetas.

El shell de Windows utiliza un sencillo espacio de símbolos para organizar todos los objetos, tales como archivos, carpetas, dispositivos de almacenamiento, impresoras, recursos de red y hasta el último elemento que pueda ser visualizado mediante el navegador del Explorador.

---

[2] También se le denomina espacio de nombres. Jorge Pascual, et al. Programación Avanzada con Windows 2000, pág 1036. Sin embargo, el término espacio de símbolos da una acepción más amplia de los objetos que contiene el shell. Michael Tischer, y Bruno Jennrich. Opus citatum, pág. 1218.

Aunque el espacio de símbolos es similar a la estructura de directorios del sistema de archivos, el primero contiene más tipos de objetos que sólo archivos y directorios. Y a partir de él se pueden visualizar el contenido de Mis sitios de red, la Papelera de reciclaje o de Mi PC. A su vez, a partir de Mi PC se pueden visualizar las unidades de disco, el Panel de control, las Impresoras y las Conexiones de red y acceso telefónico y así sucesivamente.

El Explorador utiliza interfaces COM para permitir al usuario acceder y manipular los datos del espacio de símbolos y desplegarlos de una manera gráfica. En el modelo de COM se estipula que se utilicen interfaces de este tipo para manipular y extender las estructuras internas del espacio de símbolos en lugar de acceder directamente a sus datos. De esta manera, el espacio de símbolos se maneja internamente como objetos COM.

Todo lo que el usuario puede ver en el escritorio pertenece al espacio de símbolos del shell. Dentro del espacio de símbolos cada objeto, es decir, cada archivo, carpeta o grupo de programas, cuenta con una identificación única que les distingue de los demás objetos. Esta condición permite que los elementos se mantengan separados y que se pueda acceder a ellos individualmente.

Fólders

Un *fólder* es un objeto del shell cuyo comportamiento está codificado en un módulo COM. Expone una interface, **`IShellFolder`**, que es común al shell de Windows. Por medio de esta conexión, un fólder puede decirle al shell cómo diseñar su contenido, qué icono usar para identificarlo, y qué texto empleará para describirlo. Por ejemplo, esto es lo que Mi PC hace para tener la apariencia de un fólder. Tiene una capa de código que detecta todas las unidades de disco disponibles en la computadora y añade un subárbol a la vista del Explorador para cada unidad.

Conceptualmente, un *fólder* es algo similar a un directorio en un sistema de archivos, pero puede o no puede tener una locación en un directorio físico. Si no está asociado de esta manera, se le denomina *fólder virtual*. Se pueden distinguir dos categorías principales de fólders: *fólders ordinarios*, también llamados *fólders de archivos (file folders)* o *carpetas* y los *fólders personalizados*. Naturalmente, los elementos contenidos en una carpeta son archivos, cuyos atributos son nombre, tipo, tamaño, última fecha de modificación, etcétera. Los elementos que se encuentran en cualquier otro tipo de fólder pueden ser archivos,

generalmente con un conjunto extendido de atributos, pero pueden ser algo completamente diferentes cómo impresoras o nodos de red.

Cada diferente tipo de fólder posee una capa de código diferente que proporciona su comportamiento. Por ejemplo, en el caso de los fólders de archivo, esto significa explorar el sistema de archivos para recuperar los archivos y los subfólders que lo componen, desplegándolos en un control List View[3]. Y en el caso de fólders como el de Impresoras y faxes, el sistema cuenta el número de impresoras instaladas y conectadas actualmente y despliega un icono para cada una de ellas. Podemos tener fólders prácticamente de cualquier tipo y que tengan cualquier comportamiento. Los directorios son sólo uno de los tipos posibles.

Del conjunto de fólders que no son directorios, existe una pequeña porción que son llamados *fólders especiales*. De hecho, éstos son fólders personalizados, que el shell de Windows proporciona por omisión y difieren de los directorios en los siguientes aspectos:

- Pueden contener archivos y otros objetos.
- Pueden proporcionar una visión diferente de su contenido.
- Pueden elegir no estar alojados en un directorio físico.
- Son parte de un grupo definido del sistema para el que se ofrecen un conjunto específico de funciones.

Por otra parte, un fólder especial, es un tipo particular de fólder que posee un módulo COM que le proporciona un comportamiento específico. Este módulo es la razón por la que a un nuevo nodo que se añade al espacio de símbolos se le denomine *extensión del espacio de símbolos*.

Un fólder especial está diseñado para hacer que la información del sistema esté disponible a través de un interfaz de usuario aceptable. En la mayoría de los casos, ésto significa que el fólder proporciona una vista de su contenido que es más o menos consistente con la vista típica que ofrece un directorio. Por supuesto, esta clase de información depende del tipo de fólder.

Los fólders especiales también pueden contener archivos, de la misma manera que los directorios ordinarios, sin embargo, los representan de una manera ligeramente diferente, es

[3] El control List View es uno de los controles comunes de Windows y representa la información de cuatro maneras diferentes: vistas en miniatura, mosaicos, iconos, lista o presentar los detalles de los objetos.

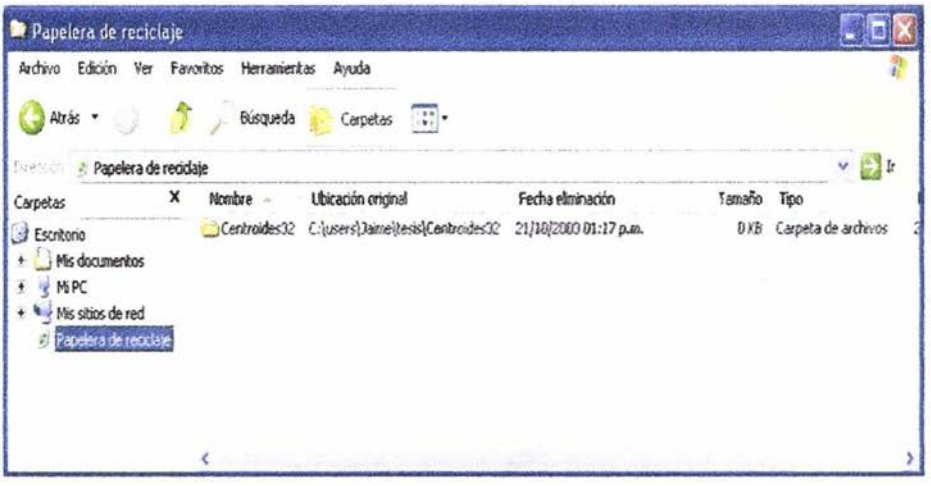


Fig. 3.4 La papelera de reciclaje es un tipo de fólder especial.

decir, mostrando diferentes atributos. Esto es debido a que un fólder especial le da a un archivo un significado diferente y no lo trata como una entrada normal del sistema de archivos, por esta razón es por la que se les llama especiales. La Papelera de Reciclaje, por ejemplo, puede contener archivos ordinarios que sean ocultos, ya que está diseñada para mostrar la lista actual de archivos que han sido marcados para borrarse y presenta otros atributos, como la ubicación original y la fecha de borrado de los mismos, como se puede observar en la figura 3.4.

La mayoría, pero no todos los fólders especiales se encuentran alojados en directorios físicos de uno de los discos. Normalmente se trata de un directorio de sólo lectura cuyo contenido es todo el necesario para desplegar la información de la mejor manera.

O desde otro punto de vista; la mayoría de los fólders especiales necesitan un directorio para almacenar sus datos. Dicho directorio puede estar en cualquier parte del disco y representa la unión de ese fólder con el resto del shell, es decir, representa la localización en el espacio de símbolos donde el fólder ha sido colocado. El contenido de este directorio no necesariamente se muestra como una lista de archivos, sino que solamente se interpreta y se presenta de ésta manera.

La habilidad que tiene un fólder para contener absolutamente cualquier cosa nos lleva a otro par de conceptos importantes que veremos a continuación: objetos de archivo y PIDLs.

Objetos de archivo (File Objects)

Un *file object* u *objeto de archivo* es un elemento que se encuentra en un fólder genérico y puede ser un archivo, un registro, un bloque de memoria, un dispositivo conectado, entre otros. Los términos "*folder items*", "*elementos de fólder*" y "*objetos de archivo*" son expresiones equivalentes que hacen referencia a los elementos individuales que están contenidos en un fólder. Si el fólder en el que se encuentra es un directorio, entonces el objeto archivo no resulta ser más que un archivo. Por tanto, la palabra archivo, es más específica que objeto de archivo ya que se refiere a una entrada específica en el sistema de archivos. Así, un archivo es un objeto de archivo, sin embargo, no todos los objetos de archivo son necesariamente archivos.

Por otra parte existe un problema que yace detrás de los conceptos generalizados de fólder y elementos de fólder. ¿De qué manera podemos garantizar la unicidad de cada elemento dentro del espacio de símbolos en forma segura? Si el shell coincide con el sistema de archivos, entonces el *nombre totalmente calificado* del archivo nos proporcionaría una excelente garantía de unicidad. Sin embargo, no nos sería posible tener dos archivos con el mismo nombre y ruta. De ésta manera, cuando un fólder se vuelve algo más general que un directorio de archivos, se necesita una manera más general de identificar esos elementos.

PIDLs

Un *PIDL* (acrónimo en inglés para *Pointer to an IDentifier List*) o *apuntador a una lista de identificadores* es una estructura de datos que se utiliza para identificar un elemento que se encuentra en un fólder de manera única y es mucho más versátil que un nombre totalmente calificado; garantizando la unicidad de cada elemento no sólo dentro del fólder mismo, sino en todo el espacio de símbolos. Y más importante aún, permite manejar archivos y objetos archivo de manera transparente. Para comprender la estructura y el rol de los PIDLs analizaremos su estructura binaria y la compararemos con los nombres de rutas que reemplazan.

Un *nombre totalmente calificado* es una cadena con un formato muy particular: Está compuesto de una concatenación de subcadenas, cada una de las cuáles identifica a un nivel en la jerarquía del sistema de archivos. Entonces, tenemos primeramente el nombre de la unidad, luego el o los directorios que forman la ruta, el nombre del archivo y finalmente la extensión del mismo, todos ellos separados por diagonales invertidas y la extensión

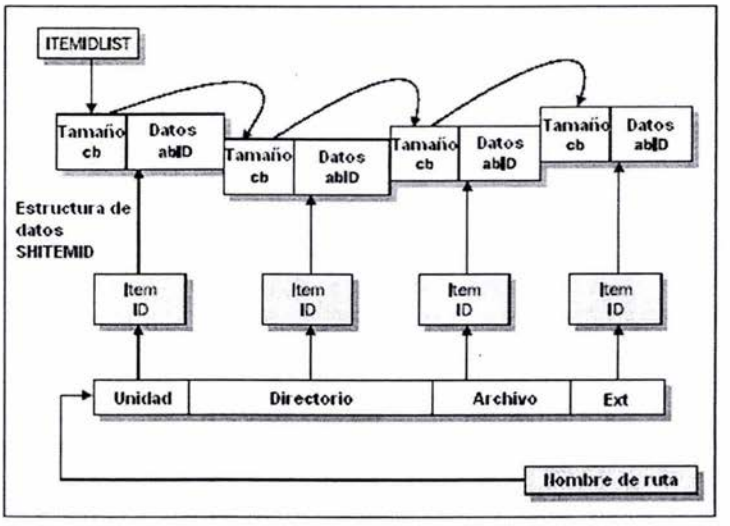


Fig. 3.5 Relación entre un nombre de ruta y un PIDL.

mediante un punto. Lo que generalmente percibimos como nombre totalmente calificado no es más que un apuntador a todos los elementos concatenados, en este caso un apuntador a una cadena. Conceptualmente, lo podemos ver como un apuntador a un arreglo de estructuras, cada una de las cuáles identifica a un elemento en la ruta del archivo.

La figura 3.5 ilustra la relación entre un nombre de ruta y un *PIDL*. Al mismo tiempo da una idea de cómo un identificador de lista está organizado en memoria.

Los objetos intermedios que forman el nombre de ruta son mapeados a los elementos identificadores (ITEM ID) que forman un *PIDL* y son interpretados a través de una estructura SHITEMID, como se ve a continuación:

```
typedef_ struct _SHITEMID
{
     USHORT cb;
     BYTE abID[1];
} SHITEMID, *LPSHITEMID;
```

dónde cb denota el tamaño del elemento identificador, es decir, el número de bytes tomados por los datos asociados con el elemento y usados para identificarlo y el miembro abID es el identificador del objeto.

La cadena de identificadores de elemento traza la ruta que va desde la raíz del espacio de símbolos hasta un elemento específico que se encuentra en un fólder determinado. La lista

de identificadores reúne todos los elementos de la cadena y representa una manera de distinguir a un elemento que es único en todo el shell.

Debido a que los identificadores de elemento están constituidos por estructuras binarias que no tienen un significado concreto para el usuario, el shell utiliza "*nombres de presentación*" para cada objeto. Los nombres de presentación son los nombres con los que vemos a los objetos cuando se utiliza la ventana del Explorador.

De la misma manera que las rutas de nombres, los PIDL pueden ser absolutos o relativos. Aquellos PIDL que son descendientes directos del Escritorio, tales como Mi PC y Mis sitios de Red se considera que tienen PIDL absolutos y sólo contienen un SHITEMID. Aunque en realidad éstos son relativos al objeto Fólder Escritorio, se les denomina absolutos porque este objeto es la raíz del espacio de símbolos. Por su parte, el PIDL del Panel de control no es absoluto ya que su objeto Fólder padre es Mi PC. Similarmente, un objeto de archivo del Panel de control tiene un PIDL relativo a éste.

3.3.2 La vista del shell

El contenido de cualquier fólder se despliega en el Explorador de Windows por medio de un objeto denominado *vista del shell*. Cada fólder define su propio objeto de vista del shell y le delega todas las tareas que lo relacionan con su interfaz de usuario. El objeto vista del shell de un directorio se implementa utilizando un control List View cuyos elementos son los nombres de los archivos y de los subdirectorios. La vista del shell por omisión asigna un icono, un nombre de despliegue y un nombre de tipo a cada archivo con el que trata.

3.3.3 El espacio de direcciones del shell

Introducir código en el contexto o espacio de direcciones de otro proceso es importante debido a que permite tener acceso a aquellos objetos del otro proceso a los que no se permite manejar de manera directa. Esto resulta de gran trascendencia cuando hablamos de trabajar con el shell, ya que se puede solicitar acceso a las interfaces del mismo, modificar la interfaz de usuario, entre otras.

3.3.4 Asignación de memoria del shell

Al trabajar con el shell tarde o temprano se requiere que haya alguna asignación de memoria dentro de su espacio de direcciones y para esto el shell mismo provee un servicio de asignación de memoria que toma a la interfaz **`IMalloc`**.

Para obtener una referencia a este objeto se utiliza la función *SHGetMalloc*, que regresa una referencia al objeto **IMalloc** mantenida por el sistema. Con este apuntador se puede liberar la memoria que ha sido asignada por el shell de manera segura.

3.3.5 La barra de tareas del shell

La ventana de la barra de tareas es una parte bien conocida de la interfaz de Windows, ya que contiene el botón de Inicio. Sin embargo, lo que llamamos la barra de tareas de Windows es, de hecho, un caso especial de la familia de ventanas de barras de herramientas de aplicaciones del escritorio (application desktop toolbars). El mejor ejemplo de éstas es, tal vez, la barra de acceso directo de Office. Existe un conjunto específico de funciones y mensajes que acceden a las barras de herramientas del escritorio, pero, sólo unas cuantas afectan a la barra de tareas de Windows; como resultado de esto la barra de tareas y la barra del escritorio se consideran como objetos diferentes.

Otra concepción errónea sobre la barra de tareas es que se cree que contiene tantos botones como aplicaciones que se encuentren corriendo, sin embargo, esto resulta falso por las siguientes razones:

No todas las aplicaciones que se encuentran corriendo son mostradas en la barra de tareas.

Los únicos hijos de la barra de tareas que son botones son el botón de Inicio y el que despliega todos los elementos del área de notificación.

Entonces, lo que parece ser una colección de botones es realmente un control de tabulación con un estilo especial de tipo botón.

En si, el papel que juega la barra de tareas es el de una consola de sistema, es decir, permite el acceso a todos los programas que se encuentran corriendo.

3.4 La interacción del Explorador de Windows con el espacio de símbolos

Detrás de las escenas, cada fólder que el Explorador despliega se representa como un objeto COM, llamado un *objeto Fólder* o internamente denominado objeto *ShellFolder*. Cada vez que el usuario interactúa con un fólder o con su contenido, el shell se comunica con el objeto fólder asociado por medio de una de varias interfaces estándar del sistema. Dicho objeto hace lo que sea necesario para responder a la acción del usuario y el shell actualiza

la información que se despliega en el Explorador. Para realizar esto, el Explorador de Windows provee a los usuarios de una interfaz grafica (GUI) que les permite realizar varias tareas, como:

- *Navegar* a través de la jerarquía del espacio de símbolos y ver el contenido de los directorios y carpetas virtuales.
- *Administrar* el contenido del espacio de símbolos al mover, borrar y copiar objetos.
- *Recuperar* una variedad de información acerca de esos objetos.
- *Ejecutar* aplicaciones.

La interfaz gráfica del Explorador posee cinco componentes básicos. La figura 3.6 muestra esos componentes y dónde son desplegados comúnmente.

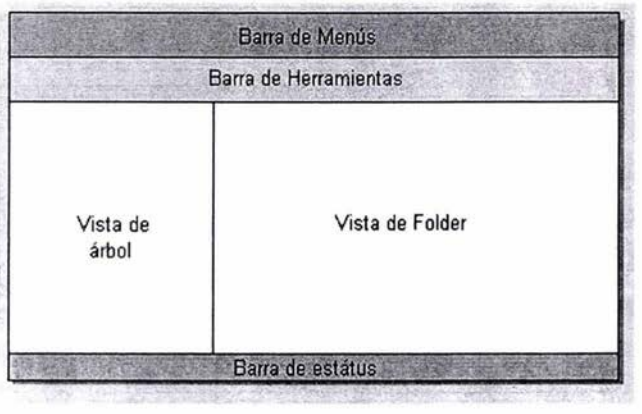


Fig 3.6 Componentes de la interfaz gráfica del Explorador de Windows.

Vista de Árbol. La vista de árbol proporciona una vista general del espacio de símbolos. Esta región alberga un control Tree View[4] que despliega cada objeto Fólder que se encuentra en el espacio de símbolos así como la posición que ocupa en la jerarquía del mismo. En esta vista se puede: desplegar u ocultar el siguiente nivel en el espacio de símbolos, copiar, mover o borrar carpetas, dar un clic derecho con el ratón y desplegar un submenú, y seleccionar una carpeta y ver el contenido de ésta en la vista de Fólder.

[4] El control Tree View es otro de los controles comunes de Windows y representa la vista jerárquica de los objetos Fólder.

Los objetos Fólder utilizan su interfaz **`IShellFolder`** para comunicarse con la que la vista de árbol. Por ejemplo, cuando el usuario da un clic en el signo de más (+) que se encuentra a un lado de un determinado icono de un fólder, el Explorador expande la rama y muestra los subfólders que contiene dicho fólder. Para obtener la información necesaria para actualizar la vista de árbol, el shell realiza varias llamadas a la interfaz **`IShellFolder`** del objeto fólder seleccionado y así obtiene información sobre: los atributos del objeto fólder, además enumera los contenidos del mismo, solicita desplegar los nombres de cada subfólder y solicita un icono que despliega junto a cada fólder. El Explorador entonces, actualiza la vista para mostrar los subfólders del fólder elegido. Esta es la manera de navegar por el espacio de símbolos para recuperar una variedad de información acerca de los objetos.

Vista de Fólder. Ésta corresponde a la vista del shell mencionada en una sección anterior. Cuando un usuario selecciona un fólder, el contenido del mismo es desplegado en la vista de Fólder. La funcionalidad de esta vista se traslapa con la de la vista de árbol, ya que los usuarios pueden copiar, borrar y mover, cambiar las propiedades de una carpeta, ver el contenido de una subcarpeta, etc, sin embargo, existen algunas diferencias entre ambas: en la vista de fólder se despliega el contenido de una sola carpeta, la vista de fólder despliega a los objetos como si se tratara de objetos fólder, la vista de fólder puede desplegar mucha más información acerca de los objetos que la vista de árbol y la vista de fólder permite a las extensiones del espacio de símbolos tener control casi completo sobre qué información es desplegada y de qué manera se hace.

Barras de Menú y de Herramientas. Al igual que muchas aplicaciones de Windows, el Explorador dota al usuario con una colección de herramientas. Una completa selección de se encuentra disponible a través de la barra de menú y aquellas herramientas que son usadas más comúnmente también se encuentran representadas por botones o cajas de edición en la barra de herramientas.

Barra de estado. Finalmente la barra de estado despliega información acerca del objeto que se encuentra actualmente seleccionado.

3.4.1 Navegando por el espacio de símbolos

En este punto ya poseemos los elementos necesarios para navegar en cualquier punto del espacio de símbolos. La manera más simple de comenzar es partiendo de la raíz del espacio de símbolos, es decir, del Escritorio, para lo cuál usamos la función *SHGetDesktopFolder* para recuperar su interfaz **`IShellFolder`**, luego para navegar hacia el resto del espacio de símbolos se enumera el contenido del fólder, se determina que objetos son subfólders y se selecciona uno, del que se toma su respectiva interfaz **`IShellFolder`** y se continua descendiendo de ésta manera hasta llegar al fólder deseado del que se recuperará su contenido y se mostrará una variedad de información de sus elementos.

3.4.2 Administración del sistema de archivos

Como el espacio de símbolos no es estático y las aplicaciones comúnmente necesitan administrar el sistema de archivos, el shell es el encargado de proporcionar varias maneras de realizar tal administración. Permite entre otras cosas que se lleven a cabo las operaciones básicas de copiar, mover, borrar y renombrar objetos. Asimismo, permite otras capacidades adicionales como la administración de los archivos conectados: los documentos HTML frecuentemente tienen un número de archivos asociados con ellos, como son gráficos u hojas de estilos, entre otros; cuando el documento primario es movido o copiado, sus archivos relacionados también son movidos o copiados automáticamente con él.; la administración de archivos por usuario: en sistemas que trabajan con varios usuarios, los archivos se administran restringidamente, es decir, cada usuario puede administrar sus propios archivos pero no los que pertenecen a otros usuarios y; por último, llevar el control de los archivos recientemente utilizados al añadirlos a la lista de documentos recientes cuando se modifican o se crean por primera vez.

Moviendo, copiando, renombrando y borrando archivos

El shell proporciona la función *ShFileOperation* que es la responsable de llevar a cabo todas las operaciones con archivos, como son:

- Copiar un objeto a otra carpeta.
- Mover un objeto a otra carpeta.
- Borrar un objeto.
- Renombrar un objeto.

Dicha función toma uno o más archivos fuente que se le proporcionen y produce sus correspondientes archivos destino. En caso de que se trate de una operación de borrado, el sistema intenta poner los archivos borrados en la papelera de reciclaje.

Administración de Directorios

De la misma manera que se administran los archivos en el shell también es posible realizar las mismas operaciones básicas con los directorios.

La creación de directorios se lleva a cabo mediante la funciones *CreateDirectory* de la API de Windows.

Notificación de los eventos del shell

Seguramente habrá notado que el Explorador es muy rápido para detectar cualquier cambio en el sistema de archivo. Periódicamente refresca la vista actual y refleja cualquier cambio que otras aplicaciones hayan provocado. Por ejemplo si se abre una ventana de comandos de DOS y una ventana del Explorador, se selecciona el mismo directorio y luego se crea un directorio en la primera, la segunda se actualizará un instante después sin intervención alguna.

Parece que algo le indica al Explorador que un nuevo fólder ha sido creado. Detrás de todo esto se encuentran los *objetos de notificación*.

Un *objeto de notificación* es un objeto altamente especializado que automáticamente es señalizado cuando detecta algún cambio en el espacio de símbolos. Por medio de un objeto de notificación se puede poner un directorio, un subárbol y más aún una unidad de sistema bajo control y observar varios eventos que se relacionan con los archivos y los fólders, como creación, renombrado, borrado, cambios de atributos, etcétera.

De manera general, el Explorador establece un objeto de notificación en el directorio que se encuentra actualmente desplegado. Cada vez que recibe una notificación de que algo ha cambiado, recarga el contenido del fólder para reflejar dichos cambios. Si lo pensamos bien, nos daremos cuenta de que el mecanismo de notificación de mensajes parece estar hecho a la medida de las necesidades del Explorador.

El Explorador no es una utilidad de monitoreo del sistema de archivos; necesita saber si algo en el fólder que esta siendo desplegado actualmente ha cambiado, en el caso de que ese cambio afecte a los datos desplegados, como nombres de archivo o de subfólders, atributos, tamaños, fechas, seguridad, etcétera. No importa la operación exacta, lo que

importa es que algo ha ocurrido. Esto parece ser un buen compromiso entre lo que concierne a la ejecución del Explorador y aquello que concierne al sistema en sí.

Después de que una aplicación lleve a cabo cualquier operación que realice cambios en el espacio de símbolos, como la función *ShFileOperation*, se debe notificar al shell mediante la función *SHChangeNotify* para indicarle que algo en la configuración del sistema ha sido modificado y en respuesta a tales cambios, el Explorador refrescará su interfaz de usuario. De esta manera se establece un canal de comunicación entre las aplicaciones y el Explorador.

3.4.3 Ejecución de aplicaciones

Una vez que tenemos seleccionado un objeto de archivo, el siguiente paso es realizar alguna acción sobre él. Por ejemplo, si el objeto seleccionado es una carpeta podemos explorarla o si es un ejecutable, simplemente abrirlo para que se ejecute. El Explorador utiliza las funciones API *ShellExecute* y *ShellExecuteEx* para poder realizar éstas y otras acciones. A dichas acciones se les denomina *verbos* y esencialmente son los comandos que se encuentran en el menú contextual del objeto seleccionado, como se puede ver en la figura 3.7 para una carpeta. Para encontrar que verbos se encuentran disponibles para un objeto, el Explorador busca primero en el registro el identificador de clase del objeto y luego los verbos que se encuentran disponibles para ese objeto. Cada verbo tiene asociada una acción que será realizada cuando el verbo sea invocado.

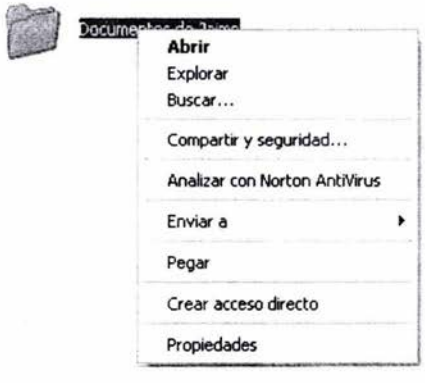


Fig. 3.7 Verbos o elementos del menú contextual de una carpeta.

Los verbos que aparezcan en el menú contextual dependerán del objeto seleccionado, sin embargo, las funciones *ShellExecute* y *ShellExecuteEx* utilizan los que se listan en la tabla 3.1 y que son los que se encuentran disponibles comúnmente.

Verbo	*Descripción*
edit	Ejecuta un editor y abre el documento para editarlo.
find	Inicia una búsqueda comenzando por un directorio especificado.
open	Ejecuta una aplicación. Si el archive no es un archive ejecutable, abre la aplicación asociada.
print	Imprime el documento
properties	Despliega las propiedades del objeto seleccionado.

Tabla 3.1 Verbos que se utilizan en las funciones ShellExecute y ShellExecuteEx.

3.5 Extensiones del shell y extensiones del espacio de símbolos del shell

Windows es un sistema operativo constituido, en gran parte, por componentes COM que se encargan de gestionar los más diversos aspectos. Un desarrollador puede acceder a los mencionados componentes para realizar las tareas que precise, desde añadir un nuevo objeto al escritorio hasta mostrar las páginas de propiedades asociadas. Desde este punto de vista, el software actuaría como cliente respecto a componentes que forman parte del sistema, alojados en servidores COM dentro de proceso.

Lo más interesante es que el sistema operativo cuenta con mecanismos que facilitan su ampliación, permitiendo a los desarrolladores crear sus propios espacios de símbolos, añadir opciones a los menús contextuales de los objetos, crear nuevas páginas de propiedades para ellos, etc.

Crear una extensión para la interfaz de Windows implica crear uno o varios componentes COM que, por regla general, deberán implementar unas interfaces predefinidas. Dichas interfaces son los puntos de unión con el sistema, el enlace entre los componentes COM que forman Windows y los creados por nosotros mismos.

Básicamente, existen dos tipos de extensiones en el mundo del Explorador: *extensiones del shell* y *extensiones del espacio de símbolos*. De manera general las extensiones del shell son código que modifica o mejora la funcionalidad de todo el shell del sistema operativo, es

decir, permiten extender las posibilidades de Windows, añadiendo funcionalidades no contempladas en el diseño original del sistema.. Por otro lado las extensiones al espacio de símbolos son sólo un tipo de extensiones del shell, puesto que añaden nuevos símbolos, que poseen sus propias características, a éste. Por tanto todas las extensiones del espacio de símbolos son extensiones del shell.

Ahora pensemos también en un navegador o browser. Un navegador es una aplicación que interactúa con el espacio de símbolos para proporcionar al usuario una representación visual y un mecanismo para manipular datos. La ventana del Explorador también es un navegador. La meta de las extensiones del espacio de símbolos es eliminar la necesidad de crear un navegador completo, únicamente se crean las interfaces para añadir los nuevos objetos para que puedan ser visualizados y administrados como si fueran parte del shell.

Al integrar la aplicación construida con el shell, se puede extender la funcionalidad del shell mismo y personalizar ciertos aspectos de su comportamiento, como las operaciones básicas llevadas a cabo con los comandos de voz, por lo tanto el presente trabajo es una extensión del shell.

La particularidad de las extensiones del shell es que permiten manipular al shell como si la aplicación desarrollada fuera parte de él. Tenemos entonces, que una extensión a la interface de Windows es un componente COM, concretamente alojado en una librería de enlace dinámico. Esto es una condición indispensable, puesto que el sistema necesitará ejecutarlo en su propio espacio de direccionamiento o proceso. Entonces, arquitectónicamente, un componente que actúa como extensión del sistema no se diferencia de cualquier otro componente y sabiendo que en el sistema existen multitud de componentes instalados, ¿cómo sabe Windows cuáles tiene que utilizar como extensiones?

Es aquí donde entra en juego el registro de Windows. Es en el registro donde se indica que un cierto tipo de archivo, cuenta con una extensión que añade opciones a su menú contextual o nuevas páginas de propiedades. Es en el registro donde se especifica si una cierta carpeta tiene sus operaciones controladas por una extensión, o si existe una extensión que actúa como un nuevo espacio de símbolos, etcétera.

Crear una extensión, por tanto, exige conocer tres elementos diferentes: las interfaces que debe implementar la extensión, las interfaces que Windows pondrá a su disposición para

comunicarse con el objeto que extiende y, por último, las entradas del registro que es preciso añadir para que la extensión sea funcional.

3.5.1 Tipos de extensiones del espacio de símbolos

Existen dos aspectos a tratar con respecto a las extensiones del espacio de símbolos antes de continuar con el tema: extensiones que tienen raíz en comparación con extensiones que no tienen raíz y su el punto de unión.

La diferencia entre extensiones que tienen raíz y aquellas que no tienen raíz es la manera en cómo se utilizan. No existe diferencia de código entre ambas. Las extensiones que no tienen raíz permiten al usuario subir un nivel a partir de la raíz de la extensión, mientras que las que tienen raíz no lo permiten. Esencialmente en las extensiones que tienen raíz, la extensión misma es la raíz del árbol y únicamente se pueden ver las ramas que posee. Por ejemplo, en la figura 3.8 se ve que el Menú de Inicio depende de un directorio padre al que se puede acceder si se desea y que se establece de acuerdo con el usuario. Como ejemplo de las extensiones que no tienen raíz véase la figura 3.9 en donde se puede ver que Mi PC es la extensión y que si se da doble clic sólo se pueden ver las ramas que posee, en este caso las carpetas de documentos personalizados y las unidades de disco.

El *punto de entrada* o *punto de unión* de una extensión del espacio de símbolos

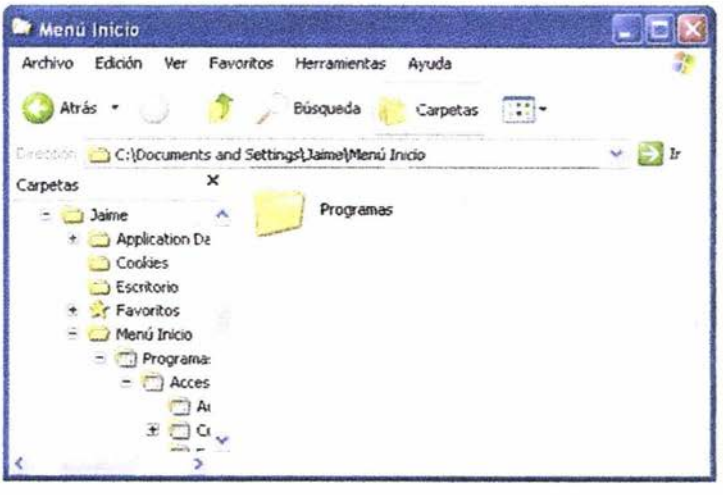


Fig 3.8 Extensión del espacio de símbolos que no tiene raíz.

corresponde al punto donde el Explorador de Windows se conecta y accede a la respectiva extensión y no resulta ser más que la raíz de la extensión.

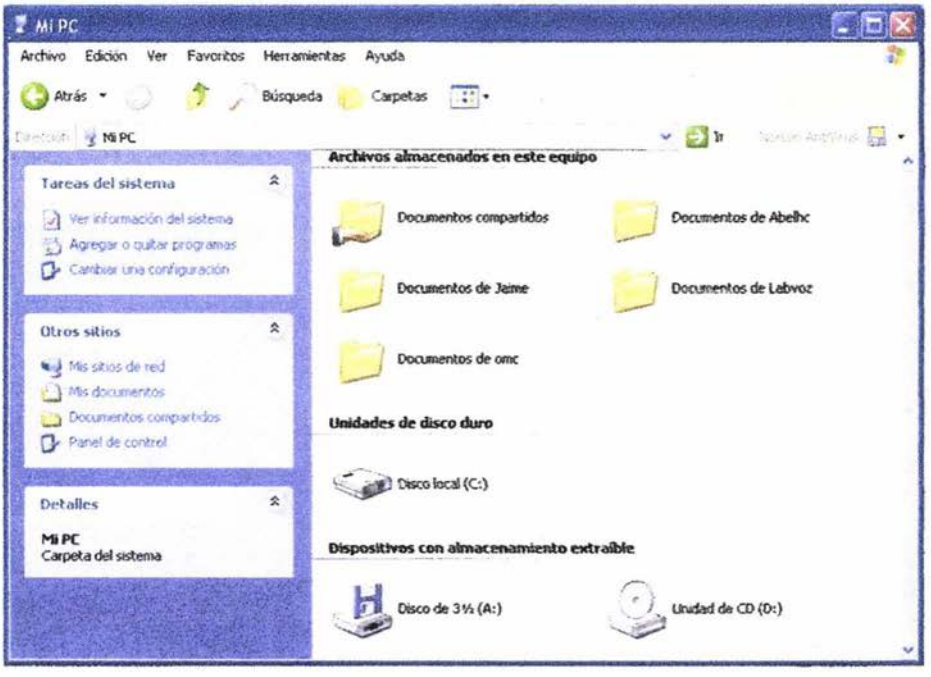


Figura 3.9 Extensión del espacio de símbolos con raíz

3.5.2 Cómo trabaja una extensión del espacio de símbolos

Una extensión al espacio de símbolos consiste de dos componentes básicos:

- Un administrador de datos.
- Una interface entre el administrador de datos y el Explorador.

El primer componente se le deja al desarrollador, ya que él es el quién decide la mejor manera de almacenar y administrar sus datos. El segundo componente se trata del código necesario para empaquetar los datos como objetos folder y manejar la interacción con el Explorador.

Se puede entender cómo se maneja una extensión del espacio de símbolos si se tiene clara interacción que existe entre el Explorador y el espacio de símbolos, ya que cuando se carga

una extensión el Explorador la ve como un objeto más que pertenece al espacio de símbolos y lo maneja como tal. De esta manera se garantiza su correcto funcionamiento.

En la figura 3.10 se observa la interacción del Explorador con el espacio de símbolos de shell y con una extensión de este espacio por medio de sus interfaces. Se puede ver, también, que el Explorador utiliza la interface `IShellFolder` para acceder a los objetos fólders, como se mencionó anteriormente. Cabe mencionar que se presentan dos interfaces muy importantes además de la interfaz `IShellFolder`, las interfaces `IShellView` e `IShellBrowser`. La primera se ha explicado de manera implícita, pues es la interfaz de la vista del shell, que permite desplegar en su control List View, los objetos del espacio de símbolos y ahora también los de la extensión misma. La segunda interface se encuentra implementada dentro del Explorador y es la encargada de hacer que la extensión se comunique con el Explorador, con una visible retroalimentación al usuario, que incluye menús, textos de estatus y barras de herramientas. Asimismo, `IShellBrowser` proporciona un flujo privado para almacenar la información que se guarda en el registro,

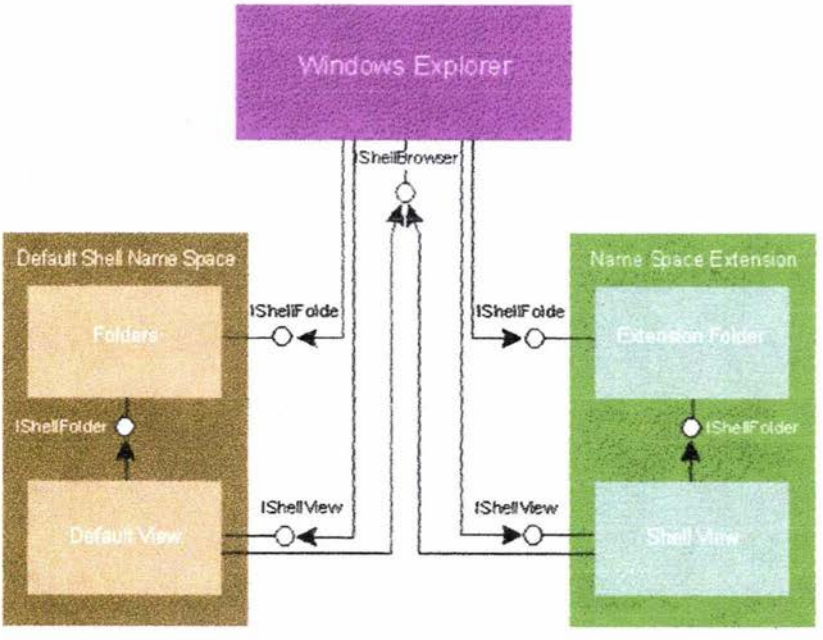


Fig 3.10 Interacción del Explorador con el espacio de símbolos y con una extensión del espacio de símbolos.

como la orientación de los elementos, el modo en que se mostraron esos elementos por última vez, o cualquier atributo que la extensión pudiera necesitar la próxima vez que el espacio de símbolos sea explorado.

3.5.3 Objetos de instancia del shell (shell instance objects)

Un objeto de instancia del shell es una clase especial de extensión del shell proporcionada por el componente ***shdocvw.dll***. A diferencia de las extensiones del shell tradicionales que requieren de un componente COM dentro de proceso para implementar un objeto. Un objeto de instancia del shell recupera todo lo que necesita saber del registro.

Cuando se le pregunta a ***shdocvw.dll*** que cree un objeto que no reconoce de otra manera, revisa la clave del registro buscando una subclave denominada *Instance*, si la encuentra regresa el CLSID para esa subclave a la función CoCreateInstance, y después regresa una instancia de ese objeto ya inicializada. De esta manera, ***shdocvw.dll*** permite trabajar con extensiones de manera mucho más sencilla y segura.

El componente ***shdocvw.dll*** es un control ActiveX que pone a disposición del cliente objetos que son manejados comúnmente por el shell de Windows, tales como los navegadores del shell, entre los que se incluyen los Fólders del sistema de archivos y el Explorador de Internet de Microsoft. De hecho la mayor parte de la funcionalidad del Explorador de Internet está contenida en este control[5]. Por medio de las interfaces **`IShellWindows`** e **`IWebBrowser2`** y de sus respectivos métodos es posible, no sólo acceder a los navegadores, sino también automatizar tanto al Explorador de Windows como al Explorador de Internet. Por ejemplo, la interfaz **`IShellWindows`** proporciona la colección de las ventanas activas de los navegadores del shell, entre éstas se incluyen aquellas ventanas que son del tipo que usan tanto el Explorador de Windows como Mi PC y las ventanas del Explorador de Internet[6]. Con la interface **`IWebBrowser2`** podemos obtener un apuntador y navegar en un fólder del sistema de archivos como si se tratara de otra página web más que estuviera siendo manejada por el Explorador de Internet. La diferencia fundamental entre el Explorador de Windows y el de Internet, aparte del tipo de

[5] David J.Kruglinski, et al. Programación avanzada con Microsoft Visual C++ 6.0. Pág. 179.

[6] El tipo ventana del Explorador de Windows es ExploreWClass, el de Mi PC es WCabinetClass y el del Explorador de Internet es IEFrame. Sin embargo, la clase de ventana de Mi PC se deriva de la del Explorador de Windows.

ventana, radica en el tipo de documento que manejan cada uno, el primero utiliza un documento de Vista de Fólder, al que se accede por medio de su interfaz **`IShellFolderViewDual`**, y el segundo, un documento de tipo HTML, que se auxilia de la interface **`IHTMLDocument2`** para acceder al documento mismo. Una vez obtenida la intefaz **`IShellFolderViewDual`** se utilizan sus métodos para obtener los elementos que se encuentran seleccionados y realizar las operaciones correspondientes.

En el presente trabajo se hace uso de este control para automatizar los comandos de voz de Windows, lo que permite que el sistema operativo sea quien gestione la vista de Fólder del Explorador concretándonos únicamente a trabajar con su contenido.

Capítulo 4

Comandos de voz

De manera general el presente sistema, Comandos de voz, consiste en la implantación de un cliente COM que se comunica con las distintas interfaces que proporcionan Windows y Word enviándoles los respectivos comandos que son reconocidos, como se muestra en la figura 4.1 a). En este esquema podemos observar que el sistema consiste de 4 módulos principales: Módulo de reconocimiento, módulo de entrenamiento, módulo de comandos de Windows y de Word y módulo de grabación de comandos. Aunque aquí se observe que los dos primeros estén contenidos en el mismo módulo esto es solamente con el fin de dar a conocer cómo está constituido el sistema y por la estrecha relación que existe entre ellos, como se verá posteriormente.

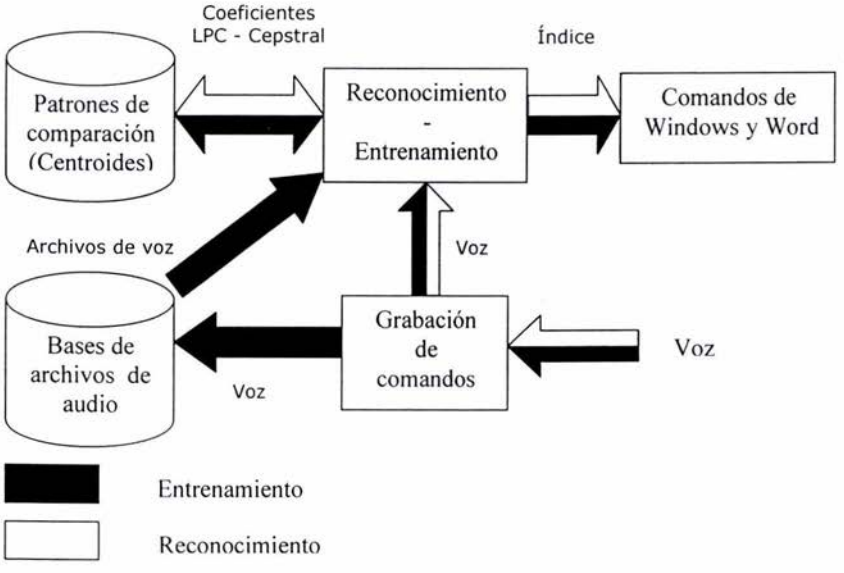


Fig. 4.1 a). Esquema general del sistema Comandos de voz.

Asimismo podemos observar que existen dos posibles rutas que se emplean en el sistema. Por una parte la del entrenamiento, que se puede observar en la figura 4.1 b), donde inicialmente se graban y almacenan los comandos que se utilizan como bases de audio, con sus respectivas repeticiones. A continuación, por medio de la cuantización vectorial, se

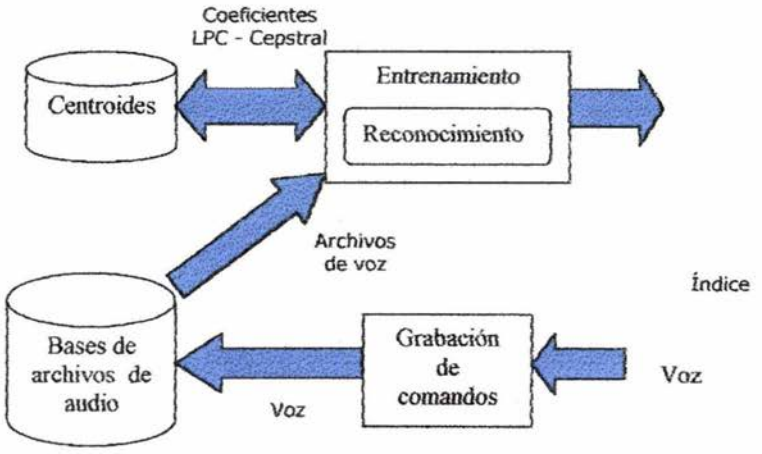


Fig. 4.1 b) Esquema general del entrenamiento de los comandos.

obtienen los coeficientes LPC-Cepstral representativos de cada comando que se utilizarán como patrones de comparación en el reconocimiento. Como parte del entrenamiento establecí que se realizara un reconocimiento de prueba de cada una de las repeticiones de los comandos para ver la aproximación o certeza de éstos y así poder retroalimentar al sistema.

Por otra parte, existe la ruta del reconocimiento, que se puede observar en la figura 4.1 c), en donde el comando que se pronuncia es procesado para obtener, también, sus respectivos coeficientes LPC-Cepstral los que se utilizan para comparar la distancia que existe entre éstos y los patrones de comparación obtenidos durante el entrenamiento. Por supuesto los coeficientes cuya distancia sea mínima son los que más se aproximan al comando pronunciado. Así se obtiene el índice de esa aproximación, que es el valor que se le pasa al módulo de Comandos de Windows y Word para la ejecución de los mismos.

Cabe mencionar que para lograr un mejor desempeño del sistema los módulos de grabación, de reconocimiento y de entrenamiento se encuentran implantados como hilos de trabajo de Windows, es decir, se realizan subprocesos que se encuentran separados del hilo principal[1], que es el que maneja la ventana principal del sistema Comandos de voz, y

[1]Un proceso es, prácticamente, un programa en ejecución y un hilo es un subproceso o un proceso hijo de un proceso principal. Cada proceso generalmente posee al menos un hilo, denominado hilo principal.. Asimismo un proceso puede tener varios hilos que realicen tareas diferentes prácticamente al mismo tiempo. Véase Andrew S. Tanenbaum. Sistemas Operativos modernos, pág. 72

únicamente envían los resultados al sistema por medio de mensajes para que éste los procese. Sin embargo, el manejo de procesos e hilos de trabajo del sistema operativo Windows se encuentran fuera del alcance de este trabajo[2]. Y en cuanto al manejo de mensajes de ventanas en Windows[3] más adelante sólo se dará una breve descripción para mostrar cómo funcionan los comandos de voz ya que también se encuentra fuera del alcance de éste trabajo.

Fig. 4.1 c) Esquema general del reconocimiento de voz.

4.1 El modelo de objetos utilizado en el entrenamiento y en el reconocimiento

De manera general, para poder realizar tanto el entrenamiento como el reconocimiento estructuré el sistema con un objeto principal: **Comando**, que corresponde a cada uno de los comandos que se desean entrenar o reconocer. Cada comando, a su vez, contiene la colección de las M repeticiones de las palabras de entrenamiento y cada palabra se encuentra asociada con un objeto **Voz** o señal de voz, que por ejemplo, utiliza las funciones *Preenfasis e Inicio_Fin* para realizar el filtrado de preénfasis y para obtener el inicio y el fin de cada señal, respectivamente. En realidad antes de realizar el filtrado de preénfasis, que se aplica para acentuar las frecuencias altas de la señal de voz, se le aplica a la señal un filtro pasobajas con el fin de limitar el intervalo de frecuencias ya que la mayor cantidad de

[2] Véase Jorge Pascual. Programación avanzada en Windows 2000. pág. 602.
[3] Idem. Pág. 45

energía de una señal de voz se encuentra por debajo de los 5 kHz. y para eliminar el ruido en altas frecuencias.

Por otra parte, se tiene que cada señal de voz puede ser dividida en N objetos **Trama** que toma un conjunto de n muestras para formar cada trama. Entre las funciones que proporciona el objeto **Trama** se encuentran *MagnitudPromedio* y *CrucesporCero*, que calculan la magnitud promedio y la tasa de cruces por cero de cada señal, respectivamente y que son utilizadas por los objetos **Voz** en la obtención del inicio y el fin de la señal. También se encuentra el método *Cepstral* que se encarga del cálculo de los coeficientes LPC-Ceptral para cada trama y que es invocada por los objetos **Voz** para obtenter dichos coeficientes de todas sus tramas. Una vez obtenidos los coeficientes, se procede a llamar a la función *Segmentacion* para realizar la segmentación lineal de las señales de voz. El entrenamiento consiste en llamar a la función *Cuantiza* de cada objeto **Comando**, que realiza el agrupamiento de los segmentos por medio de la función *Kmedias,* que como su nombre lo indica utiliza el algoritmo de agrupamiento K-Medias para obtener el respectivo codebook de cada segmento. Para que haya una mejor estimación de los centroides, la función *Kmedias* toma los vectores de manera aleatoria. Finalmente se asocian los patrones de comparación con su respectivo comando y se guardan en su correspondiente archivo.

En el reconocimiento se toma la señal de voz grabada y se asigna como un objeto **Comando** que se va a reconocer. De ésta manera se le aplican las mismas funciones para obtener su correspondientes patrones de comparación y se le divide también en cuatro segmentos. Posteriormente se invoca a la función *ReadCentroides* de los objetos **Comando** de los comandos entrenados para que devuelvan los respectivos centroides y se realicen las comparaciones mediante una función de distorsión, en este caso la *distancia Cepstral*, y obtener, así, el índice al que se aproxima más la señal de voz. De esta manera la jerarquía de objetos presentada permite que se realicen el entrenamiento y el reconocimiento de forma más rápida.

4.2 El módulo de reconocimiento

Es aquí donde se realiza el reconocimiento de los comandos pronunciados, como se muestra en la figura 4.2, donde aparece el módulo de reconocimiento al inicio de la aplicación.

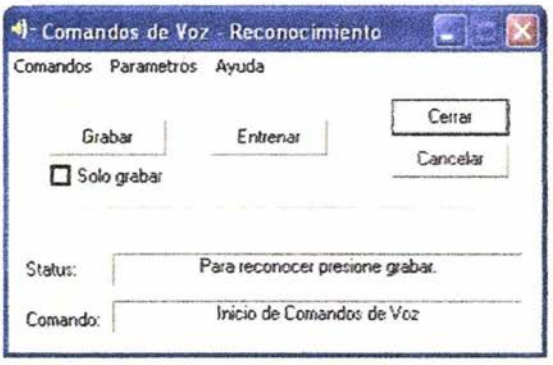


Fig. 4.2 Módulo de reconocimiento.

Para iniciar el reconocimiento basta con presionar el botón *Grabar* que, como ya se mencionó, habilitará un hilo de trabajo que realizará la grabación de la señal de voz que se desea reconocer. Terminada ésta se procede al reconocimiento mediante otro hilo de trabajo que devuelve el índice del comando más aproximado y con el se procede a la ejecución del mismo. Después de esto se está preparado para grabar de nuevo.

El botón de *Cerrar* permite salir de los Comandos de Voz.

El botón *Cancelar* tiene varias acciones dependiendo de la etapa de reconocimiento en la que esté. Con éste se puede detener la grabación, el reconocimiento o la ejecución del comando.

El campo de *Status* presenta información sobre las acciones llevadas a cabo, tales como Grabando, Reconociendo, Comando Ejecutado, etc. Asimismo muestra si se produjo algún error durante el proceso de reconocimiento.

El campo *Comando* es el encargado de presentar el nombre del comando reconocido y ejecutado.

Al seleccionar la opción de *Sólo grabar* se habilita el módulo que permite grabar los archivos de audio que se utilizarán para el entrenamiento.

4.3 El módulo de entrenamiento

En éste módulo se realiza el entrenamiento de todos los comandos con base en los parámetros configurados. En la figura 4.3 se muestra el módulo de entrenamiento. En éste existe la posibilidad de realizar la cuantización ya sea, de todos los comandos, de una selección de comandos o de un rango de ellos, de acuerdo a la opción que se encuentre

seleccionada al inicio del entrenamiento. La opción de *cuantizar todos* permite entrenar al sistema por primera vez. Las dos opciones restantes nos permiten retroalimentar a Comandos de voz en caso de que la precisión de los centroides obtenidos no haya sido muy buena. Por omisión el entrenamiento está seleccionado para cuantizar a todos los comandos.

Los botones de *Inicio* y *Cancelar* nos permiten comenzar y detener el entrenamiento, respectivamente, mientras que el botón de *Cerrar* nos permite salir del módulo de entrenamiento.

En la ventana de *palabras de entrenamiento* se presenta el conjunto de aquellas palabras que se han procesado y obtenido sus respectivos coeficientes LPC-Cepstral. Además indica si existió algún error durante el cálculo de dichos coeficientes. Si la obtención de los coeficientes se realiza exitosamente se procede a mostrar en la ventana de *palabras cuantizadas* el resultado del entrenamiento, donde se incluyen aquellas palabras que fueron agrupadas exitosamente y aquellas que no lo fueron, además de los respectivos patrones de referencia obtenidos.

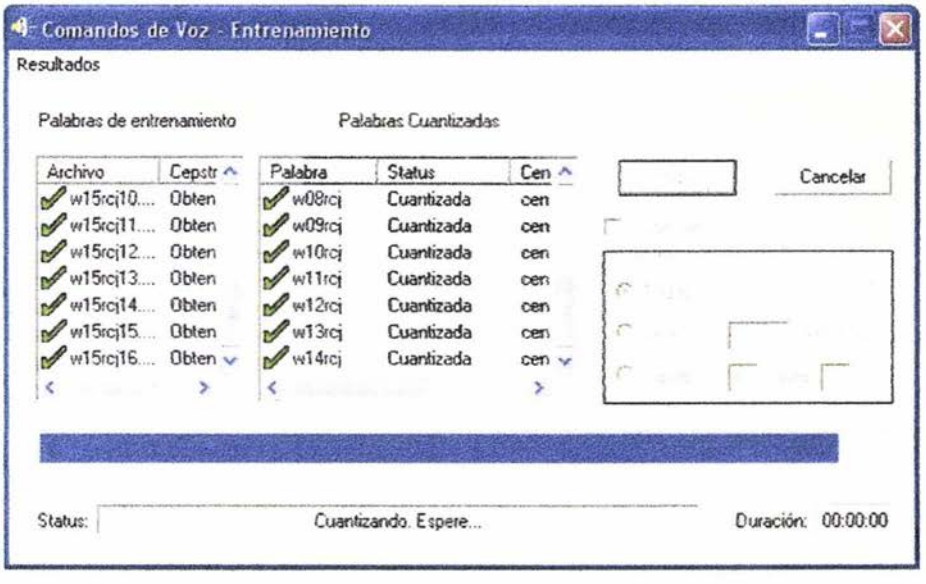


Fig. 4.3 Módulo de entrenamiento obteniendo los centroides de los comandos entrenados.

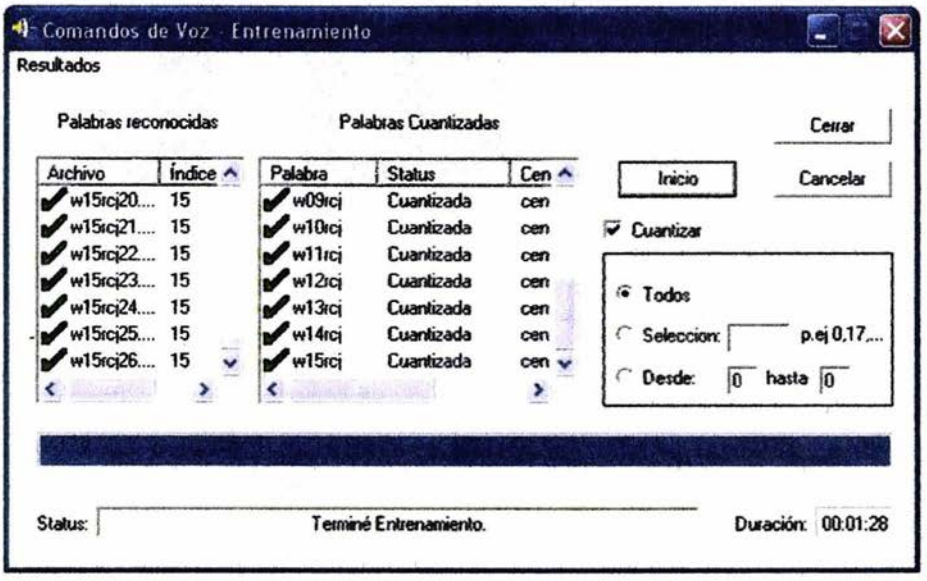


Fig. 4.4 Fase de reconocimiento de los comandos.

Terminada la cuantización se procede a realizar un reconocimiento de prueba de todas las palabras que fueron entrenadas listándolas como palabras reconocidas con su respectivo índice, como se muestra en la figura 4.4.

Los módulos de entrenamiento y de reconocimiento son mutuamente excluyentes ya que no es posible entrenar al sistema mientras se reconoce y viceversa, debido a que utilizan los mismos parámetros.

4.3.1 Configuración de Parámetros

En el software se pueden ajustar tanto los parámetros de reconocimiento como los de entrenamiento, al elegir el menú *Parámetros* y *Configurar* aparece el cuadro de diálogo de la configuración de parámetros, como se muestra en la figura.4.5.

Empecemos con la opción *Comandos de*, en la que podemos elegir qué comandos deseamos entrenar y reconocer: los comandos de Windows, los comandos de Word u otros comandos que deseemos entrenar para algún otro uso. El sistema alterna entre comandos de Windows y comandos de Word. Los comandos de Windows están definidos por omisión y cuando se abre Word automáticamente se cambia la opción a comandos de Word. Más adelante se dará la justificación de este mecanismo.

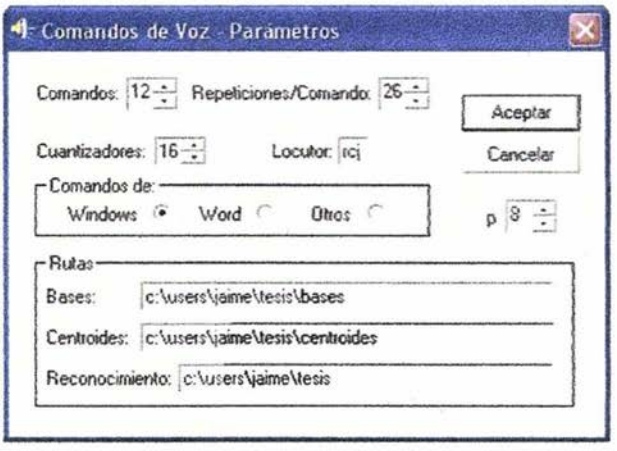


Fig. 4.5 Configuración de los parámetros.

Con la opción *Comandos* se puede ajustar el número de comandos que el sistema va a reconocer y entrenar. El rango de comandos varía desde 1 hasta 50. Por omisión el sistema está configurado con 12 comandos para Windows y 16 comandos para Word, dando un total de 28 comandos que maneja este sistema.

En la opción *Repeticiones/Comando*, se establece cuantas repeticiones de cada comando se tomarán para el entrenamiento. Su rango varía de 1 a 30. Por omisión está configurado a 26 repeticiones por comando.

La opción *Centroides* establece el número de centroides que se van a usar para entrenar al sistema y que serán utilizados para realizar el reconocimiento. Esta opción sólo puede tomar dos valores, 16 ó 32 centroides y por omisión se establece a 16 centroides.

Para que otros locutores puedan utilizar este sistema se pueden tener las iniciales del locutor y así formar los archivos de audio que utilizara el nuevo locutor para sus propias bases.

Existe la opción *p* que es el valor de la cantidad de coeficientes de predicción lineal con los que se obtienen los coeficientes LPC-Cepstral. Y esta opción está reservada para aplicaciones futuras. El valor por omisión de *p* es 8 que es con el que se trabaja comúnmente en procesamiento de voz.

Las *Rutas* establecen las rutas que toma el sistema como referencias: *Bases* es la ruta donde se encuentran los archivos que son grabados para entrenamiento. *Centroides* muestra la ruta

donde se encuentran los parámetros de cuantización resultantes del entrenamiento. Cuando se encuentra habilitado comandos de Word o comandos de Windows, a las rutas de los directorios de los campos *Bases* y *Centroides* se les aunarán los directorios Word o Windows de acuerdo con su correspondiente opción. Además, al directorio de centroides se le aunará el directorio centroides16 o centroides32 dependiendo del número de centroides que se haya seleccionado. Estos directorios serán los directorios reales donde se grabarán las bases y se obtendrán los respectivos centroides. Si se encuentra seleccionada la opción de otros comandos, los directorios que se tomarán como referencias serán los directorios que se lean en sus campos correspondientes.

Para realizar el reconocimiento se graba el comando que se quiere reconocer en un búfer temporal que será procesado para obtener sus respectivos coeficientes LPC-Cepstral, los que compararán con los centroides especificados. Entonces, *Reconocimiento* es la ruta donde se guarda dicho archivo.

4.4 Módulo de grabación

Este sistema cuenta con un módulo adicional de grabación, como se ve en la figura 4.6. Por medio de la interfaz Waveaudio de Windows se puede realizar la grabación de archivos de audio con formato PCM de 16 bits por muestra, un solo canal y frecuencia de muestreo de 11025 kHz. Este módulo ayuda a los módulos de entrenamiento y de reconocimiento.

Se le puede dar cualquier nombre al archivo, sin embargo lo ajusté por medio de los parámetros para responder a las necesidades del sistema. Con esto realizo la grabación de todas las bases que se usan en el entrenamiento y del búfer temporal que se usa en el reconocimiento.

El software consta de un módulo en el que se pueden grabarse uno a uno los archivos de voz en una ruta especificada y con el nombre de archivo especificado, dichos archivos son los que se utilizan como archivos de audio base para realizar el entrenamiento. Los archivos llevan la siguiente nomenclatura:

w*num_palabra***iniciales***num_repetición*.raw

Así, el archivo w05rcj20.raw representa la repetición número 20 del sexto comando del usuario rcj, que se almacena en el directorio Bases. Los comandos los cuenta a partir de cero y las repeticiones las cuenta a partir del uno.

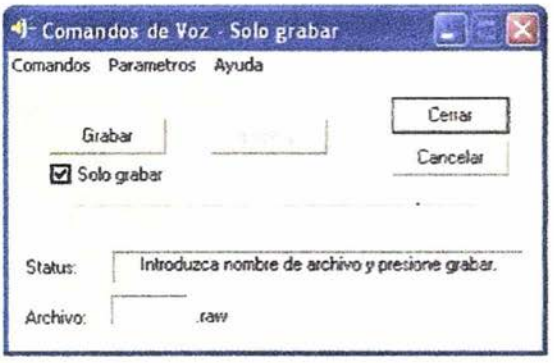


Fig. 4.6 Módulo de grabación – Opción sólo grabar.

4.5 Módulo de comandos de Windows y de Word

Este módulo se encuentra implantado internamente. Como Windows es un sistema operativo orientado a eventos, es decir, el hilo principal de una aplicación espera a que ocurra algún evento y llama a un procedimiento determinado para que maneje tal evento; y que dichos eventos se notifican por medio de mensajes que son enviados directamente a la ventana que posee cada aplicación donde son puestos en su cola de mensajes para que sean procesados se estableció que los comandos de voz también fueran implantados con estos lineamientos, es decir, como eventos que estuvieran asociados con el índice proveniente del módulo de reconocimiento. Así en la figura 4.7 se observa que el mensaje recibido que contiene el índice del comando reconocido se coloca en la cola de mensajes del sistema junto con los demás mensajes que provienen de otros eventos y que pueden ser para volver a dibujar la ventana o para desplegar algún otro tipo de información en el cuadro de diálogo, entre otros[4]. Una vez en la cola de mensajes se invoca al procedimiento que corresponde a cada mensaje. Se puede ver también que se hace una distinción entre comandos de Windows y comandos de Word ya que, como ya se mencionó, se permite que se ejecuten sólo comandos de Windows o sólo comandos de Word.

[4] Para mayor información sobre los tipos de mensajes de ventana que existen en Windows, véase Jorge Pascual, et al. Opus citatum, pág. 259.

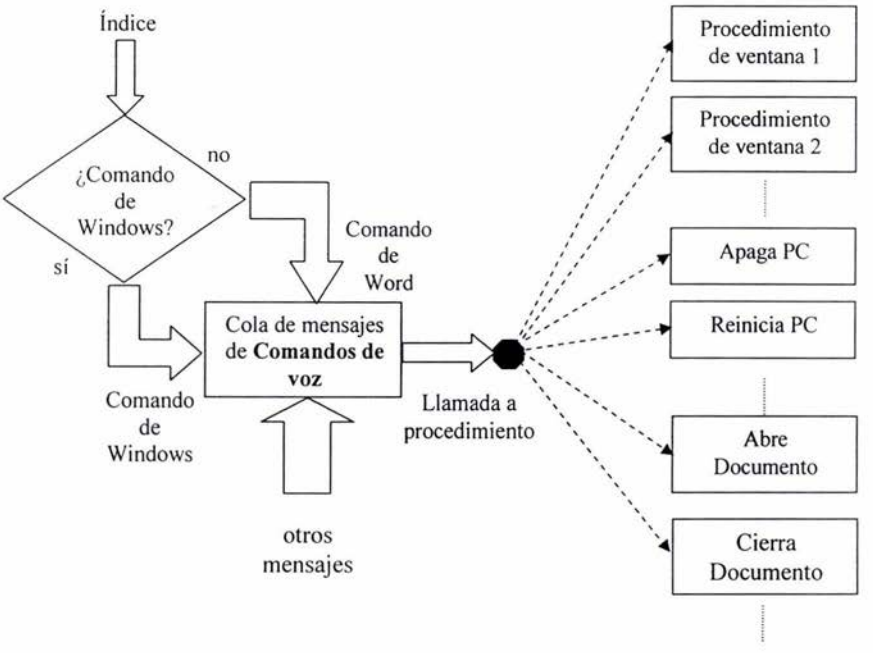


Fig. 4.7 Esbozo del procesamiento de los comandos de voz del sistema.

En lo que respecta a la manera en cómo funcionan los procedimientos asociados a cada comando se han utilizado los conceptos vistos en los capítulos anteriores para poder realizar la automatización tanto de Windows como de Word, es decir, el sistema se comunica por medio de las interfaces que estos últimos poseen y hace uso de los servicios que implementan. En este sistema he implementado solamente dieciséis comandos de Word como una muestra de los servicios que este procesador de palabras nos ofrece, sin embargo, el sistema permite la posibilidad de automatizar completamente a Microsoft Word por medio de su interfaz de despacho.

Asimismo, se ha dado una pequeña muestra de la automatización del shell Windows y de la utilización de la API de Windows al implantar solamente algunos de los comandos más representativos del Explorador de Windows, once en total.

En las figuras 4.8 a) y 4.8 b) se muestran los menús de los comandos para Windows y para Word, respectivamente, que están implantados en este sistema. Cada comando es

descriptivo por si mismo, sin embargo, como Comandos de voz utiliza un reconocedor de palabras aisladas fue necesario acotar los comandos de voz a una sola palabra. En las tablas 4.1 y 4.2 se pueden ver que cada uno de los comandos de Windows y Word se encuentra asociado con su correspondiente comando de voz y con su descripción. Si observa con atención la tabla 4.1, se puede ver que existe un comando adicional de Windows, *Sistema*, que habilita los comandos del sistema operativo y si observamos también la tabla 4.2 la descripción de *Abre Word* nos dice que además de abrir la aplicación de Word también habilita los comandos de Word. Recordemos que Comandos de voz permite alternar entre comandos de Windows y comandos de Word para ser entrenados y reconocidos. Pues es gracias a estos dos comandos como se hace posible tal acción. Pensando en que no tiene sentido pronunciar los comandos de Word si éste no está abierto nos evita el procesamiento innecesario aumentando la precisión del reconocimiento.

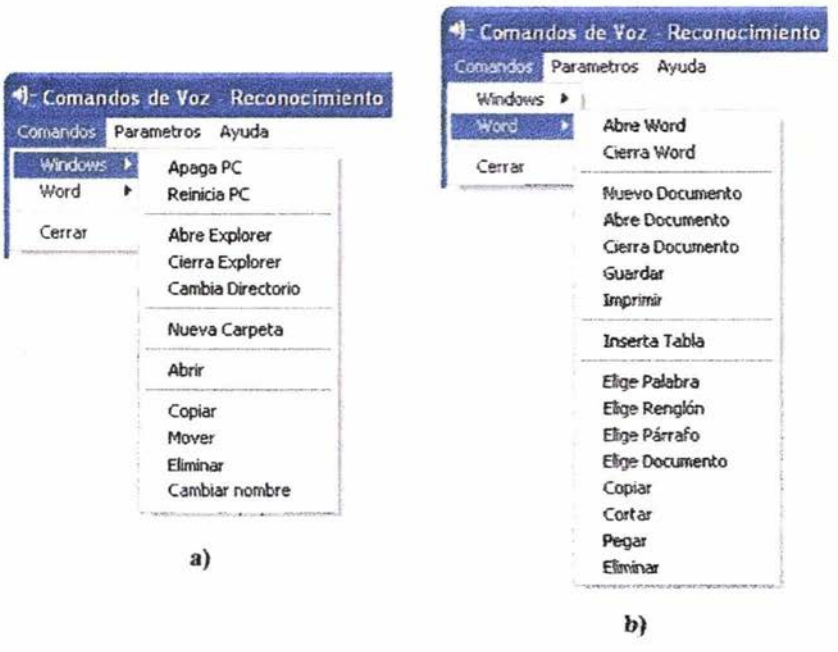


Fig. 4.8 a) Menú de comandos de Windows y b) Menú de comandos de Word.

Desde el punto de vista funcional he intercalado *Sistema* y *Abre Word* en los comandos de Word y de Windows, respectivamente, es decir, el comando de *Sistema* forma parte de los

comandos que son entrenados con Word y el comando de *AbreWord* forma parte de los comandos que son entrenados con Windows para dotar al sistema con el comportamiento descrito.

Comando	*Comando de voz*	*Descripción*
Apaga PC	"Apaga"	Apaga el equipo.
Reinicia PC	"Reinicia"	Reinicia el equipo.
Abre Explorer	"Explora"	Abre el Explorador de Windows.
Cierra Explorer	"Cierra"	Cierra la ventana del Explorador de Windows.
Nueva Carpeta	"Nueva"	Crea una nueva carpeta dentro de la carpeta actual.
Cambia Directorio	"Cambia"	Permite elegir el directorio que se desea explorar.
Abrir	"Abre"	Abre el elemento seleccionado. Si el elemento es un archivo que no es ejecutable, el Explorador ejecuta la aplicación asociada y despliega el documento. Si el elemento es una carpeta. Abre una nueva ventana del Explorador y muestra el contenido de la carpeta.
Copiar	"Copia"	Copia los elementos seleccionados a la ubicación deseada.
Mover	"Mueve"	Mueve los elementos a la ubicación deseada.
Eliminar	"Elimina"	Envía los elementos seleccionados a la papelera de reciclaje
Cambiar nombre	"Renombra"	Cambia el nombre de los elementos seleccionados.
Sistema	"Sistema"	Se habilitan los comandos del sistema operativo Windows.

Tabla 4.1 Comandos de voz asociados con los elementos del menú de comandos de Windows.

Por ejemplo, se puede estar trabajando solamente con el sistema operativo y si se desea abrir Word se pronuncia su respectivo comando, entonces se cambia automáticamente a los comandos de Word, con los que se puede continuar trabajando. Si en cualquier momento se desea regresar a los comandos de Windows se pronuncia el comando *Sistema* y automáticamente se realiza el cambio de nuevo. Cabe mencionar que, implícitamente Comandos de voz se conecta con Word aún cuando ya esté abierto lo que permite trabajar con la aplicación en el momento que se desee. Además, cuando se cierra Word

automáticamente se regresa a los comandos de Windows. Así, es posible alternar entre una y otra opción hasta que se termine de ejecutar Comandos de voz o se apague el equipo y sin que haya confusión entre unos comandos y otros.

Comando	*Comando de voz*	*Descripción*
Abre Word	"Word"	Abre la aplicación Word y habilita los comandos de Word.
Cierra Word	"Salir"	Cierra la aplicación Word
Nuevo Documento	"Nuevo"	Crea un nuevo documento en blanco.
Abre Documento	"Abre"	Despliega el cuadro de diálogo que abre un documento de Word.
Cierra Documento	"Cierra"	Cierra el documento activo.
Guardar	"Guarda"	Guarda el documento activo.
Imprimir	"Imprime"	Despliega el cuadro de diálogo de impresión.
Inserta Tabla	"Tabla"	Despliega el cuadro de diálogo de insertar tabla.
Elige Palabra	"Palabra"	Selecciona una palabra a la derecha.
Elige Renglón	"Renglón"	Selecciona el renglón actual.
Elige Párrafo	"Párrafo"	Selecciona el párrafo actual.
Elige Documento	"Documento"	Selecciona todo el documento activo.
Copiar	"Copia"	Copia la selección actual.
Cortar	"Corta"	Corta la selección actual.
Pegar	"Pega"	Pega la selección actual.
Eliminar	"Elimina"	Borra la selección actual.

Tabla 4.2 Comandos de voz asociados a los elementos del menú de comandos de Word.

Por otra parte, los comandos de Word se encuentran divididos en tres grupos. Los primeros dos, la apertura y el cierre de Word, se encargan de manejar la aplicación; los siguientes cinco comandos se encargan de mostrar el manejo de la funcionalidad de Word y los nueve restantes se encargan de la edición del contenido de un documento de Word. Por su parte los comandos de Windows se encuentran divididos en comandos del sistema operativo, apagar y reiniciar el equipo, comandos para manipular el Explorador de Windows, Abre y Cierra el Explorador y cambia de directorio que se explora. Además de los comandos de edición de archivos y carpetas del Explorador se encuentra también el de creación de carpetas y el de ejecución de aplicaciones.

Capítulo 5
Pruebas y resultados

Se probó el desempeño del sistema en dos entornos distintos de Windows, por lo que el entrenamiento, el reconocimiento y la ejecución de los comandos se realizaron en 2 PC's, la primera con procesador AMD-K6 3D (450 MHz), 56 MB en RAM, con sistema operativo Windows 98 y con Word 2000 instalado, y la segunda, con procesador Pentium 4 a 2.40 GHz., 256 MB en RAM, con sistema operativo Windows XP y con Word 2002 instalado.

Se eligieron estos dos sistemas por dos razones principales: Windows XP combina la capacidad de Windows 98 y Me para manejar el hardware con el núcleo más estable y fácil de manejar de su antecesor, Windows 2000. Y debido a que en el laboratorio de procesamiento de voz se cuentan solamente con equipos que poseen estos sistemas operativos.

Se formaron dos grupos de comandos de voz, los de Windows y los de Word. Se buscó evaluar, en primera instancia, la velocidad de entrenamiento y de reconocimiento del sistema con patrones de 16 centroides por segmento de voz y con patrones de 32 centroides por segmento de voz para ambos grupos de comandos. En el caso del reconocimiento se buscó obtener, también, el porcentaje de comandos reconocidos correctamente con patrones de 16 centroides por segmento de voz y con patrones de 32 centroides por segmento de voz para los dos grupos de comandos en las dos máquinas así como el porcentaje de todos los comandos reconocidos correctamente con los dos tipos de patrones en ambas máquinas; para ello se estableció pronunciar arbitrariamente 50 veces cada comando.

Además, se evaluó la compatibilidad del sistema en diferentes ambientes Windows para el primer grupo y la compatibilidad entre versiones de Word para el segundo grupo de comandos.

Para el equipo con procesador **AMD-K6 3D, 56 MB en RAM y Windows 98** se obtuvo lo siguiente:

ENTRENAMIENTO

q	t_{qWin} [mm:ss]	t_{rWin} [mm:ss]	$t_{totalWin}$ [mm:ss]	t_{qWord} [mm:ss]	t_{rWord} [mm:ss]	$t_{totalWord}$ [mm:ss]
16	01:45	01:28	03:13	02:13	02:10	04:23
32	01:49	01:38	03:27	02:23	02:28	04:51

Tabla 5.1 Tiempos de cuantización y de reconocimiento de prueba para 16 y 32 centroides con el equipo con procesador AMD-K6, 56 MB en RAM y Windows 98.

donde q corresponde al número de centroides o cuantizadores por segmento; t_{qWin} y t_{qWord} son los tiempos que se tarda en obtener los patrones de comparación de los comandos de Windows y Word, respectivamente y t_{rWin} y t_{rWord} son los tiempos de reconocimiento de prueba que se le hace a los comandos de Windows y Word, respectivamente, y $t_{totaWin}$ y $t_{totalWord}$ son los tiempos que duran ambos procesos, respectivamente.

Se observa que el menor tiempo de entrenamiento se obtiene cuando se utilizan patrones de 16 centroides por segmento, aunque difiere muy poco con respecto al tiempo que tarda cuando se utilizan patrones de 32 centroides por segmento.

RECONOCIMIENTO

Estos tiempos se encuentran medidos en cuanto se terminó de grabar el comando pronunciado:

q	t_{min} [ms]	t_{max} [ms]
16	269.23	312.50
32	307.69	355.77

Tabla 5.2 Tiempos de reconocimiento por comando para 16 y 32 centroides con el equipo con procesador AMD-K6, 56 MB en RAM y Windows 98.

donde q corresponde al número de centroides o cuantizadores por segmento, t_{min} y t_{max} son las aproximaciones de los tiempos de reconocimiento mínimo y máximo por comando, respectivamente. El tiempo de reconocimiento se midió en cuanto se terminó de grabar el comando pronunciado.

Se observa que el menor tiempo de reconocimiento se da cuando se utilizan patrones de 16 centroides por segmento.

En las tablas 5.3 y 5.4 se presenta la relación de comandos pronunciados vs. comandos reconocidos utilizando patrones de 16 y 32 centroides por segmento, respectivamente, y segmentación lineal de 4 segmentos. Las partes a) de cada tabla corresponden a los

comandos de Windows y las partes b) corresponden a los comandos de Word. Al observar dichas tablas tenemos que el porcentaje de cada comando reconocido correctamente se obtiene de dividir el número de veces que se reconoció correctamente cada comando entre el total de veces que se pronunció ese comando, es decir,

%cada comando reconocido correctamente = número de veces del comando reconocido correctamente / total de veces que se pronunció el comando

Los porcentajes totales obtenidos en cada tabla corresponden al porcentaje de los comandos de voz reconocidos correctamente tanto de Windows como de Word por separado, es decir, el total de sus respectivos comandos reconocidos correctamente entre el total de sus respectivos comandos pronunciados. De la misma manera el porcentaje total de comandos reconocidos correctamente por el sistema corresponde al total de comandos reconocidos correctamente entre el total de comandos pronunciados.

Usando **patrones de 16 centroides por segmento** tenemos:

%comandos de Windows reconocidos correctamente = Σ(comandos de Windows reconocidos correctamente) / total de comandos de Windows pronunciados = **591/(12*50)*100 = 98.50 %**

%comandos de Word reconocidos correctamente = Σ(comandos de Word reconocidos correctamente) / total de comandos de Word pronunciados = **780/(16*50)*100 = 97.50 %**

Para la obtención del porcentaje de todos los comandos del sistema reconocidos correctamente:

%*total* comandos del sistema reconocidos correctamente(q = 16) = Σ(todos los comandos reconocidos correctamente)/(total de comandos pronunciados) =

(comandos de Windows reconocidos correctamente + comandos de Word reconocidos correctamente)/(total de comandos pronunciados) =

(591 + 780)/(28*50)*100 = 1371/1400*100 = 97.93 %

y usando **patrones de 32 centroides por segmento** tenemos:

% comandos de Windows reconocidos correctamente = Σ(comandos de Windows reconocidos correctamente) / total de comandos de Windows pronunciados = **585/(12*50)*100 = 97.50 %**

% comandos de Word reconocidos correctamente = Σ(comandos de Word reconocidos correctamente) / total de comandos de Word pronunciados = **768/(16*50)*100 = 96.00 %**

Para la obtención del porcentaje de todos los comandos del sistema reconocidos correctamente:

%total* comandos del sistema reconocidos correctamente*(q = 32) = Σ(todos los comandos reconocidos correctamente)/(total de comandos pronunciados) =

(comandos de Windows reconocidos correctamente + comandos de Word reconocidos correctamente)/(total de comandos pronunciados) = **(585 + 768)/(28*50)*100 = 1353/1400*100 = 96.64 %**

Se observa que el mejor desempeño del sistema se obtendrá cuando se utilicen patrones de 16 centroides por segmento.

COMANDOS RECONOCIDOS / COMANDOS PRONUNCIADOS	Apaga PC	Reinicia PC	Abre Explorer	Cierra Explorer	Cambia Directorio	Nueva Carpeta	Abrir	Copia Archivo	Mueve Archivo	Borra Archivo	Cambiar nombre	Abre Word	% comandos reconocidos correctamente
Apaga PC	50												*100.00%*
Reinicia PC		49								1			*98.00%*
Abre Explorer	3		47										*94.00%*
Cierra Explorer				50									*100.00%*
Cambia Directorio	1				49								*98.00%*
Nueva Carpeta						50							*100.00%*
Abrir							50						*100.00%*
Copia Archivo								49		1			*98.00%*
Mueve Archivo									50				*100.00%*
Borra Archivo										50			*100.00%*
Cambiar nombre			3								47		*94.00%*
Abre Word												50	*100.00%*
											% total		*98.50%*

Tabla 5.3 a) Resultados del reconocimiento de los comandos de Windows usando segmentación lineal con 4 segmentos y 16 centroides en el equipo con procesador AMD-K6, 56 MB en RAM y Windows 98.

COMANDOS RECONOCIDOS / COMANDOS PRONUNCIADOS	Cierra Word	Abre Documento	Cierra Documento	Nuevo Documento	Guardar	Imprimir	Insertar Tabla	Elige Palabra	Elige Renglón	Elige Párrafo	Elige Documento	Copiar	Cortar	Pegar	Eliminar	Sistema	% comandos reconocidos correctamente
Cierra Word	50																*100.00%*
Abre Documento		50															*100.00%*
Cierra Documento			47													3	*94.00%*
Nuevo Documento				48						1	1						*96.00%*
Guardar					50												*100.00%*
Imprimir						49									1		*98.00%*
Insertar Tabla							50										*100.00%*
Elige Palabra	1						1	48									*96.00%*
Elige Renglón				1					48		1						*96.00%*
Elige Párrafo										50							*100.00%*
Elige Documento											49					1	*98.00%*
Copiar												48		1			*96.00%*
Cortar					3								47				*94.00%*
Pegar														50			*100.00%*
Eliminar															49	1	*98.00%*
Sistema			2					1								47	*94.00%*
															% total		*97.50%*

Tabla 5.3b) Resultados del reconocimiento de los comandos de Word usando segmentación lineal con 4 segmentos y 16 centroides en el equipo con procesador AMD-K6, 56 MB en RAM y Windows 98.

COMANDOS RECONOCIDOS / COMANDOS PRONUNCIADOS	Apaga PC	Reinicia PC	Abre Explorer	Cierra Explorer	Cambia Directorio	Nueva Carpeta	Abrir	Copia Archivo	Mueve Archivo	Borra Archivo	Cambiar nombre	Abre Word	% comandos reconocidos correctamente
Apaga PC	49						1						*98.00%*
Reinicia PC		50											*100.00%*
Abre Explorer	1		49										*98.00%*
Cierra Explorer				50									*100.00%*
Cambia Directorio		2			47			1					*94.00%*
Nueva Carpeta						50							*100.00%*
Abrir							50						*100.00%*
Copia Archivo								50					*100.00%*
Mueve Archivo				4					44	2			*88.00%*
Borra Archivo										50			*100.00%*
Cambiar nombre			4								46		*92.00%*
Abre Word												50	*100.00%*
												% total	*97.50%*

Tabla 5.4 a) Resultados del reconocimiento de los comandos de Windows usando segmentación lineal con 4 segmentos y 32 centroides en el equipo con procesador AMD-K6, 56 MB en RAM y Windows 98.

COMANDOS PRONUNCIADOS \ COMANDOS RECONOCIDOS	Cierra Word	Abre Documento	Cierra Documento	Nuevo Documento	Guardar	Imprimir	Insertar Tabla	Elige Palabra	Elige Renglón	Elige Párrafo	Elige Documento	Copiar	Cortar	Pegar	Eliminar	Sistema	% comandos reconocidos correctamente
Cierra Word	49															1	*98.00%*
Abre Documento		50															*100.00%*
Cierra Documento			48													2	*96.00%*
Nuevo Documento				49							1						*98.00%*
Guardar					50												*100.00%*
Imprimir						49						1					*98.00%*
Insertar Tabla							50										*100.00%*
Elige Palabra							1	48					1				*96.00%*
Elige Renglón									49						1		*98.00%*
Elige Párrafo								1		49							*98.00%*
Elige Documento											48	1			1		*96.00%*
Copiar			4									41		4		1	*82.00%*
Cortar					1			1					48				*96.00%*
Pegar														50			*100.00%*
Eliminar															50		*100.00%*
Sistema				7								2	1			40	*80.00%*
															% total		*96.00%*

Tabla 5.4 b) Resultados del reconocimiento de los comandos de Word usando segmentación lineal con 4 segmentos y 32 centroides en el equipo con procesador AMD-K6, 56 MB en RAM y Windows 98.

Para el equipo con procesador **Pentium 4 a 2.40 GHz., 256 MB en RAM y con Windows XP** se obtuvo lo siguiente:

ENTRENAMIENTO

q	t_{qWin} [mm:ss]	t_{rWin} [mm :ss]	$t_{totalWin}$ [mm:ss]	t_{qWord} [mm:ss]	t_{rWord} [mm:ss]	$t_{totalWord}$ [mm:ss]
16	00:34	00:21	00:55	00:43	00:32	01:15
32	00:23	00:23	00:56	00:40	00:36	01:16

Tabla 5.5 Tiempos de cuantización y de reconocimiento de prueba para 16 y 32 centroides con el procesador Pentium 4 a 2.40 GHz, 256 MB en RAM y Windows XP.

donde q corresponde al número de centroides o cuantizadores por segmento; t_{qWin} y t_{qWord} son los tiempos que se tarda en obtener los patrones de comparación de los comandos de Windows y Word, respectivamente y t_{rWin} y t_{rWord} son los tiempos de reconocimiento de prueba que se le hace a los comandos de Windows y Word, respectivamente, y $t_{totaWin}$ y $t_{totalWord}$ son los tiempos que duran ambos procesos, respectivamente.

Se observa que hay una disminución en cuanto a tiempo de cuantización en ambos procesos cuando se utilizan patrones de 32 centroides por segmento, sin embargo el tiempo de reconocimiento es mayor cuando se usan este tipo de patrones. Y se observa también que las diferencias en los tiempos de entrenamiento son muy pequeñas.

RECONOCIMIENTO

Estos tiempos se encuentran medidos en cuanto se terminó de grabar el comando pronunciado:

Q	t_{min} [ms]	t_{max} [ms]
16	67.31	76.92
32	73.72	115.38

Tabla 5.6 Tiempos de reconocimiento por comando para 16 y 32 centroides para el equipo con procesador Pentium 4 a 2.40 GHz, 256 MB en RAM y Windows XP.

donde q corresponde al número de centroides o cuantizadores por segmento, t_{min} y t_{max} son las aproximaciones de los tiempos de reconocimiento mínimo y máximo por comando, respectivamente.

Cabe mencionar que el sistema inicialmente poseía un hilo de trabajo, el que al ocupar espacio en memoria, llegó a modificar tanto los tiempos de entrenamiento como de

reconocimiento, pues en el caso de Windows 98 el entrenamiento de todos los comandos llegó a tardar hasta 3 horas y media y en el caso de Windows XP hasta 45 minutos. Por su parte, en cuanto al reconocimiento, en Windows 98 éste llegó a durar hasta 27 segundos y en Windows XP hasta 3 segundos, resultados que no pueden considerarse óptimos para un sistema de reconocimiento de voz. Se puede ver que este hecho repercutió significativamente en la velocidad del sistema, sin embargo, al modificar el código y quitar el hilo reasignando los comandos a eventos se obtuvieron los resultados anteriores obteniendo, así, una mejora significativa en la velocidad del sistema.

Vemos que se reducen considerablemente los tiempos de entrenamiento y de reconocimiento en comparación con el tipo de procesador y de sistema operativo y también al haber una mejora del código fuente del sistema se liberaron recursos lo que posibilitó el aumento de velocidad.

En las tablas 5.7 y 5.8 se presenta la relación de comandos pronunciados vs. comandos reconocidos para patrones con 16 y 32 centroides por segmento, respectivamente, y utilizando segmentación lineal de 4 segmentos. Las partes a) de cada tabla corresponden a los comandos de Windows y las partes b) corresponden a los comandos de Word.

Usando **patrones de 16 centroides por segmento** tenemos:

%comandos de Windows reconocidos correctamente =

Σ(comandos de Windows reconocidos correctamente) / total de comandos de Windows pronunciados = **582/(12*50)*100 = 97.00 %**

%comandos de Word reconocidos correctamente = Σ(comandos de Word reconocidos correctamente) / total de comandos de Word pronunciados = **770/(16*50)*100 = 96.25 %**

Para la obtención del porcentaje de todos los comandos del sistema reconocidos correctamente:

***%total* comandos del sistema reconocidos correctamente(q = 16)** =

Σ(todos los comandos reconocidos correctamente)/(total de comandos pronunciados) =

(comandos de Windows reconocidos correctamente + comandos de Word reconocidos correctamente)/(total de comandos pronunciados) =

(582 + 770)/(28*50) = 1352/1400 = 96.57 %

y usando **32 centroides** tenemos:

%comandos de Windows reconocidos correctamente = Σ(comandos de Windows reconocidos correctamente) / total de comandos de Windows pronunciados = **563/(12*50)*100 = 93.83 %**

%comandos de Word reconocidos correctamente = Σ(comandos de Word reconocidos correctamente) / total de comandos de Word pronunciados = **764/(16*50)*100 = 95.50 %**

Para la obtención del porcentaje de todos los comandos del sistema reconocidos correctamente:

***%total* comandos del sistema reconocidos correctamente(*q = 32*)** = Σ(todos los comandos reconocidos correctamente)/(total de comandos pronunciados) =

(comandos de Windows reconocidos correctamente + comandos de Word reconocidos correctamente)/(total de comandos pronunciados) =

(563 + 764)/(28*50) = 1327/1400 = 94.79 %

Se observa también en este equipo que el mejor porcentaje de reconocimiento lo obtendremos cuando se utilicen patrones de 16 centroides por segmento.

COMANDOS RECONOCIDOS / COMANDOS PRONUNCIADOS	Apaga PC	Reinicia PC	Abre Explorer	Cierra Explorer	Cambia Directorio	Nueva Carpeta	Abrir	Copia Archivo	Mueve Archivo	Borra Archivo	Cambiar nombre	Abre Word	% comandos reconocidos correctamente
Apaga PC	50												*100.00%*
Reinicia PC		50											*100.00%*
Abre Explorer	1		48			1							*96.00%*
Cierra Explorer				50									*100.00%*
Cambia Directorio					48				2				*96.00%*
Nueva Carpeta	1					49							*98.00%*
Abrir							46		4				*92.00%*
Copia Archivo							1	47	2				*94.00%*
Mueve Archivo							1		49				*98.00%*
Borra Archivo										50			*100.00%*
Cambiar nombre	1										45		*90.00%*
Abre Word												50	*100.00%*
												% total	*97.00%*

Tabla 5.7 a) Resultados del reconocimiento de los comandos de Windows usando segmentación lineal con 4 segmentos y 16 centroides en el equipo Pentium 4 a 2.40 GHz, 256 MB en RAM y Windows XP.

COMANDOS RECONOCIDOS / COMANDOS PRONUNCIADOS	Cierra Word	Abre Documento	Cierra Documento	Nuevo Documento	Guardar	Imprimir	Insertar Tabla	Elige Palabra	Elige Renglón	Elige Párrafo	Elige Documento	Copiar	Cortar	Pegar	Eliminar	Sistema	% comandos reconocidos correctamente
Cierra Word	49	1															*98.00%*
Abre Documento		47						1		2							*94.00%*
Cierra Documento			44													6	*88.00%*
Nuevo Documento				50													*100.00%*
Guardar					48			1					1				*96.00%*
Imprimir						48									2		*96.00%*
Insertar Tabla							48	2									*96.00%*
Elige Palabra							3	47									*94.00%*
Elige Renglón									50								*100.00%*
Elige Párrafo										50							*100.00%*
Elige Documento					1						49						*98.00%*
Copiar												50					*100.00%*
Cortar					4								46				*92.00%*
Pegar				1										48	1		*96.00%*
Eliminar															50		*100.00%*
Sistema				1											3	46	*92.00%*
															% total		*96.25%*

Tabla 5.7 b) Resultados del reconocimiento de los comandos de Word usando segmentación lineal con 4 segmentos y 16 centroides en el equipo con procesador Pentium 4 a 2.40 GHz, 256 MB en RAM y Windows XP.

COMANDOS RECONOCIDOS / COMANDOS PRONUNCIADOS	Apaga PC	Reinicia PC	Abre Explorer	Cierra Explorer	Cambia Directorio	Nueva Carpeta	Abrir	Copia Archivo	Mueve Archivo	Borra Archivo	Cambiar nombre	Abre Word	% comandos reconocidos correctamente
Apaga PC	47												*94.00%*
Reinicia PC		50											*100.00%*
Abre Explorer	2	2	44				2						*88.00%*
Cierra Explorer				46			3		1				*92.00%*
Cambia Directorio					49	1							*98.00%*
Nueva Carpeta			1			47	1		1				*94.00%*
Abrir							50						*100.00%*
Copia Archivo							5	45					*90.00%*
Mueve Archivo							4		46				*92.00%*
Borra Archivo		6					1			43			*86.00%*
Cambiar nombre	1	3									46		*92.00%*
Abre Word												50	*100.00%*
												% total	*93.83%*

Tabla 5.8 a) Resultados del reconocimiento de los comandos de Windows usando segmentación lineal con 4 segmentos y 32 centroides en el equipo con procesador Pentium 4 a 2.40 GHz, 256 MB en RAM y Windows XP.

COMANDOS RECONOCIDOS / COMANDOS PRONUNCIADOS	Cierra Word	Abre Documento	Cierra Documento	Nuevo Documento	Guardar	Imprimir	Insertar Tabla	Elige Palabra	Elige Renglón	Elige Párrafo	Elige Documento	Copiar	Cortar	Pegar	Eliminar	Sistema	% comandos reconocidos correctamente
Cierra Word	46								1		3						*92.00%*
Abre Documento		49					1										*98.00%*
Cierra Documento			43								1					6	*86.00%*
Nuevo Documento				49			1										*98.00%*
Guardar					47		2	1									*94.00%*
Imprimir						47									3		*94.00%*
Insertar Tabla							50										*100.00%*
Elige Palabra							1	49									*98.00%*
Elige Renglón									50								*100.00%*
Elige Párrafo										50							*100.00%*
Elige Documento				2			1			1	46						*92.00%*
Copiar												50					*100.00%*
Cortar					6		2						42				*84.00%*
Pegar														48			*96.00%*
Eliminar															50		*100.00%*
Sistema											1				1	48	*96.00%*
																% total	*95.50%*

Tabla 5.8 b) Resultados del reconocimiento de los comandos de Word usando segmentación lineal con 4 segmentos y 32 centroides en el equipo con procesador Pentium 4 a 2.40 GHz, 256 MB en RAM y Windows XP.

En lo que respecta a la ejecución de comandos se tienen que el sistema funciona correctamente en ambos entornos de Windows, es decir, no existen problemas de compatibilidad entre versiones de Windows. Esto se debe, por una parte, a que los algoritmos desarrollados para el sistema fueron creados con Visual C++ y con la MFC ya que programar en Windows involucra la utilización de Visual C++ y actualmente de Visual C#, sin embargo, este último se encuentra fuera del alcance de este trabajo. Entonces, por medio de la MFC es posible interactuar con diferentes ambientes Windows y obtener así la compatibilidad deseada.

Y por otra, gracias a que las interfaces COM, que permiten acceder al shell en forma confiable y, como se sabe, son una parte fundamental de la estructura de los sistemas Windows y por ende pueden ser aplicadas a cualquiera de ellos, con ciertas restricciones,

debido a la seguridad que se da en los sistemas basados en Windows NT y que resulta ser importante a la hora de querer apagar o reiniciar el equipo, por ejemplo. Además ambos sistemas operativos utilizan las mismas interfaces para acceder al Explorador de Windows, sin embargo, Windows XP al poseer una versión mejorada del componente ***shdocvw.dll*** cuenta, también, con otras interfaces nuevas y mejoradas que se heredan de las primeras, por lo que, cuando se trabaja en Windows XP o en Windows 2000, se obtiene en realidad el acceso a las interfaces primitivas para poder automatizar al sistema operativo.

En lo que respecta a Word, no se presentaron problemas de ejecución con ambas versiones de Word. Además, al incluir la librería de tipos de Word 2000 es posible automatizar esta versión y también versiones posteriores, como Word XP 2002 y 2003, ya que únicamente varían algunos parámetros opcionales, que se pueden agregar cuando sea necesario, por lo que hacemos uso indistintamente de estas versiones. Por supuesto, nuevas versiones de Word ofrecen también nuevos objetos con sus respectivos métodos y propiedades, sin embargo, los comandos básicos se mantienen, por lo que se mantiene también la compatibilidad. Ahora bien, el tiempo de ejecución de los comandos de voz es ligeramente mayor al tiempo en que el sistema operativo realiza las mismas operaciones. Situación que es inherente al mecanismo de automatización ya que las llamadas a los métodos requieren que se hagan de manera indirecta a diferencia de Windows, que las realiza de manera directa, sin embargo, el usuario final no percibiría tal diferencia.

Como resultados adicionales se obtuvieron:

En cuanto a espacio en disco:

Para **patrones de 16 centroides por segmento**:

```
28 comandos * 4 archivos de centroides/comando * 1.056 KB/archivo
de centroides = 118.27 KB en disco
```

Para **patrones de 32 centroides por segmento**:

```
28 comandos * 4 archivos de centroides/comando * 2.112 KB/archivo
de centroides = 236.54 KB en disco
```

En comparación con el espacio que se tiene de las bases de los archivos de audio:

```
28 comandos * 26 archivos/comando * 114.688 KB/archivo = 80510.98
KB = 80.51 MB en disco
```

El ejecutable del sistema ocupa **143.36 KB en disco duro** y aproximadamente **4.36 MB en RAM** al inicio. Sin embargo, en la estimación del espacio en memoria RAM

no se incluye la memoria que ocupan tanto el Explorador de Windows como Word por sí mismos ya que ésta depende del diseño que les haya dado Microsoft.

CONCLUSIONES

1. El reconocedor de palabras aisladas que utiliza el sistema ***Comandos de voz*** genera resultados que se encuentran por arriba del 90 % tanto en ambos entornos de Windows como para ambas versiones de Word, resultados que para sistemas de reconocimiento de voz se consideran bastante óptimos y hacen que sea aceptable trabajar con él.
2. En ambos equipos el mejor porcentaje de reconocimiento del sistema se obtiene cuando se utilizan patrones de 16 centroides por segmento y aunque el mejor desempeño se haya obtenido en el equipo con procesador AMD-K6 la diferencia del 1.36% con respecto al equipo con procesador Pentium 4 no fue considerable. De aquí se infiere que aumentar la cantidad de centroides por segmento no hace que el reconocimiento de los comandos sea mucho mejor y, al contrario, aumenta el tiempo de ejecución.
3. Aún así, la velocidad con que trabaja en ambos sistemas operativos, en mi opinión, lo hacen un reconocedor bastante veloz y aunque el menor tiempo de reconocimiento se haya obtenido en el equipo con procesador Pentium 4 con patrones de 16 centroides por segmento, el tiempo que se tarda con patrones de 32 centroides por segmento no es considerablemente mayor.
4. El sistema ***Comandos de voz*** es compatible con los dos entornos de Windows propuestos gracias a la facilidad que otorga la MFC para interactuar con distintas versiones de Windows y al uso de las interfaces COM para acceder y controlar al shell de Windows lo que permite que el usuario pueda manejarlo sin complicaciones.
5. De igual manera, este sistema es compatible con distintas versiones de Word al utilizar el mecanismo de automatización que facilita el manejo de este procesador de textos por los usuarios.
6. Se tiene el inconveniente que se debe trabajar únicamente con palabras que pertenezcan al diccionario de palabras ya que el sistema no realiza seguimiento de fonemas y por tanto no discrimina otras palabras.
7. Otro inconveniente que es importante mencionar es: que el sistema no se ejecuta en tiempo real. Este hecho hace que no sea tan fácil de manejar por personas discapacitadas.
8. Es importante destacar que, así como mediante COM es posible acceder a innumerables servicios del sistema operativo que no podrían ser usados de otra manera, se debe

proceder con cautela cuando se trabaje con las interfaces COM y los elementos que componen a Windows ya que se corre el riesgo de dañarlo drásticamente.

9. En mi experiencia personal, a lo largo de mi investigación tuve problemas para encontrar información acerca de cómo programar el shell de Windows, empecé leyendo la documentación de Microsoft, sin embargo, ésta no fue suficiente pues a veces resultó confusa para poder entender en qué consiste el shell de Windows por lo que fue necesario realizar una búsqueda exhaustiva de este tema y revisar distintos textos.

REQUERIMIENTOS

La siguiente tabla muestra los requerimientos mínimos del sistema que se basan a su vez en los requerimientos que establece Microsoft para Windows y para Word:

Equipo/Procesador	• Con Word 2000: Equipo con un procesador Pentium a 75 MHz o superior. • Con Word 2002: Equipo con un procesador Pentium a 133 MHz o superior.
Memoria	• Con Windows 98: 16 MB de RAM para el sistema operativo, 4 MB adicionales para Word 2000 u 8 MB para Word 2002 y 4.5 MB para el sistema de comandos de voz. • Para Windows NT Workstation: 32 MB de RAM para el sistema operativo, 4 MB adicionales para Word 2000 u 8 MB si se trata de Word 2002 y 4.5 MB para el sistema de comandos de voz.
Sistema Operativo	Windows 98 o posterior, o Windows NT® Workstation 4.0 Service Pack 3 o posterior.
Internet Explorer	Versión 4.0 o posterior para poder utiilzar los comandos de Windows.
Microsoft Word	Versión 2000 o posterior para poder utilizar los comandos de Word.

MEJORAS

Se puede adaptar el sistema comandos de voz para ser utilizado con otro tipo de análisis de predicción lineal como el LPC y con otras funciones de distorsión como la distancia de Itakura-Sahito. Se puede mejorar intentando otros tipos de segmentación, como la segmentación KM o la segmentación acústica o MLR.

Otra posibilidad es el que se realice traslape en las ventanas. No se utilizó el traslape en las ventanas debido a que se pensó en mantener en todo momento la velocidad del sistema. El

empleo del traslape de ventanas disminuye la velocidad de respuesta ya que aumenta el número de cálculos y la cantidad de memoria utilizada.

Basar el reconocimiento en umbrales para poder descartar palabras que no estén en el vocabulario y mejorar la precisión del reconocimiento; lo que podría llevar a que se hiciera el reconocimiento en tiempo real.

Otra posible mejora sería portar el código que se tiene a Visual C# para que pueda trabajar con versiones posteriores de Windows XP. Para ambientes no Windows sería necesaria otro tipo de implantación con lenguaje C++ o con Java.

Se puede extender el sistema para que también se pueda navegar en Internet mediante comandos de voz, ya que al hacer uso del componente ***shdocvw.dll*** es posible automatizar también el Explorador de Internet.

APLICACIONES

Con las mejoras podría aplicarse a personas discapacitadas para que puedan manejar los entornos de Windows y de Word sin necesidad de utilizar las manos.

Me enfrenté al problema de que el sistema de reconocimiento de voz en el que me basé sólo funcionaba con un equipo en particular debido a que sólo estaba adaptado para una tarjeta de sonido en particular, SoundBlaster SB Pro, por lo que decidí adaptar el sistema para que el mismo sistema operativo fuera quién realizara la grabación por medio de sus propios controladores y que se pueda aplicar a otros equipos que cuenten con diferentes tarjetas de sonido. El sistema posee un módulo de grabación que puede utilizarse para grabar comandos o palabras que se utilicen en aplicaciones Windows.

Asimismo permite entrenar comandos que sean independientes de los comandos de Windows y Word y del sistema operativo de Windows para que puedan ser utilizados en otras aplicaciones de reconocimiento de voz.

Apéndice A

El entorno de usuario de Windows

A.1 El sistema operativo Windows

El *sistema operativo* es el intermediario entre la computadora y el usuario; es el software básico que permite al usuario comunicarse con la computadora y gestionar los recursos que ésta posee de una manera cómoda y eficiente. Recursos fundamentales como la memoria, las impresoras, las unidades de disco, el teclado o el mouse, son gestionados a un nivel básico desde el momento en que se enciende la computadora y de esta manera el usuario puede interactuar con ellos sin la necesidad de profundizar en el conocimiento de tales aspectos..

En este apéndice se ofrece una perspectiva general de algunos de los elementos básicos que se incluyen en el sistema operativo Windows XP Profesional y que permiten al usuario controlar el ambiente de cómputo, algunas veces conocido como shell. Los elementos nombrados en las siguientes secciones son considerados como las bases del entorno de usuario de Windows y también la fuente de la interacción del usuario con las aplicaciones de Windows. No obstante, no se pretende cubrir todos los aspectos que posee el entorno de Windows sino únicamente proporcionar al usuario una manera de entender el presente trabajo. Si se desea profundizar más en el tema es necesario consultar el manual del usuario de Windows XP o la ayuda que el sistema nos ofrece.

El sistema operativo Windows trabaja principalmente con *archivos* y *carpetas*. Un *archivo* es una colección de texto o datos guardados bajo un nombre único. La mayoría de la información almacenada en la computadora se encuentra en archivos. De entre los archivos que destacan se encuentran los que corresponden a las aplicaciones y a los documentos. Una *aplicación* es un software o programa que se utilizan para realizar un determinado tipo de trabajo, como el procesamiento de textos o el manejo de una base de datos y un *documento* es todo aquello que se cree con una aplicación, incluyendo cualquier información que se escriba, edite, presente en pantalla o guarde. Las *carpetas* son contenedores para las aplicaciones, los archivos y otras carpetas que se utilizan para organizar la información en la computadora. Las carpetas son consideradas como el equivalente a los directorios de algunos otros sistemas de archivos.

A.2 El escritorio

Cuando se inicie Windows quedará abierto en el escritorio, como se muestra en la figura A.1. El *escritorio* representa el área de trabajo principal para el usuario, llena la pantalla y forma el respaldo visual para todas las operaciones, y todas ellas se realizarán dentro de los límites del escritorio. El escritorio es el lugar adecuado para que el usuario coloque y pueda alcanzar fácilmente los objetos que se encuentran en el sistema de archivos. El escritorio también sirve como un espacio de trabajo privado para una computadora que se encuentra conectada en red, por medio de la cuál un usuario puede navegar y acceder a otros objetos que se encuentran en otros equipos.

Fig. A.1 El escritorio y la barra de tareas de Microsoft Windows.

A.2.1 La barra de tareas

La *barra de tareas* es un componente especial del escritorio que se utiliza para conmutar entre ventanas abiertas y acceder a los comandos globales y otros objetos que se usan

comúnmente en Windows. Asimismo la barra de tareas permite al usuario acceso a las aplicaciones y notifica de sucesos del sistema y de las aplicaciones aún cuando éstas ya no estén activas. Generalmente se encuentra ubicado en la parte inferior del escritorio, no obstante, el usuario puede cambiarla de lugar si lo desea. La figura A.2 muestra la barra de tareas y los elementos que la componen.

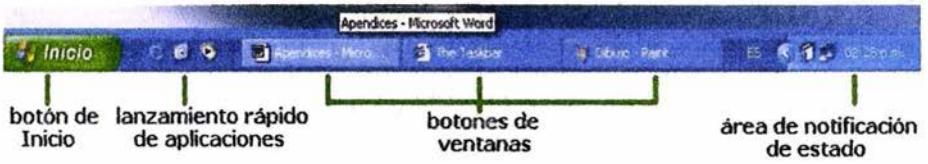


Fig. A 2 Elementos de la barra de tareas.

Como se puede observar, al lado izquierdo de la barra de tareas se encuentra el *botón de Inicio*, éste despliega un menú especial, mostrado en la figura A.3 y que incluye el nombre del usuario en la parte superior; en el área central izquierda se encuentran las aplicaciones que vienen incluidas con Windows y aquellas que se utilizan más frecuentemente, además del menú que accede al resto de las aplicaciones; en el área central derecha se encuentran entre otras cosas, los comandos para abrir o encontrar archivos y las partes de Windows a las que se accede más a menudo, como Mis documentos o Mis imágenes. Por último, en la parte inferior se encuentran las opciones de apagar el equipo o cambiar de usuario, con la primera es posible reiniciar Windows o cerrarlo y con la segunda es posible cerrar la sesión actual.

Junto al botón de inicio se encuentra la *barra de lanzamiento rápido de aplicaciones*. Esta área especial de la barra de tareas ha sido diseñada para permitir que los usuarios tengan un fácil acceso a Internet, al correo, al escritorio y a otros servicios básicos del sistema y se utiliza también para colocar aquellas aplicaciones de uso más frecuente.

En la parte media de la barra de tareas se colocan los *botones de ventanas*. Cada vez que un usuario abre una aplicación, se abre una ventana principal y su correspondiente botón se coloca automáticamente en la barra de tareas. Un botón de este tipo proporciona el acceso a la ventana de una aplicación abierta y es una manera muy conveniente para el usuario de alternar entre aplicaciones abiertas. La barra de tareas automáticamente ajusta el tamaño de los botones para acomodarlos como le sea posible y si no el espacio es cada vez más

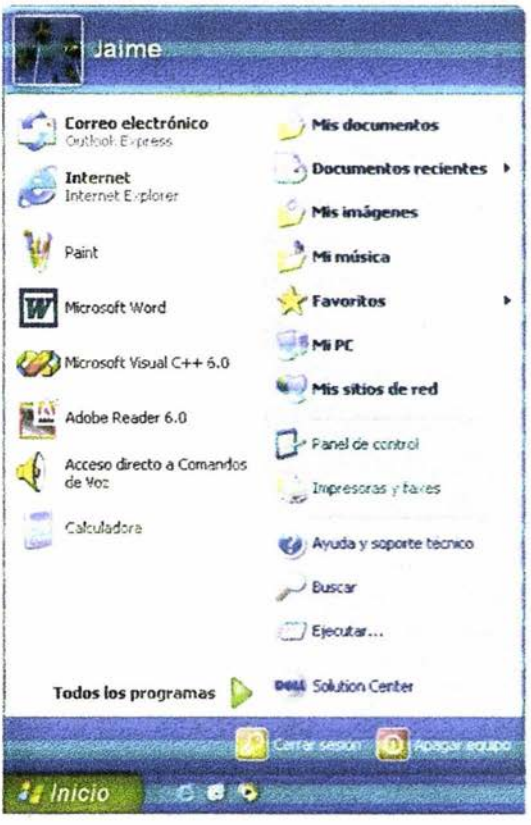


Fig. A.3 Elementos del menú de inicio.

reducido, es decir, que se abren más aplicaciones, lo que hace la barra de tareas es que también agrupa a todas los programas en uso del mismo tipo. Cuando el botón sea examinado, la barra de tareas automáticamente proporcionará una ventana, denominada Tooltip, que desplegará el título completo de la aplicación. Cuando se minimiza la ventana, su botón permanece en la barra de tareas y cuando la ventana se cierra su respectivo botón es removido de ella.

En el lado opuesto del botón de inicio se encuentra el *área de notificación de estado*. esta área contiene elementos que representan a aquellos programas que se cargan automáticamente al encender la computadora, los programas residentes, y proporciona, también, otro tipo de información adicional, como la indicación del idioma, en este caso

español o ES como lo representa Windows en la barra de tareas. El área de notificación por omisión incluye el reloj y el indicador de volumen.

A.2.2 Iconos

Los iconos son la representación pictórica de objetos, como documentos o aplicaciones, en el entorno de Windows y se les puede encontrar ya sea en el escritorio o en las ventanas. Windows utiliza los iconos debido a la facilidad que resulta de asociar una imagen con un determinado objeto para trabajar con él. En general, existen varios tipos de iconos que Windows maneja comúnmente y que pueden ser de utilidad familiarizarse con ellos antes de profundizarse más en este sistema operativo. Algunos de los más comunes se observan en la tabla A.1 a. y en la tabla A.1 b se muestran los iconos de los elementos que son parte de Windows y que el sistema operativo pone a disposición del usuario.

Icono	*Descripción*
 Icono de aplicación	Los iconos de aplicación representan a las aplicaciones en Windows, por ejemplo Microsoft Word.
 Icono de Documento	Los iconos de documento representan a los archivos asociados a las aplicaciones, como por ejemplo, un documento de Word.
 Acceso directo a Comandos de Voz	Los iconos de acceso directo proporcionan acceso rápido a otro elemento. Un icono de acceso directo utiliza el icono del tipo de archivo al que está ligado traslapado con el símbolo de liga.
 Carpeta	Las iconos de carpetas representan a las carpetas del sistema, que contienen a su vez a otras carpetas y archivos.

Tabla A.1 a) Algunos iconos que se utilizan comúnmente para trabajar en el entorno de usuario de Windows.

Icono	*Descripción*
 Mi PC	Proporciona el acceso e información a unidades de disco, cámaras, escáners y otro tipo de hardware conectado al equipo.
 Mis sitios de red	Permite el acceso e información sobre carpetas o archivos que se encuentran en otros equipos.
 Internet Explorer	Busca y muestra información de sitios Web en Internet.
 Papelera de reciclaje	Almacena los archivos y carpetas borrados.
 Panel de control	Proporciona el acceso a las propiedades de dispositivos instalados y recursos, por ejemplo, letras, monitores y teclados. Con éste es posible configurar y personalizar el aspecto de la computadora.

Tabla A.1 b) Iconos de elementos de uso común en Windows.

A.2.3 Ventanas

Se pueden abrir ventanas desde los iconos. La interfaz de las ventanas proporciona los medios para ver y editar la información, desplegar el contenido y las propiedades de los objetos. Asimismo pueden proporcionar información acerca de los parámetros y las entradas para completar comandos, paletas de opciones, configuración o herramientas, mensajes para informar sobre una situación en particular.

Durante el trabajo con aplicaciones en el escritorio aparecen dos tipos de ventana principales: ventanas de aplicación y ventanas de documento.

Cuando se inicia cualquier aplicación, ésta se desplegará en una ventana, denominada ventana de aplicación, que aparecerá en el escritorio. Las ventanas de aplicación contienen una aplicación en ejecución. En la parte superior de la ventana de aplicación aparece el nombre de la misma el documento asociado y la barra de menús que tiene la aplicación.

Las ventanas de documento son ventanas secundarias que aparecen dentro de una ventana de aplicación. Se llaman así porque contienen documentos o archivos de datos. Pueden

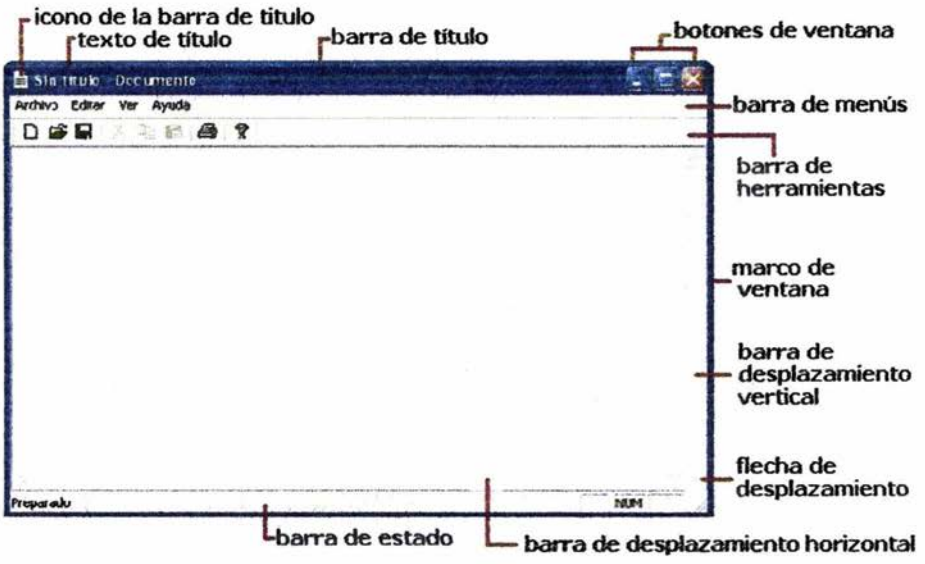


Fig A.4 Partes de una ventana.

abrirse varias ventanas de documento a la vez. Las ventanas de documento tienen la misma barra de menús que la ventana de aplicación correspondiente. Los comandos que afecten a la aplicación tendrán efecto sobre el documento.

En la figura A.4 se muestran las partes de una ventana que se describen a continuación:

- Las ventanas definen su extensión por medio de un marco o borde.
- Una ventana de aplicación incluye la pequeña versión del icono del objeto. Este icono aparece en la esquina superior izquierda de la barra de título y representa el objeto que está siendo visto en la ventana. Si la ventana contiene una herramienta o utilidad, esto es, una aplicación que no crea, no carga ni guarda sus propios archivos de datos, entonces el icono corresponde al icono de la aplicación. Si, al contrario, la aplicación crea, carga y guarda documentos o archivos de datos y la ventana representa la vista de uno de sus archivos, el icono corresponde al icono de documento. En el icono de la barra de título también se encuentra el cuadro de menú del sistema y resulta de mayor utilidad cuando se usa el teclado para mover, maximizar, minimizar, cerrar y modificar su tamaño.

- La barra de título muestra el nombre de la aplicación o documento. Si hay varias ventanas abiertas, la barra de título de la ventana activa, aquella en la que se esté trabajando, será de distinto color o intensidad de las demás barras de título. El texto del título puede ser el nombre de la aplicación y el nombre de un documento, o bien, el nombre de un directorio u otro archivo de datos.
- La barra de herramientas es un panel que contiene un conjunto de controles, diseñados para proporcionar acceso rápido a comandos u opciones específicas.
- En la barra de menús se muestran los menús disponibles. Un menú contiene la lista de comandos o acciones que pueden realizarse con Windows.
- Las barras de desplazamiento permiten mostrar en pantalla partes de un documento, cuando el documento completo no cabe en una sola ventana. El cuadro de desplazamiento indica el punto del documento donde se está trabajando y mediante las flechas de desplazamiento se pueden mover a otra parte del documento. Asimismo, una ventana puede tener una barra de desplazamiento vertical o una horizontal o ambas.
- Generalmente, cada ventana posee tres botones: el de minimizar  , el de maximizar  y el de cerrar  , con los que se puede reducir la ventana a un icono, ampliar la ventana de la aplicación activa de tal manera que ocupe todo el escritorio o cerrar la ventana, respectivamente. Cuando se maximiza la ventana, el botón de maximizar cambia por el botón de restaurar  , que permite regresar la ventana a su tamaño original.
- La barra de estado es un área que despliega información acerca del estado actual de lo que está siendo visualizado en la ventana o cualquier otra información contextual, como el estado del teclado.

Técnicas comunes de interacción. El *mouse* es el principal dispositivo de entrada que el usuario emplea para interactuar con el entorno de Windows. El mouse está ligado con un gráfico en la pantalla, denominado apuntador  . Por su parte, el teclado se utiliza principalmente para introducir y editar información textual. Sin embargo el entorno de

usuario también admite el uso del teclado para navegar, habilitar modos de operación, modificar entradas y realizar ciertas operaciones de manera abreviada.

La *selección* es el principal medio por el cuál el usuario identifica objetos en el entorno de usuario Una vez que el usuario seleccione un objeto podrá realizar acciones sobre dicho objeto.

La selección con el mouse se lleva a cabo cuando se coloca el apuntador sobre el objeto o la locación deseados y se da un clic[1], con el botón primario, que generalmente es el botón izquierdo. También es posible seleccionar un rango de objetos, primero damos un clic por encima de donde se encuentra el primer objeto y luego, manteniendo oprimido el botón, arrastramos haciendo que se resalten todos aquellos objetos que deseemos seleccionar, como se puede ver en la figura A.5. En cuanto se haya seleccionado el último objeto deseado dejamos de oprimir el botón izquierdo.

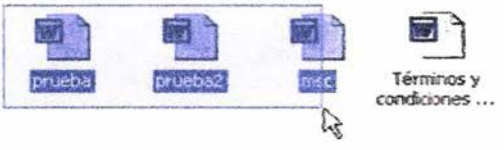


Fig A.5 Ejemplo de selección de múltiples elementos

Cuando se utiliza el teclado para seleccionar elementos, los objetos seleccionados se definen por el foco de entrada[2], que generalmente es el elemento que se encuentra seleccionado. Se puede utilizar el teclado para ampliar la función de selección del mouse, es decir, para seleccionar múltiples objetos que no son adyacentes se seleccionan dando clic sobre cada uno de ellos mientras se mantiene oprimida la tecla Ctrl.

La otra forma de identificar o acceder a un objeto es navegando hacia él. Podemos navegar por medio del mouse y por medio del teclado. El hecho de que los usuarios pueden navegar a través del entorno usando tanto el teclado como el mouse es un principio fundamental y una de las principales ventajas que ofrece Windows al usuario, aunque la navegación también puede darse con comandos específicos o elementos de la interface, por ejemplo, las operaciones de vista algunas veces pueden considerarse como una forma de navegar. El abrir documentos, dar clic en una liga o alternar entre ventanas son formas de moverse de un contexto a otro.

---

[1] A la acción de oprimir alguno de los botones del mouse se le denomina *clic*.

[2] El *foco de entrada* es la indicación de la locación hacia donde la entrada está siendo dirigida

Navegar con el mouse es sencillo; basta con que el usuario mueva el mouse hacia la dirección que desee y al mismo tiempo el apuntador se mueve en la dirección correspondiente en pantalla. Después selecciona el objeto y si es una carpeta puede dar doble clic para acceder a su contenido y seguir navegando hasta llegar al objeto deseado.

La navegación con el teclado requiere que el usuario presione teclas específicas y combinaciones de ellas para mover

Las teclas básicas que se utilizan para navegar son las teclas de Inicio y Fin, las cuatro teclas de dirección, ← → ↑ ↓ las de Avanzar y Retroceder Página, y la tecla de tabulación. Además, cuando dichas teclas se presionan en combinación con la tecla Ctrl se incrementa su unidad de movimiento, por ejemplo, cuando se presiona la flecha derecha, ésta mueve un caracter a la derecha en un campo de texto y al presionarla en combinación con la tecla Ctrl.hace que se desplace una palabra a la derecha en un campo de texto. A diferencia del mouse, las teclas de navegación generalmente afectan las selecciones existentes cuando se utilizan.

Uso de los menús. Los comandos de Windows se presentan en menús. Un *menú* es una lista de comandos disponibles en la ventana de una aplicación Cada aplicación tienen sus propios menús, cuyos nombres aparecen en la barra de menús situada en el extremos superior de cada ventana de aplicación. Windows permite seleccionar un menú y, a continuación, un comando del mismo. Al seleccionar ese comando se ejecutará la acción correspondiente. Para seleccionar un menú hay que señalar el nombre del menú y hacer clic en el mismo para abrirlo. Para acceder directamente a un elemento de menú determinado, puede arrastrar el apuntador de selección por el menú y dar nuevamente clic en el comando deseado.

Existen tres tipos de menús, los *menús desplegables*, los *menús contextuales* y los *menús en cascada*. Un menú desplegable o simplemente menú aparece como un panel con la lista de sus elementos ordenados en una columna. Los menús contextuales o menús de atajo proporcionan una manera eficiente para el usuario de acceder a las operaciones de los objetos, generalmente son activados cuando se da un clic derecho del mouse. Un menú en cascada, también conocido como menú jerárquico o menú hijo, es un submenú de un elemento de menú y los nombres de los comandos para los que se abre un menú en cascada

están seguidos de una flecha hacia la derecha. En la figura A.6 se encuentran ejemplos de menús.

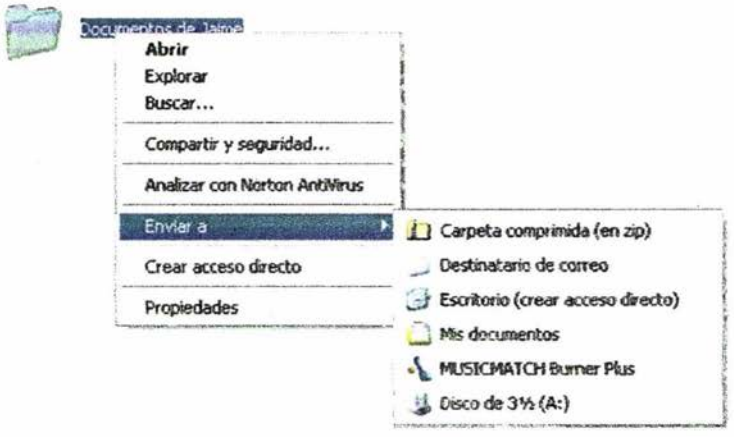


Fig. A.6 Menú contextual y menú en cascada.

A.3 Trabajando con archivos y carpetas

El Explorador es una herramienta indispensable en el sistema operativo Windows, ya que con ella podemos organizar y controlar los archivos y carpetas de los distintos sistemas de almacenamiento que dispongamos, como el disco duro o la unidad de CD.

Por medio del Explorador de Windows es posible realizar tareas de administración de archivos y carpetas como visualizarlos, eliminarlos, copiarlos, moverlos o cambiarles el nombre.

La forma más rápida de abrir el Explorador es mediante el icono que puede estar en la barra de tareas o en el escritorio. Si no se tiene creado este icono entonces seguir la siguiente secuentcia:

Pulsar el botón de *Inicio*, selecciona *Todos los programas*, selecciona *Accesorios* y selecciona el *Explorador de Windows*.

La ventana de la figura A.7 es una ventana similar a la que se puede encontrar al abrir el Explorador de Windows, sin embargo, tal vez el aspecto cambie un poco ya que el usuario puede configurarla a su gusto. A continuación se explicarán las partes que lo componen.

Fig. A.7 Explorando con Mi PC

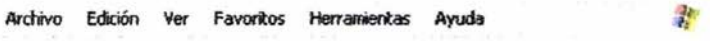
El Explorador consta principalmente de dos secciones, en la sección izquierda se encuentra el *árbol de directorios* y en la sección derecha encontramos la *vista de carpetas*.

El *árbol de directorios* indica la lista de unidades y carpetas que se tienen en la computadora. Únicamente aparecen unidades y carpetas, no archivos. En este árbol aparecerán carpetas como Mis documentos, Mi PC, Mis sitios de red y la Papelera de reciclaje.

La *vista de carpetas* se mostrará el contenido de la carpeta que tengamos abierta en la sección del árbol de directorios. En este caso aparecen los archivos que hay en la carpeta *texto*. De acuerdo con el tipo de vista que tengamos activado se verá distinto tipo de información sobre los archivos. Al dar clic en el menú *Ver* podemos observar que tenemos las opciones de vista: **Vistas en miniatura**, **Mosaicos**, **Iconos**, **Lista** y **Detalles**. En este caso se muestran el nombre del archivo, la fecha de modificación y el tipo de documento que se encuentran en la vista de Mosaicos.

Como en cualquier ventana de Windows tenemos la *barra de título* que muestra el nombre de la carpeta en la que nos encontramos.

Tenemos también la *barra de menús* que contiene los menús que nos permitirán acceder a todas las operaciones que se pueden realizar sobre un archivo o carpeta. Al dar clic en cada menú se mostrarán las opciones que lo componen.



La *barra estándar* contiene botones para agilizar las operaciones más utilizadas.

Si no se encuentra visible esta barra seleccionemos el menú *Ver*, elegir la opción Barra **de herramientas** y a continuación seleccionar la opción **botones estándar**.

El botón 

nos permitirá ir a la página anterior que hayamos visto. El botón que se encuentra al lado, cuando esta activo, permite ir una página hacia adelante.

El botón Arriba  nos permitirá subir de nivel, es decir, situarnos en la carpeta que contiene la carpeta actual.

El botón de Búsqueda  nos muestra una ventana en la que podemos buscar el archivo que nosotros le digamos.

El botón  Carpetas hace que en la parte izquierda de la ventana se vea la estructura de las carpetas o bién una zona con las tareas más frecuentes según el archivo o la carpeta que tengamos seleccionado, en esta zona podemos encontrar, entre otros, los siguientes botones:

- El botón  permite **copiar** a otra carpeta archivos o incluso otra carpeta.
- El botón  permite **mover** carpetas o archivos o otro lugar.
- Con el botón  podremos **eliminar** los elementos seleccionados.
- El botón  permite **cambiar el nombre** al archivo o a la carpeta seleccionados.

Cuando no tenemos ningún elemento seleccionado podemos tener el botón  que nos permite **crear una nueva carpeta** dentro de la carpeta actual.

Cuando se realice una de las operaciones de copiar o mover elementos aparecerá un cuadro de diálogo, como se ve en la figura A.8, mostrando el título del tipo de operación a realizar, el árbol de directorios para que seleccionemos la carpeta a donde queremos mover o copiar los elementos seleccionados y las opciones de *Cancelar* y *Crear nueva carpeta*, así como la opción para efectuar la operación, en este caso *Copiar*.

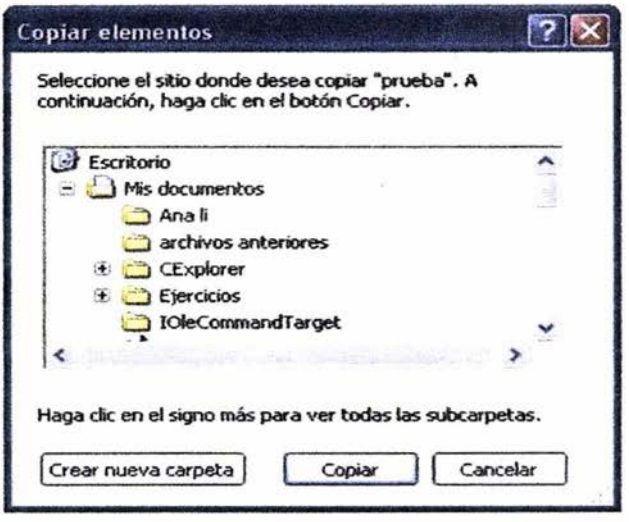


Fig. A.8 Cuadro de diálogo para copiar los elementos seleccionados.

El resultado que obtenemos al copiar o al mover es similar, ya que con los dos tendremos el archivo o carpeta en el lugar que nosotros deseemos, pero con la diferencia de que cuando se mueve, la carpeta o archivo original desaparece mientras que al copiar se mantiene el archivo o carpeta original. Entonces mover un elemento consiste en cortarlo y pegarlo en otro sitio.

Cuando decidimos eliminar los elementos seleccionados lo que hacemos en realidad es moverlos a la papelera de reciclaje, un espacio donde son almacenados en caso de que queramos recuperarlos. Y permancen ahí hasta que vaciemos la papelera o decidamos restaurar los elementos, ya sea en su ubicación original o en una ubicación distinta.

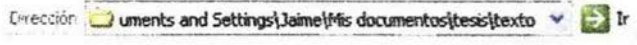
Al elegir la opción para cambiar el nombre del elemento seleccionado, aparece el nombre del elemento resaltado y encerrado en un recuadro  donde podemos escribir el nuevo nombre y para hacer que tales cambios se lleven a cabo podemos pulsar la tecla **Intro** o hacer clic fuera del elemento seleccionado.

El Explorador de Windows nos da la posibilidad de deshacer el último cambio que hayamos realizado en la estructura de directorios. Para ello en el menú *Edición* tenemos la opción **Deshacer**.

Por otra parte, Windows también nos da la posibilidad de *Abrir* un documento o ejecutar una aplicación haciendo doble clic en el elemento que deseamos o en el menú *Archivo* del Explorador elegir la opción **Abrir**.

El último botón,  , nos permite cambiar las vistas de las carpetas.

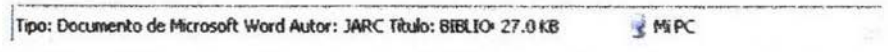
La *barra de direcciones* es muy conocida en Internet porque es en ella donde aparece la dirección de la página web que estemos visualizando. En el explorador de Windows el funcionamiento es similar, pero muestra el nombre de la carpeta en la que nos encontramos.



Dando clic en la flecha negra aparecerá la estructura con los discos de nuestra computadora.

Si escribimos un nombre en la barra de direcciones y pulsamos la fecha de Ir Windows buscará ese nombre en Internet.

La *barra de estado* muestra información adicional sobre los elementos que tenemos seleccionados. Esta barra es opcional, para activarla ir al menú *Ver*, y seleccionar la opción **Barra de estado**.



De la figura de la ventana del explorador de Windows en la cual tenemos seleccionado un objeto, se trata de un documento de Microsoft Word y en la barra de estado muestra la información sobre que tipo de archivo es, su fecha de modificación y su tamaño en bytes, además de otros atributos que son específicos de un documento de Word, como el autor y el título.

En el caso de que se trate de una unidad o una carpeta, la barra de estado mostrará el número de carpetas y archivos que contiene y el espacio libre que queda en la unidad en la cual nos encontramos.

Si se trata de varios objetos seleccionados, en la barra de estado se podrá apreciar la información del número de objetos seleccionados y el tamaño total de los archivos

seleccionados. Esta barra es bastante útil ya que podemos saber rápidamente, por ejemplo si los archivos seleccionados entran en un disquete.

Descritos los elementos anteriores podemos tener una idea general del entorno de usuario de Windows, aunque no son todas las características que el sistema posee tenemos los elementos necesarios para entender el presente trabajo por lo que ahora nos remitiremos a exponer el ambiente de Word XP, versión 2002.

Apéndice B

El entorno de usuario de Word

Microsoft Word es un procesador de palabras que permite, de manera fácil y práctica, crear todo tipo de textos con diversas presentaciones dependiendo del tipo de trabajo que se desee elaborar: informes, cartas, artículos, incluso libros completos y páginas web, entre otros.

Mucha de la versatilidad de Word proviene de su interacción con el entorno de Windows. Al arrancar Word aparece una pantalla como en la figura B.1, donde se muestran también los elementos que lo componen. Nota: La figura mostrada puede no coincidir con las de los equipos de otros usuarios, ya que cada usuario puede decidir que elementos quiere que se vean en cada momento, sin embargo, lo que interesa es describir los componentes que nos van a servir para trabajar con Word.

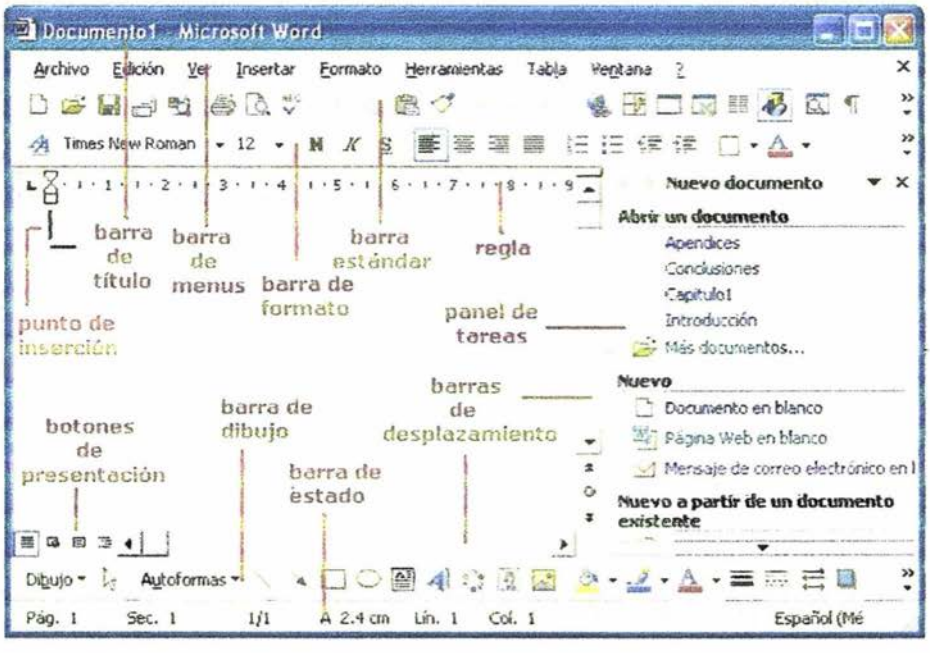


Fig. B.1 Componentes de Microsoft Word.

Desde la barra de menús se pueden ejecutar todos los comandos de Word. En Word 2002 la barra de menús tiene un comportamiento "inteligente", que consiste, básicamente, en mostrar sólo los comandos más importantes y los que el usuario va utilizando.

La *barra de herramientas estándar* contiene iconos para ejecutar de forma inmediata algunos de los comandos más habituales, como *Guardar*, *Copiar*, *Pegar*, etc.

La *barra de formato* contiene las operaciones más comunes sobre formatos, crear estilos y seleccionar todo el texto que tenga el mismo formato. Con ella es posible poner texto en negritas, cambiar el tipo de fuente, etcétera.

El elemento que define la posición dentro de un documento en Word es el *punto de inserción* y nos indica dónde se va a escribir la próxima letra que tecleemos y con ayuda de las *barras de desplazamiento* nos colocaremos en la parte del documento que deseemos.

El *panel de tareas* ofrece una ubicación para las acciones más utilizadas a la hora de trabajar en Office XP. El panel de tareas evita tener que desplazarse una y otra vez a la barra de menús y buscar opciones en listas de acciones. El panel de tareas es sensible al contexto, es decir, que mostrará información diferente según lo que estemos haciendo en el momento de abrirlo. Se puede abrir un archivo, crear un documento nuevo o aplicar estilos simplemente con hacer clic con el mouse. El panel de tareas se encuentra a la derecha de la pantalla y aparece al iniciar por primera vez un programa de Office XP.

Los *botones de presentación* nos muestran las diferentes formas de ver un mismo documento.

B.1 Edición básica

En esta sección se mostrará lo necesario para editar documentos. Desplazarnos, seleccionar, eliminar, copiar, pegar y deshacer. También veremos cómo buscar y reemplazar palabras.

B.1.1 Desplazarse por un documento

Una de las ventajas que han aportado los procesadores de texto es la facilidad para modificar y corregir. El primer paso en ese proceso es colocarnos en el lugar donde vamos a efectuar la modificación.

Por otra parte, cuando estemos viendo un documento Word que no cabe en una pantalla, necesitaremos movernos por el documento para colocarnos en la parte que nos interese. Tanto en un caso como en otro será interesante conocer, además de las flechas de

desplazamiento, otras formas que existen para desplazarse por el documento, y así poder elegir la más útil en cada momento.

Podemos utilizar desplazamientos cortos dentro de una misma pantalla:

- Con el mouse desplacemos el apuntador hasta el punto que deseemos y al hacer clic, el punto de inserción se colocará en ese lugar.

  Si utilizamos las teclas de dirección. Las teclas izquierda y derecha desplazan el punto de inserción un caracter a la izquierda o la derecha según corresponda, y las teclas arriba y abajo desplazan el punto de inserción una línea en la dirección correspondiente. La tecla Fin nos lleva al final de la línea y la tecla Inicio al principio ella.

- Con el teclado podemos desplazarnos utilizando las combinaciones siguientes:

Para desplazarse	***Presione las teclas***
Una palabra a la izquierda	Crtl + flecha izquierda
Una palabra a la derecha	Crtl + flecha derecha
Un párrafo arriba	Crtl + flecha arriba
Un párrafo abajo	Crtl + flecha abajo

También podemos desplazarnos a lo largo de todo el documento:

- Utilizando las teclas *AvPág* y *RePág* que avanzan y retroceden una pantalla completa, respectivamente. Tomando en cuenta que no es lo mismo una pantalla que una página, ya que el tamaño de la pantalla está limitada por el monitor, mientras que la longitud de la página la definimos nosotros.
- Utilizando las siguientes combinaciones de teclas:

Para desplazarse	***Presione las teclas***
Una página adelante	Crtl + AvPág.
Una página atrás	Crtl + RePág.
Al principio del documento	Crtl + Inicio
Al final del documento	Crtl + Fin

B.1.2 Seleccionar

Para realizar muchas operaciones como copiar, cambiar el formato, etcétera, previamente hay que decirle a Word en qué parte del texto tiene que actuar, en esto consiste seleccionar.

El texto seleccionado se identifica claramente porque se pone en video inverso, es decir, el fondo negro y los caracteres en blanco. Podemos seleccionar con el mouse o con el teclado. Para seleccionar mediante el mouse podemos:

- Arrastrarlo, colocando el apuntador al principio de la selección, presionar el botón izquierdo y, sin soltar el botón, mover el apuntador hasta el final de la selección. Observaremos como lo seleccionado aparece en vídeo inverso.
- Hacer clic y luego doble clic, colocando el apuntador en una palabra y al hacer doble clic, la palabra completa quedará seleccionada. O también podemos colocar el apuntador justo al inicio de una línea, donde veremos que el apuntador cambia de forma y se convierte en una flecha; después hacemos clic y la línea completa quedará seleccionada; si hacemos doble clic otra vez, el párrafo completo quedará seleccionado.
- Y para seleccionar un gráfico o una imagen basta con hacer clic encima del objeto y el gráfico quedará enmarcado por un recuadro negro.

Por su parte, para seleccionar mediante el teclado podemos abreviarlo con las siguientes combinaciones:

Para seleccionar	***Presione las teclas***
Un carácter a la derecha	Mayús. + flecha derecha
Un carácter a la izquierda	Mayús. + flecha izquierda
Palabra a la derecha	Ctrl + Mayús.+ flecha derecha
Palabra a la izquierda	Ctrl + Mayús.+ flecha izquierda
Hasta el final de la línea.	Mayús. + Fin
Hasta el principio de la línea.	Mayús. + Inicio
Una línea abajo	Mayús. + flecha abajo
Una línea arriba	Mayús. + flecha arriba
Hasta el final del párrafo	Ctrl + Mayús. + flecha abajo
Hasta el principio del párrafo	Ctrl + Mayús. + flecha arriba
Una pantalla abajo	Mayús. + AvPág
Una pantalla arriba	Mayús. + RePág
Todo el documento	Ctrl + E

B.1.3 Eliminar

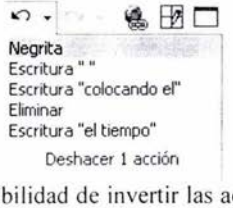
Para borrar o eliminar lo seleccionado basta con presionar la tecla Supr, otra forma de borrar sin seleccionar previamente es utilizando las teclas, como se indica en la siguiente tabla:

Para borrar	*Presione las teclas*
Un carácter a la izquierda	Retroceso (BackSpace)
Una palabra a la izquierda	Ctrl + Retroceso
Un carácter a la derecha	Supr
Una palabra a la derecha	Crtl + Supr

B.1.4 Deshacer y rehacer cambios

Word nos da la posibilidad de corregir los errores fácilmente. Por ejemplo, si acabáramos de borrar un párrafo completo y nos diéramos cuenta que ese no era el párrafo que deseábamos borrar. Con un solo clic podríamos deshacer la acción errónea y recuperar el párrafo.

- Para deshacer la ultima acción realizada, pulsamos el icono deshacer  de la barra de herramientas. O con la ayuda del menú Edición, eligiendo la opción Deshacer escritura, en este caso. Otra forma más de deshacer los cambios es pulsando CTRL + Z.
- Para deshacer las últimas acciones realizadas hacemos clic en la flecha que se encuentra a la derecha del icono de deshacer. Entonces se mostrará una lista con las últimas acciones para deshacer, colocando el apuntador en esa lista podremos deshacer varias acciones a la vez. Por ejemplo, al colocar el apuntador en la tercera línea de la lista desharemos las tres últimas acciones. Esta lista nos indica de forma abreviada cuáles son las diferentes acciones que podemos deshacer, por ejemplo, en la lista tenemos la línea: Escritura "colocando el", que nos dice que podemos deshacer la escritura de la frase que empieza por "colocando el".



Word también nos ofrece la posibilidad de invertir las acciones del comando deshacer. Por ejemplo, si ponemos en letra cursiva un párrafo y deshacemos la acción porque pensamos que no queda bien con este estilo, y al cabo de un momento cambiamos de opinión, podemos dar clic en el icono Rehacer  y volver a dejarlo en cursiva. Asimismo si

queremos rehacer varias acciones este icono también cuenta con una lista, similar a la del comando deshacer, para efectuar los cambios deseados.

B.1.5 Copiar, cortar y pegar

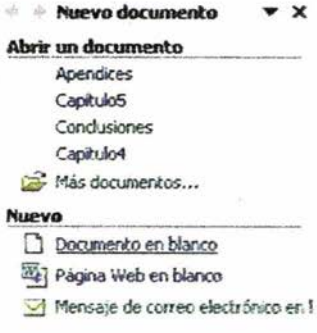
Cuando hablamos de copiar nos referimos a colocar una copia del texto o los objetos seleccionados en otro lugar; mientras que cuando hablamos de cortar queremos decir quitar los elementos seleccionados de un lugar para llevarlo a otro. Se pueden utilizar varios métodos.

Para llevar a cabo estas operaciones necesitamos seleccionar el elemento, ya sea un carácter, una palabra, un párrafo, etcétera, que queramos copiar o cortar. Después hacer clic en el icono copiar o cortar de la barra de herramientas estándar. Enseguida coloquemos el apuntador en el punto de destino y hagamos clic en el icono pegar . Con el menú Edición se pueden llevar a cabo las mismas acciones eligiendo las opciones correspondientes a estos comandos.

Con el teclado, primero seleccionamos el texto que deseamos copiar o mover y después para copiar teclear Ctrl + C o para cortar teclear Ctrl + X. Por último, ir a la posición donde vamos a pegar y teclear Ctrl + V.

B.2 Crear, guardar y abrir documentos

Para crear un nuevo documento podemos dar clic en el icono de la barra de herramientas estándar donde automáticamente se crea un documento en blanco para que podamos trabajar con él. Si utilizamos el comando Nuevo del menú Archivo aparece el panel de Nuevo documento donde podemos elegir entre otras cosas abrir un documento reciente o crear un documento en blanco.



Para guardar documentos podemos utilizar los comandos Guardar y Guardar como... del menú Archivo o dar clic en el icono de la barra de herramientas estándar. Al utilizar el comando Guardar como, Word mostrará un cuadro de diálogo como el que se ve en la figura B.2, que permite cambiar el nombre del archivo, el tipo de documento y la carpeta donde se guardará el documento. Al utilizar el comando Guardar solamente se guardarán los cambios que se hayan efectuado en el documento en su ubicación actual. Sin embargo, si se utiliza el comando Guardar con un documento nuevo, es decir, que no ha sido guardado antes, se abrirá el mismo cuadro de diálogo para Guardar como.

Para guardar un archivo se deben tomar en cuenta los campos Guardar en, Nombre de archivo y Guardar como tipo, del cuadro de diálogo que indican la carpeta donde se

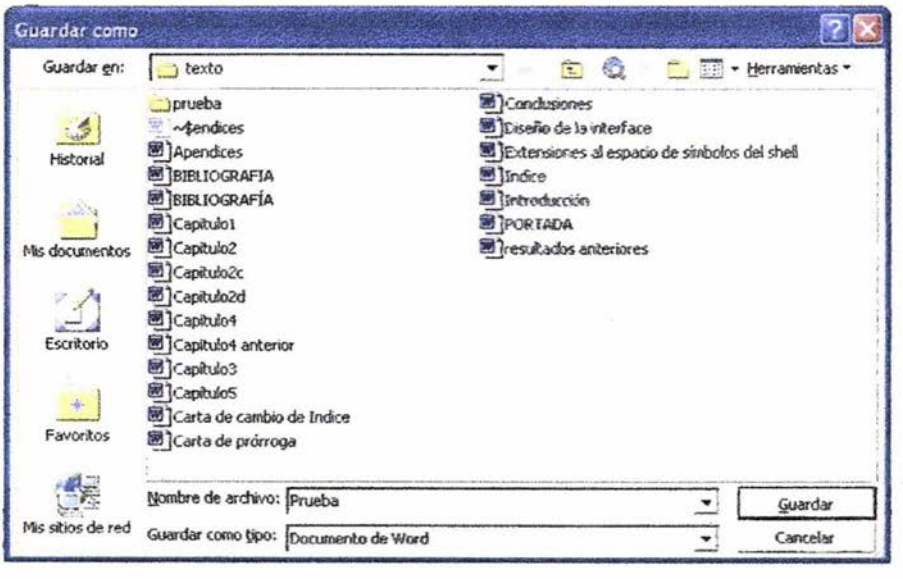


Fig. B.2 Cuadro de diálogo para guardar documentos de Word.

guardará el documento, el nombre del documento y el tipo de documento que será guardado, respectivamente. Por omisión el campo de Guardar en utiliza la carpeta de Mis Documentos y el tipo de documento es el Documento de Word. Asimismo en la parte central de este cuadro de diálogo se muestran los documentos de Word y las carpetas que hay dentro de la carpeta actual, que en este caso se llama texto.

En caso de que ya exista el documento que se está guardando en la ubicación actual. Word mostrará un cuadro de diálogo preguntando si se desea reemplazar el archivo existente.

Otras opciones que presenta el cuadro de diálogo son: el botón que permite buscar en Internet. Cuando se da clic en el botón se elimina el archivo o carpeta seleccionados. Haciendo clic en el icono se abre un menú en el que se puede seleccionar el formato con el que se verá la lista de documentos.

En este cuadro de diálogo existe también un menú Herramientas para que se puedan elegir una de las opciones disponibles que permiten eliminar, cambiar de nombre y agregar a Favoritos el archivo seleccionado. Si la computadora forma parte de una red se puede conectar un archivo remoto a una unidad de red. Se pueden ver también las propiedades del archivo. Asimismo se puede acceder a las Opciones al guardar, Opciones de seguridad, Opciones Web y Comprimir imágenes. Por último, se puede guardar una versión del archivo.

Para comenzar a trabajar con un documento necesitamos abrirlo por lo que podemos dar clic directamente en el icono o elegir la opción Abrir del Menú Archivo. A

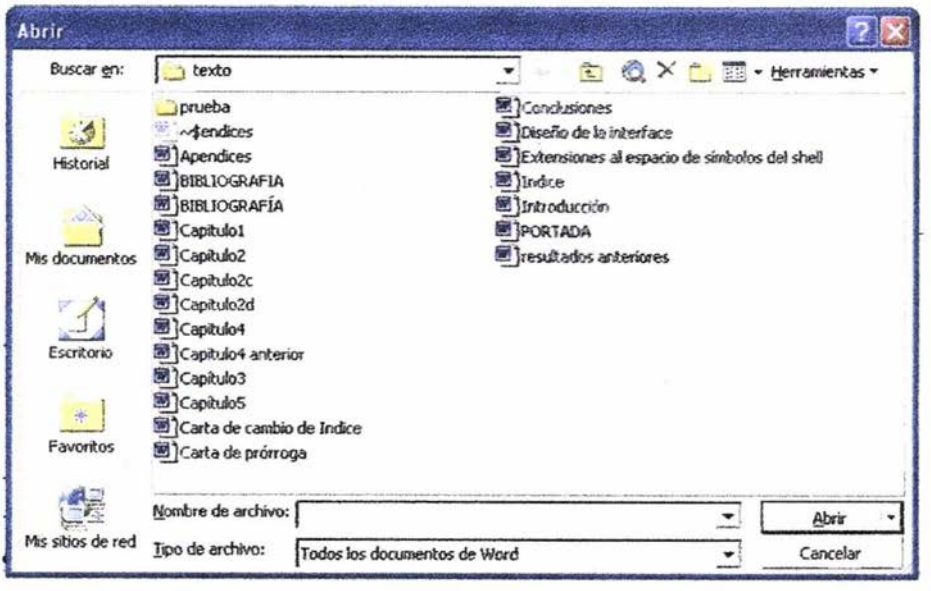


Fig. B.3 Cuadro de diálogo para abrir documentos.

continuación aparecerá un cuadro de diálogo como el de la figura B.3. Éste es similar al del comando Guardar. La diferencia principal estriba en que dispone de algunas opciones para buscar el documento que queremos abrir.

Normalmente podremos localizar el documento que queremos abrir en la lista que se nos muestra, y bastará con hacer doble clic sobre él para abrirlo. Si conocemos el nombre del documento bastará escribirlo en el campo Nombre del archivo y hacer clic en el botón Abrir. Si no lo encontramos en la carpeta actual podemos buscarlo manualmente desplazándonos por la estructura de carpetas.

Si todavía no lo hemos encontrado, disponemos de la opción Buscar, dentro del comando Herramientas, mediante esta búsqueda automática, puedes especificar condiciones que debe cumplir el documento que se busca.

B.3 Formatos

B.3.1 Formato de un texto

Cuando hablamos del formato de un texto nos referimos el aspecto del texto o la forma de presentar el texto en el documento. Por ejemplo al hablar de poner una palabra en cursiva, alinear un párrafo a la izquierda o colocar un borde sombreado a una tabla se está hablando de las operaciones típicas de formato que sólo afectan a la forma en cómo vemos el texto, pero no al contenido del texto.

Aunque lo fundamental cuando escribimos un texto es lo que se dice en él, la forma en la que lo presentemos también tiene mucha importancia.

Un texto con un buen contenido pero con mal formato pierde mucha calidad. Por esta razón Word 2002 nos da la posibilidad de dar un formato atractivo a nuestro texto. En Word podemos clasificar las acciones que tienen que ver con el formato en tres grandes grupos:

1. *Formato de caracter.* Afectan a los caracteres en sí mismos como el tipo de letra o fuente, tamaño, color, etcétera.
2. *Formato de párrafo.* Afecta a grupos de caracteres como líneas y párrafos, por ejemplo alineación y sangrías.
3. *Otros formatos.* Aquí incluimos el resto de acciones que se pueden hacer sobre el formato como tabulaciones, cambio a mayúsculas, numeración y viñetas, bordes y sombreados, etcétera.

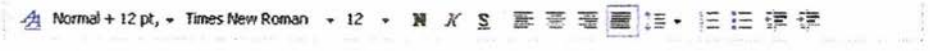
Además de las técnicas antes mencionadas existen otras posibilidades para dar formato al texto: *los estilos* y *las plantillas*. Los estilos y plantillas son adecuados para dar un formato definido previamente. Por ejemplo, en una empresa suele existir una plantilla para escribir cartas, de esa forma todas las cartas tienen un aspecto característico de esa empresa.

No siempre es necesario utilizar plantillas y estilos. Cuando queremos dar un determinado formato a una línea del documento no hace falta definir un estilo, lo haremos con los comandos básicos. Entonces los estilos y plantillas no son más que una agrupación de formatos básicos. Así que lo primero que hay que aprender es a manejar esos formatos básicos, como se verá a continuación.

Formato caracter. Fuentes

Los caracteres corresponden a todas las letras, números, signos de puntuación y símbolos que se escriben como texto. Las letras escritas con WordArt o las letras incluidas en imágenes, no se consideran caracteres y por tanto no se les pueden aplicar los formatos que vamos a estudiar.

Veamos las posibilidades más comunes para variar el aspecto de los caracteres que están disponibles en la barra de formato o en una parte de la barra estándar: *fuente, tamaño y estilo de fuente*.



Fuente. Un apartado a elegir con cuidado es la fuente del texto ya que determinará en gran medida el aspecto del texto.

Para cambiar el tipo de letra o fuente lo primero que tenemos que hacer es seleccionar el texto sobre el que queremos realizar el cambio. A continuación hacer clic sobre la flecha que hay al lado de la fuente actual, esto ocasionará que se abra una ventana con las fuentes disponibles.

Observemos que el nombre de cada fuente está representado en este menú de fuentes, de manera tal que podemos ver el aspecto que tiene antes de aplicarlas.

El menú desplegable tiene dos zonas separadas por una doble línea horizontal, en la parte superior se encuentran las últimas fuentes utilizadas y en la parte inferior todas las disponibles.

Una vez hemos encontrado la fuente que buscamos basta con hacer clic sobre ella para aplicarla.

Si conocemos el nombre de la fuente que queremos, podemos hacer clic sobre el recuadro y teclearlo directamente, en lugar de desplazarnos por el menú desplegable para buscar la fuente.

Las fuentes *TrueType* aparecen con una doble T delante. Este tipo de fuente se verá igual en la pantalla que en la impresora.

El tamaño. De manera similar podemos cambiar el tamaño de la fuente. Seleccionando el texto y haciendo clic en la fecha que se encuentra junto al cuadro de tamaño para elegir el tamaño que deseemos, o podemos escribirlo directamente.

La unidad de medida es el punto (72 puntos = 1 pulgada = 2.54 cm.), los tamaños más utilizados son 10 y 12 puntos.

El estilo. Una vez fijada la fuente y el tamaño podemos cambiar el estilo a uno de los tres disponibles: **negrita**, *cursiva* y <u>subrayado</u>  Basta seleccionar el texto y hacer clic en el botón correspondiente.

Observemos que al aplicar un estilo, el botón correspondiente queda presionado. Para quitar un estilo que hemos aplicado previamente, seleccionamos el texto y volvemos a hacer clic sobre el estilo.

También se pueden aplicar varios estilos a la vez, como ***negrita y cursiva***. Simplemente hay que aplicar los estilos consecutivamente.

Mediante las opciones del menú **Formato**, en la opción *Fuente* se pueden manejar las opciones que acabamos de ver y otras más como el color de los caracteres, el subrayado, los subíndices, etcétera.

Formato párrafo

En Word 2002, un *párrafo* es el texto comprendido entre dos marcas de párrafo ¶, normalmente las marcas de párrafo no se ven, para hacerlas visibles, hacer clic en el icono marca de párrafo de la barra estándar.

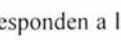
Se inserta una marca de párrafo cuando se pulsa la tecla de retorno de carro o INTRO. Cuando estamos introduciendo texto y llegamos al final de la línea automáticamente el texto continúa en la siguiente línea, pero no se inserta marca de párrafo.

Al insertar un párrafo, este toma las mismas características de formato del párrafo anterior. Para cambiar las características de formato de un párrafo, basta con seleccionar su marca de párrafo y modificar las características que queramos.

Los párrafos son unidades dentro del documento Word que tienen sus propias características de formato, pudiendo haber diferentes formatos entre dos párrafo.

Las marcas de párrafo contienen los códigos que definen el formato del párrafo en el que se encuentran. Manipular una marca de párrafo tiene consecuencias sobre el formato de ese párrafo. Antes de borrar texto, es conveniente hacer visibles las marcas de párrafo para evitar borrar una marca de párrafo accidentalmente. Si queremos borrar todo el párrafo también debemos borrar su marca de párrafo.

Las características más importantes de formato de párrafo son *la alineación* y *la sangría* y suelen estar disponibles en la barra de formato. También es posible acceder a ellas desde el menú **Formato**, en la opción *Párrafo*.

Alineación. Cuando hablamos de alinear un párrafo nos referimos, normalmente, a su alineación respecto de los márgenes de la página. Estos son los botones para fijar la alineación  , que corresponden a las alineaciones izquierda, centrada, derecha y justificada, respectivamente. Por ejemplo:

Este párrafo tiene establecida alineación izquierda.

Este párrafo tiene establecida la alineación centrada.

Este párrafo tiene establecida alineación derecha.

Y este párrafo tiene una alineación justificada.

Sangría. Aplicar una sangría a un párrafo es desplazar el párrafo una cierta distancia hacia la derecha o izquierda. Por omisión el desplazamiento es de 1.25 cm. La sangría se realiza seleccionando el párrafo y haciendo clic en uno de estos botones  de la barra de formato, de acuerdo a como deseemos desplazar ya sea hacia la izquierda o hacia la derecha, respectivamente.

Otros formatos. Tabulaciones

Las tabulaciones son posiciones fijas a las cuales se desplaza el apuntador cuando pulsamos la tecla de tabulación TAB. Para ir a la tabulación anterior pulsar MAYÚS + TAB.

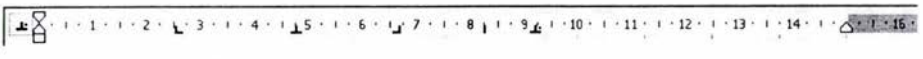
Cuando necesitamos insertar texto en columnas o tablas pueden ser muy útiles las tabulaciones. Word 2002 tiene por defecto definidas tabulaciones cada 1,25 cm. Pero se pueden establecer tabulaciones en las posiciones que deseemos. Además podemos definir la alineación para cada tabulación.

Vamos a ver cómo establecer tabulaciones utilizando la regla horizontal. Tener en cuenta que las tabulaciones afectan solo al párrafo en el que se definen. Para establecer cada tabulación se deben realizar estos dos pasos:

1. Hacer clic en el extremo izquierdo de la regla horizontal para seleccionar la alineación de la tabulación que vamos a insertar. Por cada clic que hagamos irán apareciendo rotativamente las siguientes alineaciones: *izquierda, centrada, derecha, decimal y línea de separación.*
2. Colocar el apuntador en la regla horizontal en la posición en la que deseemos establecer la tabulación y hacer clic, veremos como se inserta el icono con la tabulación seleccionada.

Si queremos modificar la posición de una tabulación basta colocar el apuntador en el icono de esa tabulación y hacer clic y arrastrarlo a la nueva posición.

Por ejemplo, en la regla de la imagen se han establecido las siguientes tabulaciones:



Izquierda en la posición 2.5

Centrada en la posición 4.7

Derecha en la posición 6.7

Linea vertical de separación en la posición 8.2

Decimal en la posición 9.2

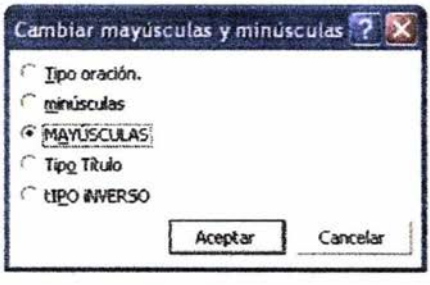
Teniendo en cuenta que la escala varia un poco las proporciones quedaría una tabla parecida a esta:

Localidad	Comunidad	País	Indice TGH
Vigo	Galicia	España	44,5
Don Benito	Extremadura	España	3,451
Glasgow	Escocia	Reino Unido	125,5

Observe la alineación de cada tabulación, la alineación decimal coloca la coma decimal en la misma columna. También se pueden fijar las tabulaciones mediante el menú **Formato**, *Tabulaciones*, dando la posición exacta en cm. y especificando un carácter de relleno, como puntos o guiones, entre las tabulaciones.

Cambio a mayúsculas

En el *menú* **Formato**, la opción *Cambiar a mayúsculas y minúsculas...* nos ofrece cinco posibilidades para cambiar las mayúsculas del texto seleccionado:



Tipo oración. La primera letra después de cada punto en mayúsculas el resto en minúsculas.

minúsculas. Todas las letras en minúsculas.

MAYÚSCULAS. Todas la letras en mayúsculas.

Tipo Título. La primera letra de cada palabra en mayúscula y el resto de la palabra en minúsculas.

tIPO iNVERSO. La primera letra de cada palabra en minúscula y el resto de la palabra en mayúsculas.

También podemos manejar las mayúsculas mediante el teclado presionando Mayúsculas + F3, las palabras del texto seleccionado cambian alternativamente de las tres formas, siguientes: MAYÚSCULAS, minúsculas y Tipo oración.

Copiar formato

En la barra estándar tenemos disponible el icono para copiar formato  . Este icono permite copiar las características de formato de un texto para aplicarlas a otros textos, solo copia el formato dejando el propio texto igual que estaba. Este icono se puede utilizar para copiar un formato una vez o para copiar un formato varias veces.

Una vez: Hacer clic sobre la palabra de la cual queremos copiar el formato; ir a la barra de herramientas y hacer clic en el icono de copiar formato, el apuntador tomará la forma de brocha, colocarlo sobre la palabra en la que queremos copiar el formato, y hacer clic sobre ella.

Varias veces: Hacer clic sobre la palabra de la cual queremos copiar el formato, hacer doble clic en el icono de copiar formato el apuntador tomará la forma de brocha, a continuación hacer clic sobre tantas palabras como se desee, y para acabar volver a hacer clic en el icono de copiar formato.

En ambos casos, si queremos copiar el formato sobre un conjunto de palabras o líneas, una vez el apuntador tome la forma de brocha, seleccionar las palabras o líneas y el nuevo formato se copiará sobre ellas. Copiar formato, copia tanto el formato carácter como el formato párrafo.

Puede ser útil copiar formato, por ejemplo, en el caso que tengamos un documento con varios títulos de puntos o apartados, si decidimos cambiar el formato de esos títulos bastaría con hacerlo en el primer título y luego copiar el formato a los demás títulos del documento.

B.4 Impresión

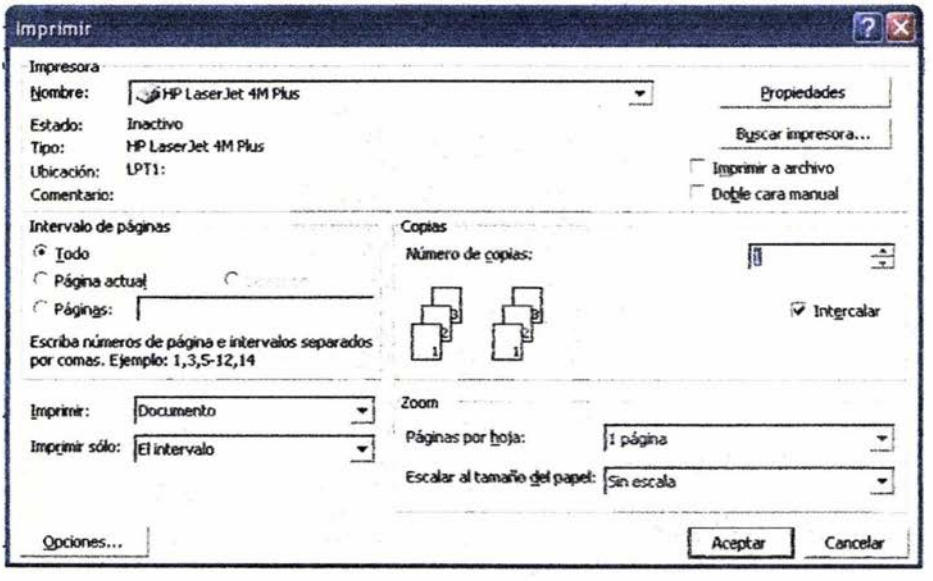
Se puede imprimir el documento de dos formas:

a) Desde el icono imprimir  de la barra estándar. Se utiliza cuando no queremos cambiar ninguna característica de impresión y se desea imprimir todo el documento con las opciones definidas hasta ese momento.

b) Desde el menú **Archivo**, *Imprimir* (Ctrl + P). Cuando queremos cambiar alguna característica de impresión. Por ejemplo, el número de copias, imprimir sólo alguna página del documento, etcétera.

Antes de mandar la primera impresión del documento, es conveniente comprobar las opciones definidas en ese momento, para ello tenemos que utilizar el menú Imprimir.

Desde el menú Archivo, seleccionar Imprimir o bien directamente Ctrl + P, aparecerá el cuadro de diálogo que se muestra a continuación.



Rellenamos las opciones deseadas y pulsamos el botón Aceptar.

Haciendo clic en el botón Opciones, se abre otra ventana en la que podemos seleccionar varias opciones, entre otras la de Orden Inverso que resulta muy útil cuando tenemos una impresora que deja las hojas boca arriba, si tenemos esta opción activada, se empieza por imprimir la última hoja dejando de esta manera las páginas ordenadas.

Las diferentes opciones que ofrece cada elemento de este cuadro de diálogo son:

Impresora. En el recuadro Impresora, podemos elegir la impresora por la que queremos que salga la impresión. Se utiliza cuando tenemos más de una impresora conectadas a nuestro equipo, ya sea directamente o por red. Al dar clic en el botón *Propiedades* podemos

cambiar algunos parámetros como el tipo de papel, la orientación del papel, si queremos impresión en color o blanco/negro, etcétera.

Intervalo de páginas. En este recuadro indicamos las páginas que queremos imprimir, basta con dar clic en la opción deseada:

- *Todo*. Imprime todo el documento.
- *Página actual*. Imprime la página en la que se encuentra situado el punto de inserción en ese momento.
- *Selección*. Si tenemos seleccionada alguna parte del documento, podemos dar clic en esta opción para que nos imprima únicamente el texto seleccionado.
- *Páginas*. Permite indicar qué páginas queremos que nos imprima. Si queremos páginas salteadas, poner los números de página separados por coma, por ejemplo: 2,8,10 imprimiría las páginas 2, 8 y 10 Para indicar un intervalo, poner la página inicial y la página final separadas por un guión, por ejemplo: 4-7 imprimiría las páginas 4,5,6 y 7. Se puede indicar un intervalo sin página inicial para indicar 'hasta', por ejemplo: -3 imprimiría las paginas 1,2 y 3 (hasta la página 3); o bien sin página final para indicar 'desde', por ejemplo 12- imprimiría las páginas desde la página 12 hasta la última ambas inclusive. También podemos combinar cualquiera de las formas anteriores, por ejemplo:2,3,10- imprimiría las páginas 2,3,10,11...hasta la última.

Copias. En el recuadro Copias se indica el número de copias que queremos, es decir nos imprime lo que le hemos indicado en el recuadro Intervalo de páginas, tantas veces como indicamos en Número de copias.

Si la opción Intercalar no está activada, imprime una copia entera y después otra copia, mientras que si activamos Intercalar imprime todas las copias de cada página juntas.

Zoom. En el recuadro Zoom tenemos dos opciones.

- *Páginas por **hoja***. Permite elegir cuántas páginas por hoja deseamos, por ejemplo 4 imprimiría 4 páginas en una sola hoja de papel. De este modo podemos ahorrar mucho papel si queremos tener impresas las versiones provisionales de nuestros documentos.

- *Escalar al tamaño del papel.* Permite indicar el papel que tenemos en nuestra impresora, por ejemplo A4.

B.5 Tablas

Las tablas permiten organizar la información en filas y columnas, de forma que se pueden realizar operaciones y tratamientos sobre las filas y columnas. Por ejemplo, obtener el valor medio de los datos de una columna o para ordenar una lista de nombres.

Otra utilidad de las tablas es su uso para mejorar el diseño de los documentos ya que facilitan la distribución de los textos y gráficos contenidos en sus casillas. Esta característica se emplea sobre todo en la construcción de páginas Web para Internet.

Vemos, pues, que esta forma de organizar los datos es mucho más potente que utilizando las tabulaciones u otros métodos.

Una tabla está formada por *celdas* o casillas, agrupadas por filas y columnas, en cada casilla se puede insertar texto, números o gráficos.

B.5.1 Creación de tablas

Se puede crear una tabla de tres formas equivalentes: con el menú Tabla, con el icono de la barra estándar o dibujándola con el mouse, según el tipo de tabla será más útil un método u otro.

Menú Tabla

Para insertar una tabla debemos ir al menú **Tabla**, en el submenú **Insertar** elegir la opción *Tabla* y se abrirá un cuadro de diálogo en el que debemos indicar:

Tamaño de la tabla. Donde indicamos el número de columnas y el número de filas.

Autoajuste. Aquí tenemos tres opciones para definir las dimensiones de la tabla.

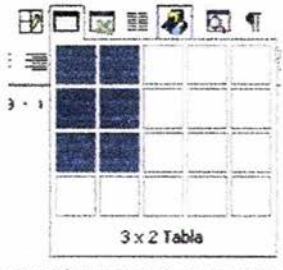
- *Ancho de columna fijo.* Si lo dejamos en automático ajustará el ancho para que la tabla ocupe todo el espacio entre los márgenes de la página.
- *Autoajustar al contenido.* El ancho dependerá de la cantidad de texto o gráficos que contenga cada columna.
- *Ajustar a la ventana.* El tamaño se ajusta al tamaño de la ventana del visualizador Web, si cambia el tamaño de la ventana, la tabla se ajusta al nuevo tamaño.

Autoformato. Mediante este botón podemos elegir entre varios formatos ya establecidos. Word aplicará las características del formato elegido a nuestra tabla.

Recordar dimensiones para tablas nuevas. Guarda las dimensiones, ajustes y formato actuales de la tabla para utilizarlos como valores por defecto cuando creemos nuevas tablas.

Icono

Al hacer clic en el icono tabla  de la barra estándar se abre una ventana como la que se muestra a la derecha. Moviendo el mouse dentro de la rejilla, podremos seleccionar el número de filas y columnas fácilmente .

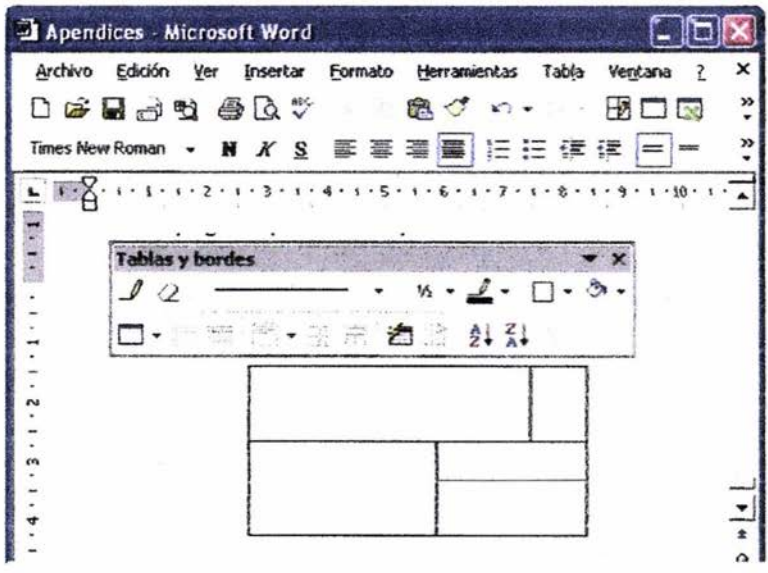


Por ejemplo, en el caso que se muestra se va a crear una tabla de 3 x 2, tres filas y dos columnas. Este método es el más sencillo y no permite opciones de formato, opciones que se pueden aplicar más adelante si lo creemos conveniente.

Dibujando una tabla

Ir a menú **Tabla** y seleccionar *Dibujar tabla* o hacer clic en el icono del lápiz de la barra de Tablas y bordes, el apuntador tomará la forma de un lápiz. Hacer clic y arrastrar el apuntador para dibujar el rectángulo con el tamaño total de la tabla. A continuación dibujar las filas y columnas, como si lo hiciéramos con un lápiz. En la barra de Tablas y bordes tenemos iconos para trabajar con las tablas que veremos más adelante.

Este método es más flexible a la hora de diseñar tablas irregulares, como la que se muestra en la imagen.



B.5.2 Desplazarse, seleccionar y borrar en las tablas

Una vez tenemos creada la tabla para introducir contenido en ella no hay diferencia entre introducir texto dentro o fuera de una tabla.

La mayor parte de las funciones sobre formato están disponibles en el texto de las tablas, se puede poner texto en negrita, cambiar el tamaño, se pueden incluir párrafos y se pueden alinear de varias formas como se hace normalmente. Nos podemos desplazar por las celdas con las teclas de movimiento del apuntador, se puede seleccionar, copiar y borrar el texto

de las celdas de forma normal, pero además, hay algunas formas específicas de desplazarse, seleccionar y borrar para las tablas que vamos a ver a continuación.

Desplazarse

Para colocarse en una celda, basta hacer clic en ella con el apuntador.

Para desplazarse	*Presione las teclas*
Una celda a la izquierda	MAY + TAB
Una celda a la derecha	TAB
Una celda arriba	flecha arriba
Una celda abajo	flecha abajo
Al principio de la fila	Alt + Inicio
Al final de la fila	Alt + Fin
Al principio de la columna	Alt + AvPág
Al final de la columna	Alt + RePág

Al pulsar la tecla TAB en la última celda de la tabla se crea una nueva fila.

Seleccionar

Para seleccionar una celda colocar el apuntador justo encima del lado izquierdo de celda, y cuando el apuntador tome la forma de una pequeña flecha negra inclinada hacer clic y la celda se pondrá en vídeo inverso.

Para seleccionar una columna colocar el apuntador justo encima de la columna, y cuando el apuntador tome la forma de una pequeña flecha negra que apunta hacia abajo hacer clic y la columna quedará en vídeo inverso.

Para seleccionar una fila hacer lo mismo que para seleccionar una celda pero haciendo doble clic o también colocando el apuntador a la izquierda de la fila y haciendo clic.

También se pueden seleccionar celdas, filas y columnas haciendo clic dentro de la celda y arrastrando a lo largo de las celdas que queramos seleccionar.

Borrar

Para borrar una celda, columna o una fila basta con seleccionarla y pulsar la tecla Retroceso (Backspace), si sólo queremos borrar su contenido pulsar la tecla Suprimir. Al borrar una

celda Word nos preguntará sobre la forma de desplazar el resto de las columnas. También se pueden realizar todas estas funciones desde el menú Tabla.

Todas las operaciones sobre las tablas se pueden ejecutar desde el menú Tabla, no obstante, para trabajar más cómodamente con las tablas, Word pone a nuestra disposición una barra de herramientas llamada Tablas y bordes con las funciones más habituales, también disponemos de un menú contextual especial para tablas, sin embargo estos se encuentran fuera del alcance de este texto.

BIBLIOGRAFÍA

Anónimo, ***Curso de Word 2002***. aulaClic. S.L.
http://www.aulaclic.es/word2002/f_word2002.htm. Octubre 2002.

Anónimo, ***Curso de Windows XP***. aulaClic. S.L.
http://www.aulaclic.com.es/winxp/f_windowsxp.htm. Marzo 2002.

Esposito, Dino. ***Visual C++ Windows Shell Programming***, Wrox. USA. 1998.

Flores Espinosa, Andrés. ***Reconocimiento de Palabras Aisladas***. Pontificia Universidad Católica del Perú. http://www.alek.pucp.edu.pe/~dflores/tesis/INDEX.html. 1998.

Gardida Degollado, Arturo, ***Reconocimiento de Palabras Aisladas Utilizando Cuantización Vectorial***, Tesis, FI. UNAM. 1998.

Herrera Camacho, José Abel. ***Apuntes de Procesamiento Digital de Voz***. UNAM. 2000.

Kruglinski, David J. et al. ***Programación avanzada con Visual C++ 6.0***. McGraw-Hill/Interamericana de España. Microsoft Press. España. 1999.

Pascual Jorge, et al. ***Programación Avanzada en Windows 2000. Con Visual C++ y MFC***. Osborne/McGraw-Hill. España. 2000.

Proakis, J. et al. ***Digital Signal Processign, Principles, Algorithms and Applications***. Macmillan Publishing, 1992 USA.

Rabiner, Lawrence R. y Biing-Hwang Juang. ***Fundamentals of Speech Recognition***. Prentice-Hall, 1993 E.U.

Tanenbaum, Andrew S. ***Modern Operating Systems***. 2a. ed. Prentice Hall. E.U. 2001.

Tischer, Michael y Bruno Jennrich, ***PC Interno 5. Programación de Sistemas***. Grupo Editor AlfaOmega. México. 1998.

User Experience Team. ***The Windows User Experience***. Microsoft Press. http://www.msdn.microsoft.com/library/en-us/dnwue/html/welcome.asp 1999.

HEMEROGRAFÍA

Anónimo. ***The Seven Simple Concepts of COM***.
http://www.clipcode.net/content/seven_simple_rules_of_com.htm. 2001

Microsoft. ***Windows Shell***. Microsoft Corporation. http::/msdn.microsoft.com/library/en-us/dnanchor/html/anch_WinShell.asp. 2004.

Campbel, David. ***Extending the Windows Explorer with namespace extensions***. Microsoft Systems Journal. 1996

Chandler, Damon. ***Exploring Posibilities. Part I: Designing an Explorer-style List View***, http://www.cbuilderzine.com/features/1999/10/cb199910dc_f/cb199910dc_f.asp, 1999

Chandler, Damon. ***Exploring Posibilities. Part II: Wrapping up our List View component***, http://www.cbuilderzine.com/features/2000/06/cb200006dc_f/cb200006dc_f.asp, 2000.

Chen, Raymond. ***Creating Shell Extensions with Shell Instance Objects***. Microsoft Corporatin. Febrero 2000.

Dermirhan, Mustafa. ***IEHelper - Internet Explorer Helper Class***. The Code Project. http://www.codeproject.com/shell/IEHelper.asp. Junio 2003.

Duke, Robert. ***Connect to a Running Instance of Internet Explorer***. Microsoft Knowledge Base Article 176792. http://support.microsoft.com/default.aspx?scid=kb;EN-US;176792. Marzo 2003.

Miller, Michael J. et al. ***El nuevo Windows: Grandes eXPectativas***. Trad. Verania de Parres. PC Magazine en español. Año 12. No. 11. Noviembre 2001.

Turner, Lori, ***Automating Microsoft Office 97 and Microsoft Office 2000***. Microsoft Corporation. 2000.

Vemula, Venu y Robert Walker. **Connecting to Running Instances of Internet Explorer.** Codeguru. http://www.codeguru.com/Cpp/I-N/ieprogram/article.php/c4403/. Enero 2001.