FRONT-END PARA EL CONTROL ORAL DE APLICACIONES EN ENTORNO WINDOWS

Jon Sanchez Iñaki Sainz Iker Luengo Eva Navas Inmaculada Hernáez
Departamento de Electrónica y Telecomunicaciones
Universidad País Vasco – Euskal Herriko Unibertsitatea
e-mail : ion,inaki,ikerl,eva,inma@bips.bi.ehu.es

***Abstract*- This paper presents the development of an oral interface to control e-mail applications as well as other commercial applications by means of voice, providing a user-friendly interface. It is fully configurable using plain text files, being able to manage any program with a graphic environment that works in Windows Operating System, using functions from the Windows API. Hidden Markov Models provide the speech recognition ability, using recursive training of triphone models. The text to speech system is already designed and integrated in a dynamic library. The application is focused on customers with some vision or movement handicap and it is designed to be used in Basque language.**

I. INTRODUCCIÓN

Con el progreso de las nuevas tecnologías y la introducción de sistemas interactivos, se ha incrementado enormemente la demanda de interfaces amigables para comunicarse con las máquinas. Dado que la voz es el medio de comunicación más natural para los humanos, no es de extrañar el intento de desarrollar nuevas formas de comunicación oral con las máquinas. Esto es especialmente importante a la hora de ofrecer servicios informáticos a personas con discapacidades motoras o visuales, que pueden conseguir comunicarse con los sistemas informáticos utilizando una interfaz oral. Para conseguir estos objetivos, es necesario proporcionar a las interfaces la capacidad de generar y reconocer el habla.

En el presente artículo se presenta un sistema que permite el control oral de aplicaciones con interfaz gráfica en entorno Windows y se analizan las técnicas que han posibilitado su diseño.

Para llevar a buen puerto dicho trabajo, se han tenido que afrontar dos dificultades principales. Por un lado se han debido superar las dificultades inherentes a los sistemas de reconocimiento que son debidas a la enorme variabilidad del habla (según el locutor, estado de ánimo) y a las condiciones del entorno (acústica, micrófono, ruidos...). Por otro lado ha sido necesario considerar de manera especial la integración de las distintas tecnologías y módulos empleados.

La primera parte del artículo analiza las tecnologías utilizadas para el desarrollo del control vocal, particularmente de sistemas de lectura de correo electrónico. En la segunda se analizan los sistemas desarrollados, tanto el Lector de Correo Electrónico Controlado por Voz (LCECV), como el front-end genérico para el control de aplicaciones. Por último, se recogen unas conclusiones.

II. TECNOLOGÍAS UTILIZADAS

Para diseñar un software que controle diversas aplicaciones mediante una interfaz vocal, es necesario integrar distintas tecnologías.

Fig 1. Esquema general de las tecnologías a utilizar.

Tal y como puede verse en la Fig. 1, las tecnologías utilizadas en este trabajo se pueden dividir en tres bloques principales: Reconocimiento del habla, Control de la Aplicación y Síntesis de Voz. A continuación se explica brevemente cada uno de ellos.

A. *Reconocimiento del Habla*

El sistema de reconocimiento que se ha utilizado en este trabajo puede caracterizarse según la clasificación clásica de este tipo de sistemas [1] en los siguientes aspectos:

- *Tamaño del vocabulario*: el sistema de reconocimiento utilizado es de vocabulario reducido, con menos de 20 palabras.
- *Dependencia del hablante*: se han realizado dos sistemas de reconocimiento diferentes, uno dependiente

del locutor, que únicamente es válido para la persona para la que se prepara el sistema y otro independiente del locutor, que puede ser utilizado por cualquier usuario.

- *Separación entre palabras*: el reconocimiento se realiza sobre palabras aisladas o comandos. Este tipo de sistemas representan el caso más sencillo desde el punto de vista de la tarea de reconocimiento, ya que existen silencios entre las palabras a reconocer.

Para la implementación del sistema de reconocimiento con las características descritas se han utilizado los Modelos Ocultos de Markov [2] (HMM, *Hidden Markov Models*). Estos HMM se pueden ver como una máquina de estados finitos en la que el siguiente estado depende únicamente del estado actual, y asociado a cada transición entre estados se produce un vector de observaciones.

En el proceso de aplicación de los HMM al reconocimiento del habla se distinguen dos fases diferenciadas: la fase de entrenamiento y la de reconocimiento.

La fase de entrenamiento de los modelos se realiza previamente a su utilización en el sistema de reconocimiento. En ella se generan los modelos HMM, uno para cada palabra a reconocer, partiendo de señales de voz del hablante que utilizará el sistema en el caso de que éste sea dependiente del locutor, o de señales de muchos locutores diferentes en el caso de los sistemas independientes de locutor. Para lograr modelos que representen de manera óptima el conjunto de señales de entrenamiento, se hace uso del algoritmo Baum-Welch [3], con 32 gaussianas y coeficientes cepstrales mel con valores de delta y de aceleración.

La fase de reconocimiento se realiza en el momento en que se utiliza el sistema. En esta fase, se hace uso del algoritmo de Viterbi que compara los parámetros de la señal de entrada con los modelos generados en la fase de entrenamiento, con el fin de obtener la secuencia de estados más probable, identificando de este modo el comando que ha sido pronunciado.

Para la implementación del módulo de reconocimiento basado en HMM, se ha utilizado la herramienta HTK en su versión 3.2.1 [4]. Se trata de un conjunto de aplicaciones gratuitas pero muy potentes entre las que destacan HRest para la fase de entrenamiento, y HVite para el reconocimiento.

Se han desarrollado dos tipos de entrenamiento, con el fin de desarrollar dos sistemas diferentes, uno dependiente y otro independiente del locutor:

- Para el primero de ellos, realizado con el objeto de obtener un modelo dependiente de locutor, se utiliza la utilidad de entrenamiento de modelos HRest con 20 grabaciones de voz por cada comando a reconocer. Para facilitar la tediosa tarea de que cada locutor tenga que grabar sus 20 señales de voz por comando, se ha desarrollado una aplicación que automatiza tanto la grabación (valiéndose de la grabadora de Windows) como la generación de modelos. Los modelos utilizados en este caso son modelos de palabra.

- En el segundo tipo de entrenamiento, se busca generar modelos robustos e independientes del locutor, haciendo uso de las múltiples señales de voz en euskera de la base de datos SpeechDat FDB1060 [5], grabada a través de la red telefónica fija y que contiene señales procedentes de 1060 locutores distintos. Los modelos utilizados para este tipo de entrenamiento son los basados en trifonemas. La elección de este tipo de modelos se debe a que recogen el efecto de coarticulación del lenguaje mejor que los modelos de monofonema, obteniendo una menor tasa de error [6]. Sin embargo, presentan un inconveniente ya que resulta complicado disponer entre las señales de entrenamiento, de todos los trifonemas posibles y en número suficiente [7] para que los modelos sean bien entrenados. El conjunto de herramientas HTK, proporciona facilidades para crear modelos de trifonemas a partir de los de monofonema y así se han utilizado en este trabajo. Otra cuestión que se ha tenido en cuenta es que la base de datos utilizada para crear los modelos es telefónica, mientras que las señales que después van a utilizarse llegarán al sistema por vía microfónica.

Una vez completado el entrenamiento, se utiliza la utilidad HVite para la fase de reconocimiento. Este programa se configura para que calcule el umbral de ruido a la entrada del micrófono de modo que se pueda detectar el inicio de los comandos. Esta comprobación realiza un modelado del canal microfónico de entrada, de manera que el sistema es capaz de obtener resultados a pesar de que los modelos de trifonemas han sido obtenidos a partir de una base de datos telefónica. Con esta configuración, los índices de acierto en el reconocimiento han alcanzado un 92% para el sistema dependiente de locutor, en pruebas realizadas en vivo. El sistema que utiliza la base de datos SpeechDat FDB1060, asimismo, consigue un índice de reconocimiento correcto del 91’8% en sistemas de reconocimiento de palabras aisladas con gramáticas reducidas y modelos basados en trifonemas.

Por último, el software de reconocimiento se configura para que emita un pitido al finalizar la captura del umbral de ruido. De esta forma la interfaz resulta más amigable al indicarle al usuario cuándo puede comenzar a dar órdenes.

B. *Síntesis de Voz*

La síntesis de voz se ha realizado mediante el conversor de texto en habla para el euskara AhoTTS [8]. Éste es un sintetizador modular, multilingüe y multiplataforma, que permite varios modos de funcionamiento. Para esta aplicación en particular se ha utilizado el algoritmo MBROLA [9] para el motor de síntesis, y la curva de entonación se ha calculado utilizando el modelo de Fujisaki [10].

Desde los programas que controlarán el sistema se accede a las capacidades de voz a través de una librería dinámica [11], que provee de funciones para el acceso de frases y ficheros de texto, así como para la configuración de ciertos aspectos de la dicción y el control de la reproducción de audio.

C. *Control de las aplicaciones*

Se han desarrollado dos modelos de control de aplicaciones. El primero (LCECV), se trata de un cliente desarrollado desde cero, con funciones básicas para la

utilización de mensajes de correo electrónico. Partiendo de él, se va controlando la utilidad de reconocimiento HVite, de manera que mediante comandos orales se puede controlar la propia ejecución del programa, que a su vez también gestiona las llamadas a la DLL de AhoTTS para tener una salida oral.

El segundo modelo de control de aplicaciones desarrollado es un front-end, es decir, un programa que es capaz de arrancar y controlar externamente aplicaciones con interfaz gráfico que funcionen bajo sistema operativo Windows, utilizando para ello la API que provee el propio sistema operativo.

III. SISTEMAS DESARROLLADOS

A. *Lector de Correo Electrónico Controlado por Voz (LCECV)*

Este cliente de correo ha sido desarrollado íntegramente en lenguaje Visual C++, y se puede utilizar para el envío/recepción hacia/desde servidores SMTP [12] y POP3 [13]. Incluye funciones básicas de manejo de correo electrónico, tales como lectura, respuesta, borrado, creación de nuevo de mensaje, navegación entre mensajes, etc.

Para tener acceso de una manera sencilla a los servicios de correo electrónico, se utiliza la MAPI (Messaging Application Programming Interface) de Windows. Tal y como se muestra en la Fig. 2, la MAPI realiza la función de puente entre nuestra aplicación y un cliente de correo compatible (p.e. Outlook Express) ofreciendo facilidades para el manejo tanto de los mensajes como de sus campos [14].

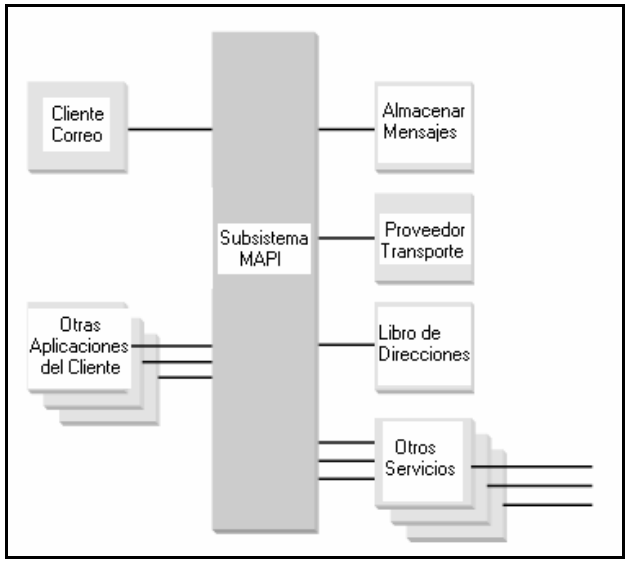

Fig. 2: Facilidades brindadas por MAPI a las aplicaciones del cliente.

Se parte del cliente de correo como punto central sobre el que integrar los dos módulos de interfaz oral, el de síntesis de voz y el de reconocimiento del habla. Para el primero, se utiliza linkado dinámico de la librería, de manera que sólo se carga un módulo en memoria en caso de que dos aplicaciones hagan uso de ella de forma simultánea. Para la integración del sistema de reconocimiento se sigue el siguiente esquema:

- Se crea el proceso HVite con una gramática inicial (conjunto de palabras posibles), así como un proceso hilo que recoja su salida.
- Se configura HVite para que genere un fichero con la palabra reconocida, el hilo lee el contenido y envía, dependiendo del comando detectado, un mensaje diferente al cliente de correo.
- Por último, el cliente de correo realiza la función pertinente, y, si fuera necesario cambiar de gramática, termina el proceso HVite anterior y crea uno nuevo. La utilización de varios conjuntos de gramáticas, en lugar de una sola, reduce de forma considerable la tasa de error en el reconocimiento.

La Fig. 3 muestra el aspecto del lector de correo electrónico controlado por voz.

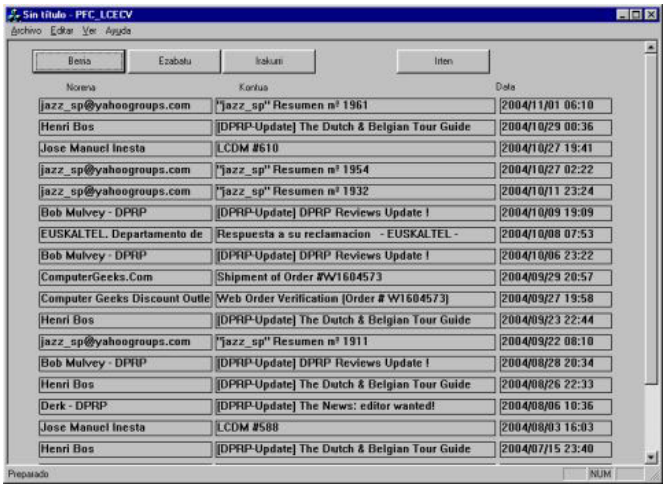

Fig. 3: Aspecto de la ventana principal del Lector de Correo Electrónico Controlado por Voz.

B. *FRONT-END*

Con los conocimientos adquiridos en el desarrollo del lector de correo electrónico, se plantea un proyecto más ambicioso, el cual, manteniendo intactos los módulos de reconocimiento y síntesis, busca controlar cualquier aplicación de Windows.

1. Características

Se trata de un sistema para desarrollar programas capaces de controlar mediante comandos vocales cualquier aplicación en el sistema operativo Windows. Para ello, se procesa un fichero de configuración donde se especifican las aplicaciones que se van a soportar. Por otro lado, cada una de esas aplicaciones dispone de un archivo de configuración propio con el siguiente contenido:

- Mensajes de bienvenida y cierre de la aplicación.
- Tareas a llevar a cabo por la aplicación, para cada uno de los comandos soportados. Estos comandos serán lanzados mediante órdenes vocales, por lo que el módulo de reconocimiento permite generar nuevas gramáticas acordes a las necesidades de control del sistema.

Para que el Front-End sea capaz de manejar la información de los ficheros de configuración se sigue el siguiente formato basado en los siguientes tipos de etiquetas:

- *Comandos*: se refiere principalmente a órdenes recibidas mediante la interfaz de reconocimiento del habla. El nombre del comando debe coincidir con la palabra reconocida por HVite.
- *Tareas*: cada una de las acciones que deben realizarse para completar el trabajo que realiza cada comando. Cada tarea se especifica mediante una etiqueta que refleja el tipo de operación, seguida por los argumentos (caso de que se requieran), uno por línea y dejando una línea en blanco entre tareas adyacentes.

Se pueden clasificar las etiquetas disponibles para la realización de tareas en los siguientes grupos:

- *Inicio y Finalización de Aplicaciones*: Se realiza con las funciones de la API de Windows CreateProcess y TerminateProcess.
- *Captura de Manejadores*: Se refiere a los números que identifican unívocamente ciertos elementos de una interfaz gráfica (Ventana, subventanas, menús, botones, cajas de texto...). Se trata de funciones de la API que necesitan una serie de argumentos (Título, clase, id, jerarquía...) para encontrar el manejador deseado.
- *Acción sobre Ventanas*: Se refiere al envío de mensajes a un elemento previamente identificado mediante su manejador. Hay diferentes tipos de mensajes y argumentos que permiten realizar funciones tales como Maximizar/minimizar/cerrar ventana, pulsar botón, escribir/coger texto de ventana, etc.
- *Manejo de Ficheros*: Permiten abrir un fichero y guardar todo o parte de su contenido en una variable interna del Front-End, asi como el borrado de archivos.
- *Lectura*: Invocan las capacidades de síntesis de la librería dinámica, permitiendo la lectura de frases pasadas como argumento.
- *Control del Sistema de Reconocimiento*: Permiten modificar la configuración y argumentos de la herramienta HVite. Se utilizan para expresar una nueva gramática y/o detener el proceso de HVite anterior.
- *Otras*: Funciones que simulan la pulsación de teclas (o combinación de ellas), salto condicional dentro del fichero de configuración, dormir el Front-End un tiempo determinado, etc.

2. Implementación

Utilizando la arquitectura ya descrita, se han desarrollado los ficheros de configuración necesarios para controlar los siguientes clientes de correo comerciales: Outlook, Outlook Express y Mozilla Mail.

Mediante menús leídos, el programa va guiando al usuario, permitiendo iniciar cualquiera de las tres aplicaciones de correo mencionadas. En todos los casos se permite iniciar mediante comandos de voz las funciones básicas de utilización de correo electrónico: lectura, respuesta, borrado, creación de nuevo de mensaje, y navegación entre mensajes, además de incluir la posibilidad de grabar una señal de voz e incluirla como archivo adjunto en un mensaje, de forma automática.

IV. CONCLUSIONES

Se ha desarrollado un sistema de control de aplicaciones mediante interfaz vocal para entorno Windows, utilizando para ello técnicas de reconocimiento automático del habla y de conversión de texto a voz. Se han conseguido unas tasas de reconocimiento de 92% utilizando modelos dependientes de locutor, y del 91% utilizando modelos independientes de locutor sobre la base de datos SpeechDat_EU.

Utilizando este sistema de control, se han desarrollado dos aplicaciones, una específica de correo electrónico (LCECV), y otra plataforma totalmente configurable para el control mediante interfaz oral de cualquier aplicación Windows con interfaz gráfica. El control de las aplicaciones externas se realiza, principalmente, mediante funciones de la API de Windows.

V. AGRADECIMIENTOS

Este proyecto ha sido parcialmente financiado por el programa Etortek del Departamento de Industria, Comercio y Turismo del Gobierno Vasco.

VI. REFERENCIAS

[1] L. R. Rabiner, and R.W. Schafer, “Digital Procesing of Speech Signal”. Prentice Hall.

[2] L.R. Rabiner, “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition”. Proc. IEEE 77(2), pp. 257-285, 1989.

[3] Y. Wu, A. Ganapathiraju and J. Picone, “Baum-Welch Re-estimation of Hidden Markov Model”. Department of Electrical and Computer Engineering, Mississippi State University. 1999.

[4] HTK Team website: http://htk.eng.cam.ac.uk/

[5] I. Hernaez, I. Luengo, E. Navas, M. Zubizarreta, I Gaminde and J. Sanchez, ”The Basque Speech_Dat (II) Database: A Description and First Test Recognition Results”. Eurospeech 2003

[6] M.-Y. Hwang, X. Huang and F. Alleva, “Predicting unseen triphones with senones”. Proceedings of ICASSP 93.

[7] S. Young and Others, *“The HTK Book”*.

[8] I. Hernaez, E. Navas, J.L. Murugarren, and B. Etxebarria; ”Description of the AhoTTS Conversion System for the Basque Language”. 4th ISCA Tutorial and Research Workshop on Speech Synthesis, 2001

[9] B. Etxebarria, I. Hernaez, I. Madariaga, E. Navas, J.C. Rodríguez and R. Gándara, “Improving quality in a speech synthesizer based on the MBROLA algorithm”. EUROSPEECH 99, Budapest

[10] E. Navas, I. Hernaez and J. Sanchez: “Basque Intonation Modelling for Text to Speech Conversion”. ICSLP 2002.

[11] S. Castro, I. Madariaga: “Diseño y Desarrollo de una librería dinámica para la síntesis y reproducción de textos en euskera”. Proyecto de Fin de Carrera. Euskal Herriko Unibertsitatea, 2004.

[12] J. Klensin, AT&T Laboratorios, “RFC 2821 - Simple Mail Transfer Protocol”. The Internet Working Group, 2001

[13] J. Myers, Carnegie Mellon, M. Rose, Dover Beach Consulting, Inc., “RFC 1939 - Post Office Protocol - Version 3”. The Internet Working Group, 1996

[14] D. Grundgeiger, “CDO and MAPI Programming”. O’Reilly , 2000