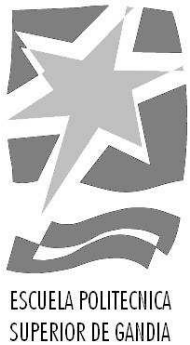
UNIVERSIDAD POLITÉCNICA DE VALENCIA

ESCUELA POLITÉCNICA SUPERIOR DE GANDÍA

Grado en Ingeniería de Sistemas de Telecomunicación

Sonido e Imagen



”Desarrollo de un sistema de reconocimiento del habla en Android”

TRABAJO FINAL DE GRADO

Autor:
Emilio Granell Romero

Directores:
Carlos D. Martínez Hinarejos
Vicent Tamarit Ballester

GANDIA, 2012

الانسان عدو ما يجهل:
علم لغة
تجتنب بلاهة حرب؛
انشر ثـقافة تكتسب شعباً لشعب

نعيم بوطانوس

"El ser humano es enemigo de aquello que ignora:
Enseña una lengua:
evitarás la estupidez de una guerra;
divulga una cultura:
conseguirás hacer popular a un pueblo entre las gentes de otro."

Naïm Boutanos

Resumen

En este trabajo se presenta un sistema de interpretación automática distribuido. Este sistema está compuesto por una aplicación cliente desarrollada para dispositivos móviles con sistema operativo Android, que interactúa con los usuarios, y un servidor encargado de las tareas automáticas de reconocimiento del habla y traducción. El dominio de interpretación está limitado a una tarea concreta basada en la interacción que un turista podría tener a su llegada a un hotel. Nuestro sistema es capaz de traducir frases habituales en una comunicación oral en la recepción de un hotel del castellano al inglés.

Abstract

In this work we present a distributed system for automatic interpretation. This system consists of a client application developed for mobile devices with Android operating system, which interacts with users, and a server dedicated to the automatic speech recognition and machine translation. The domain of interpretation is limited to a particular task based in the interaction that a tourist may have on arrival at a hotel. Our system is able to translate standard sentences in oral communication at the reception of a hotel from Spanish into English.

"A todos los que siempre creyeron en mí
más de lo que yo mismo nunca creí"

En especial me gustaría expresar mi agradecimiento:

A los directores de este proyecto Carlos y Vicent,
por ofrecerme su experiencia con total disponibilidad
durante la realización de este proyecto.
A Pati, por echarme una mano
durante la comprobación del sistema
y la revisión de este documento,
y por poner su voz en el vídeo de demostración.
A todos los voluntarios (familiares y amigos)
que participaron desinteresadamente
en la comprobación de *Hermes*.

Índice general

CAPÍTULO 1

INTRODUCCIÓN

"With realization of one's own potential and self-confidence in one's ability, one can build a better world."

His Holiness, The Dalai Lama

En el mundo actual las distancias se han reducido y muchas fronteras han desaparecido, como es el caso de la Unión Europea. Dadas estas condiciones, viajar es mucho más fácil y rápido que en el siglo pasado y, por ello, cada día son más necesarios los intérpretes de idiomas. Buscando intérpretes memorables en la historia de la humanidad, encontramos en la antigua mitología griega al dios Hermes, hijo de Zeus y la ninfa Maia. Hermes es el dios olímpico mensajero, de las fronteras y los viajeros que las cruzan, de los pastores y las vacadas, de los oradores y el ingenio, de los literatos y poetas, del atletismo, de los pesos y medidas, de los inventos y el comercio en general, de la astucia de los ladrones y los mentirosos. Hermes, como mensajero de los dioses viajó a través de las fronteras y actuó como intérprete para los seres humanos.

1.1. Interpretación automática

Según la Real Academia de la Lengua Española, un intérprete es *"una persona que explica a otras, en lengua que entienden, lo dicho en otra que les es desconocida"*. Partiendo de esta definición, un sistema de interpretación automático estará compuesto por un sistema de reconocimiento automático del habla (RAH) y por un sistema de traducción automática (TA), en el que:

RAH: Reconocimiento Automático del Habla. Es una parte de la Inteligencia Artificial que tiene como objetivo permitir la comunicación hablada entre seres humanos y computadoras, siendo capaz de procesar la señal de voz emitida por el ser humano y reconocer la información contenida en ésta, convirtiéndola en texto.

TA: Traducción Automática. Es un área de la lingüística computacional que investiga el uso de modelos computacionales para traducir de un lenguaje natural a otro.

1.1.1. EuTrans

Un trabajo destacable en interpretación automática es el sistema de traducción automática de voz a voz para telefonía EuTrans [1, 18].

EuTrans es capaz de traducir llamadas de una lengua a otra. El usuario recibe a través del teléfono una traducción sintética de la frase pronunciada. EuTrans opera en dominios restringidos donde alcanza altas productividades. En la actualidad está completamente operativo para dos

sentidos de traducción: italiano-inglés (tarea *FUB*) y español-inglés (tarea del *turista*). Estas tareas operativas atienden a toda aquella comunicación oral que pueda producirse en la centralita telefónica de un hotel, tanto para llamadas internas como externas, caso de la traducción italiano-inglés (tarea *FUB*), y a aquella que pueda producirse en el mostrador de un hotel, en el caso de la traducción español-inglés (tarea del *turista*).

EuTrans está basado en el reconocedor/traductor de habla continua ATROS, el cual utiliza modelos de estados finitos acústico-fonéticos, léxicos y sintáctico/traductores. La utilización de traductores de estados finitos como modelos sintácticos permite a ATROS obtener la traducción de manera síncrona con el reconocimiento. Todos estos modelos son entrenados automáticamente, lo que permite al sistema una adaptación sencilla a cualquier nueva tarea.

1.1.2. Estándar ETSI 201 108

El estándar ETSI 201 108 [4] define las características de un sistema de reconocimiento del habla distribuido completo, como son: el procesado de la señal, el algoritmo de extracción de características, los algoritmos de compresión, la transmisión y la calidad.

1.2. Dispositivos móviles

Los dispositivos móviles son aparatos de pequeño tamaño, que poseen algunas capacidades de procesamiento, con conexión permanente o intermitente a una red, con memoria limitada y diseñados específicamente para una función, aunque pueden llevar a cabo otras más generales.

Con un dispositivo móvil que disponga de capacidad de procesamiento, memoria y conexión a una red, se puede utilizar una arquitectura cliente-servidor para realizar tareas complejas que superen los límites de sus capacidades. La arquitectura cliente-servidor es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a un servidor y éste le da respuestas.

1.3. Motivación y objetivos

Los dispositivos móviles, teléfonos inteligentes y tabletas cada día cuentan con una mayor capacidad de procesamiento, cantidad de memoria y mejor conectividad. Actualmente podemos encontrar en el mercado una gran cantidad de modelos distintos con diferentes sistemas operativos como son: Android, iOS, Windows Phone, Symbian y BlackBerry.

El objetivo de este proyecto es desarrollar un sistema de interpretación automática que haga posible la comunicación entre dos personas que no tienen un idioma en común, utilizando para ello un dispositivo móvil. Basándonos en EuTrans, limitaremos el dominio de interpretación a la tarea del *turista*, es decir, nuestro sistema será capaz de traducir frases habituales en una comunicación oral en la recepción de un hotel del castellano al inglés.

Para el desarrollo de la arquitectura de este sistema de interpretación automática nos hemos basado en el estándar ETSI 201 108 [4], por lo que este sistema estará compuesto por un servidor, encargado de realizar el reconocimiento/traducción, y una aplicación cliente instalada en un dispositivo móvil con la que el usuario interactuará. En nuestro caso:

El servidor será virtual funcionando bajo Debian, en el que se instalará el sistema de reconocimiento automático del habla y del texto manuscrito iATROS [14] (improved ATROS) desarrollado por el grupo de investigación PRHLT (Pattern Recognition and Human Language Technologies).

La aplicación cliente será desarrollada en Android, y el dispositivo móvil sobre el que se instalará es una tableta Samsung Galaxy Tab, que cuenta con un procesador de 1 GHz, 16 GB de memoria interna, 512 MB de RAM y conectividad WiFi.

Hemos elegido esta tableta, además de por sus características físicas, por su sistema operativo Android 2.2 (que está basado en GNU/Linux), además de un servidor funcionando bajo Debian (por la libertad que ofrece el utilizar software libre).

Aunque actualmente se encuentran en el mercado soluciones interesantes de RAH y TA para dispositivos móviles, la ventaja que ofrece nuestro sistema frente a estas soluciones comerciales es la posibilidad de entrenar y adaptar los modelos acústicos y de traducción a tareas determinadas para obtener mejores resultados.

1.4. Organización del documento

En el Capítulo 2 se presenta una introducción a los fundamentos teóricos utilizados para la realización de este proyecto. El capítulo central de este trabajo es el Capítulo 3; en él se ofrece el diseño general de nuestro sistema de interpretación automática (*Hermes*), la arquitectura utilizada y cómo se han aplicado los conceptos abordados en el capítulo anterior para realizarlo. El proceso de comprobación del sistema implementado y los resultados obtenidos se encuentran en el Capítulo 4. En el último, el Capítulo 5, explicamos las conclusiones a las que llegamos en función de los resultados obtenidos durante las pruebas realizadas y los detalles que quedan abiertos al concluir este proyecto para trabajo futuro. Al final del documento se encuentra la bibliografía consultada para la realización de este trabajo, y además un conjunto de apéndices en los que se incluye información menos relevante; en el Apéndice A se explica cómo crear un servidor virtual utilizando VirtualBox, en el Apéndice B se define el procedimiento para instalar iATROS en el servidor virtual, en el Apéndice C se encuentra el conjunto de frases utilizadas para la comprobación del sistema, en el Apéndice D se presenta la encuesta de satisfacción utilizada para valorar el sistema y en el último, el Apéndice E, se encuentra la transcripción de las respuestas a la encuesta de satisfacción anterior de los usuarios que participaron en la comprobación del funcionamiento de *Hermes*.

CAPÍTULO 2

FUNDAMENTOS TEÓRICOS

"In principio erat Verbum, ..."

Biblia Sacra, Ioannem 1, 1

En este capítulo se describe la base teórica de este proyecto, que se fundamenta en los siguientes conceptos: el reconocimiento del habla, el procesado de la voz para el reconocimiento del habla, la comunicación en sistemas cliente-servidor, los dispositivos móviles y sus sistemas operativos.

2.1. Reconocimiento automático del habla

El lenguaje hablado es una capacidad natural de la especie humana y el habla supone la materialización de esa capacidad a través de una lengua. De todas las posibles formas de comunicación que existen es la más versátil y se manifiesta en todas las sociedades humanas, cosa que no ocurre con la escritura. Conforme aumenta la presencia de aparatos electrónicos en nuestra vida diaria, la interacción con ellos es cada vez más frecuente y sin duda facilitar la comunicación por voz es uno de los grandes retos de la actualidad. Se trata de conseguir que las máquinas más modernas puedan utilizarse a través de la forma de comunicación más antigua: el habla. Actualmente se trabaja en dos direcciones para conseguir que los computadores *"entiendan"* el lenguaje hablado:

RAH: Reconocimiento Automático del Habla. Se trata de encontrar la secuencia de palabras del lenguaje a la que corresponde el sonido que representa la señal vocal.

PLN: Procesamiento del Lenguaje Natural. Esta vertiente es más ambiciosa que la anterior, pues se trata de que a partir de una frase el sistema logre verificar su corrección sintáctica, léxica y semántica, con el fin de obtener su significado.

Para abordar estos problemas hay dos caminos posibles: los sistemas basados en el conocimiento, *"knowledge-based"*, y los sistemas basados en la estadística, *"data-based"*. Los primeros buscan, en RAH, que el reconocimiento se lleve a cabo a partir de reglas acústico-fonéticas, que se basan en características de la forma de la onda de entrada. Estas reglas deben ser descritas por un experto humano. Este método ha obtenido resultados muy pobres en RAH y algo mejores en PLN, debido, en gran medida, a la dificultad de los seres humanos de sistematizar su conocimiento para poder implementarlo en un computador.

Los sistemas basados en los datos, en cambio, han obtenido mejores resultados en ambas direcciones. Este método busca obtener *conocimiento* a partir de ciertas características que se

extraen de la onda acústica, mediante análisis estadísticos. Como ventaja respecto a los sistemas basados en el conocimiento, aquí las reglas no las define una persona, sino que son obtenidas mediante análisis estadísticos que pueden describirse fácilmente mediante algoritmos. Presentan, por otra parte, una desventaja: la necesidad de adquirir un corpus anotado lo suficientemente grande y representativo que pueda servir de referencia.

En este proyecto la dirección que se ha seguido ha sido la basada en técnicas estadísticas. Para ello es necesario definir tres modelos: el sintáctico o modelo del lenguaje, el léxico y el acústico.

Modelo sintáctico

El modelo sintáctico, también denominado modelo del lenguaje, define todas las posibles frases susceptibles de ser reconocidas por el sistema. En un nivel más general incluiría todas las posibles frases sintácticamente correctas del lenguaje, lo que generaría un modelo inabarcable para la capacidad de cómputo actual. Por ello el modelo de lenguaje se define generalmente para una tarea concreta. Este modelo, además de representar la sintaxis del lenguaje, puede incorporar la información semántica de las palabras. Para implementarlos se utilizan gramáticas probabilísticas [20] (o su equivalente como autómata), o N-gramas [15]. En el caso en el que se incluya semántica además de reconocimiento se suelen utilizar transductores de estados finitos (TEF).

Modelo léxico

El modelo léxico le indica al sistema la pronunciación de cada una de las palabras que componen el modelo sintáctico. El modelo léxico describe la pronunciación de cada palabra como una secuencia de símbolos que representan sonidos, definidos mediante los modelos acústicos. Suelen implementarse como autómatas finitos deterministas (AFD). Por ejemplo, la palabra *"Juan"* generaría el modelo de estados finitos de la Figura 2.1.

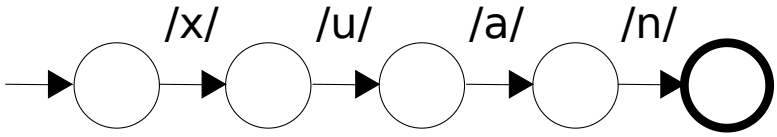


Figura 2.1: Autómata de estados finitos para la palabra *"Juan"* (Nótese que se trata de la transcripción fonética).

Modelo acústico

El modelo acústico es el modelo asociado a cada sonido. En el caso del habla suele asignarse uno a cada fonema del lenguaje, de forma que sea posible comparar las características de una secuencia acústica con los modelos descritos, a fin de obtener una frase del lenguaje como hipótesis de la frase pronunciada. La implementación más común de estos modelos es mediante Modelos Ocultos de Markov contínuos (HMM[1] por sus siglas en inglés).

Un HMM es un modelo de estados finitos probabilístico. La notación de un HMM suele ser $\lambda = (A, B, \pi)$, donde:

- A indica las probabilidades de transición entre estados.
- B contiene las probabilidades de emisión de cada símbolo para cada estado.
- π es la probabilidad, para cada estado, de ser inicial.

[1] Hidden Markov Model

Esta definición corresponde con la de un HMM genérico. A partir de aquí se pueden distinguir entre HMM discretos y continuos. La diferencia está en cómo se definan las probabilidades de emisión, si es mediante probabilidades discretas o mediante funciones continuas. Los HMM continuos habitualmente utilizan mixturas de distribuciones Gausianas como funciones de probabilidad. A pesar de que en la literatura se encuentran otras funciones de densidad de probabilidad, el uso de las Gausianas multivariadas es el recomendado, debido a que pueden usarse para aproximar cualquier función de densidad [16].

Generar un modelo acústico es una tarea muy laboriosa que un proyecto como este no puede abarcar; por esta razón se ha recurrido a un conjunto de modelos ya existente, entrenados a partir del corpus Albayzin [3]. Este corpus se obtuvo con la colaboración de varias universidades españolas, y en él han participado 100 locutores representantes de las variedades geográficas y sociales más importantes del español; se ha mantenido también una distribución igualitaria de ambos sexos. En total se grabaron 4097 frases, que suponen un total de, aproximadamente, tres horas y media de señal.

La tipología de los modelos es de tres estados, de izquierda a derecha, con bucles y sin saltos. Se utiliza una mixtura de 128 gaussianas (con matriz de covarianza diagonal) de 33 componentes y cada modelo es monofonema.

2.1.1. iATROS

iATROS [14] es el acrónimo de *"Improved Automatically Trainable Recognizer Of Speech"*. iATROS es un reconocedor automático tanto para habla como para texto manuscrito y está compuesto de dos módulos de preprocesado y extracción de características (para la señal de voz y para las imágenes de texto manuscrito) y un módulo principal de reconocimiento. Los módulos de preprocesado y extracción de características proporcionan vectores de características al módulo de reconocimiento, que utiliza HMM y modelos de lenguaje para realizar la búsqueda de las mejores hipótesis del reconocimiento.

2.1.2. Reconocimiento: Algoritmo de Viterbi

Para realizar el reconocimiento de una frase dicha, iATROS despliega los tres modelos anteriores, obteniendo un HMM sobre el que buscar el camino más probable que generaría la frase pronunciada. Cada transición del TEF que representa el modelo de lenguaje se despliega en el autómata de la palabra emitida en la transición, definido en el modelo léxico, y éste, a su vez, despliega cada una de sus transiciones en un HMM, definido en el modelo acústico. El resultado final se puede asumir (de manera simplificada) como un enorme HMM que representa al modelo de lenguaje.

El algoritmo de Viterbi [5] es la piedra angular del reconocimiento del habla y pieza fundamental en iATROS. Es el encargado de, dado un HMM y una observación, calcular la secuencia de estados del modelo que más probabilidad tiene de haber generado esa observación. En este caso concreto, a partir de la secuencia sonora adecuadamente preprocesada, se trata de buscar dentro del modelo de lenguaje desplegado el camino que más probabilidad tiene de haber generado esa onda.

El método computa, dado un HMM, la secuencia más probable de estados S que ha seguido el HMM para producir una observación O. Expresado formalmente, el problema a resolver consiste en: dado un HMM λ y una secuencia de observaciones O, se busca la secuencia de estados S^* de modo que:

$$S^* = \arg\max_S \Pr(O, S|\lambda) \tag{2.1}$$

La implementación más eficiente de Viterbi es como algoritmo iterativo, ya que se puede

expresar como un algoritmo de programación dinámica; sin embargo, es habitual formularlo de forma recursiva.

1. **Inicialización**: $\forall i \in estado$ Hacer
   $\delta_1(i) = \pi_i b_i(o_1)$
   $\psi_1(i) = 0$
2. **Recursión**: $\forall\, t/t = 2, ..., T \;\; \forall\, j/j \in estado$ Hacer
   $\delta_t(j) = \text{máx}_i[\delta_{t-1} a_{ij}] b_j(o_t)$
   $\psi_t(j) = \arg\max_i [\delta_{t-1}(i) a_{ij}]$
3. **Finalización**:
   $P^* = \text{máx}_i[\delta_T(s)]$
   $s_T^* = \arg\max_{s \in S_F} [\delta_T(S)]$
4. **Recuperación del camino**: $\forall t/t = T-1, ..., 1$ Hacer
   $s_t^* = \psi_{t+1}(s_{t+1}^*)$

T es la longitud de la observación O, P^* es la probabilidad del camino más probable, s_T^* es el estado final más probable y los s_t^* forman la ruta de estados más probable (la secuencia de estados que resulta útil para el reconocimiento). Durante la ejecución del algoritmo se considera que cada estado del HMM tiene una puntuación, que representa la probabilidad de transitar desde el estado inicial hasta ella. En realidad, puesto que el orden de magnitud de las probabilidades puede llegar a ser muy pequeño, para no perder información esta puntuación es el logaritmo cambiado de signo de la probabilidad. Esto no sólo evita el *underflow*, sino que además facilita los cálculos, pues las multiplicaciones de probabilidades son ahora sumas. En la formulación anterior estas probabilidades parciales se encuentran en δ_i.

El problema de encontrar $\Pr(O|\lambda)$ es NP-Duro. El coste del algoritmo de Viterbi en su implementación iterativa (por programación dinámica) es $O(NTB)$, siendo B el factor de ramificación efectiva del algoritmo de programación dinámica asociado, y N el número total de estados. Para reducir el espacio de búsqueda (y con ello el coste del algoritmo) y mejorar el resultado del reconocimiento, se añaden al reconocedor tres parámetros heurísticos: Beam, Grammar Scale Factor y Word Insertion Penalty.

A continuación se detallan estos parámetros:

Beam: Cada transición del modelo de lenguaje se despliega, como se ha visto, en un HMM sobre el que se aplica el algoritmo de Viterbi. Si la transición recibe una puntuación mayor que la mejor puntuación más este factor, se poda y dejan de explorarse los caminos que desde ella se pudieran generar. Con este factor puede controlarse el número de caminos que hay que explorar. Para valores bajos muchos caminos se podarán, lo que previsiblemente repercutirá en un mayor error de reconocimiento pero en un menor tiempo de cálculo, dado que cada iteración requiere actualizar la puntuación de menos caminos.

Grammar Scale Factor: Este parámetro modifica el peso de las probabilidades del modelo de lenguaje frente al modelo acústico. Es útil para equilibrar la influencia del modelo acústico y la gramática. Valores altos dan más importancia a la gramática, mientras que valores bajos hacen que el modelo acústico guíe el reconocimiento.

Word Insertion Penalty: Penaliza la inserción de nuevas palabras. Cuanto más alto es este factor, más se premia a las frases cortas en número de palabras. Este parámetro es necesario porque frases largas, o incluso palabras, pueden decodificarse como palabras más cortas, rompiendo el significado original. Por ejemplo palabras compuestas como *"automóvil"*, pueden ser reconocidas como *"auto"* y *"móvil"*. Ajustando este parámetro se controla este efecto en el reconocimiento.

Para adaptar el reconocedor a nuevas tareas basta con modificar el modelo de lenguaje (es decir, el autómata que representa el lenguaje de la tarea), así como el modelo léxico, incluyendo todas las palabras de la tarea. El modelo acústico, dado que es más general, puede reutilizarse en varias tareas, siempre que la lengua sea la misma.

2.2. Análisis de la señal de voz

En el reconocimiento del habla, la señal de voz, una vez digitalizada, se procesa para producir una nueva representación paramétrica de la voz. Esta representación es en forma de secuencia de vectores de características principales, que deben representar la información contenida en la envolvente del espectro.

2.2.1. Modelo de generación de voz

El sistema de generación de voz se puede modelar como un sistema compuesto por un filtro variable en el tiempo, un generador de ruido aleatorio y un generador de impulsos, como se puede ver en la Figura 2.2.

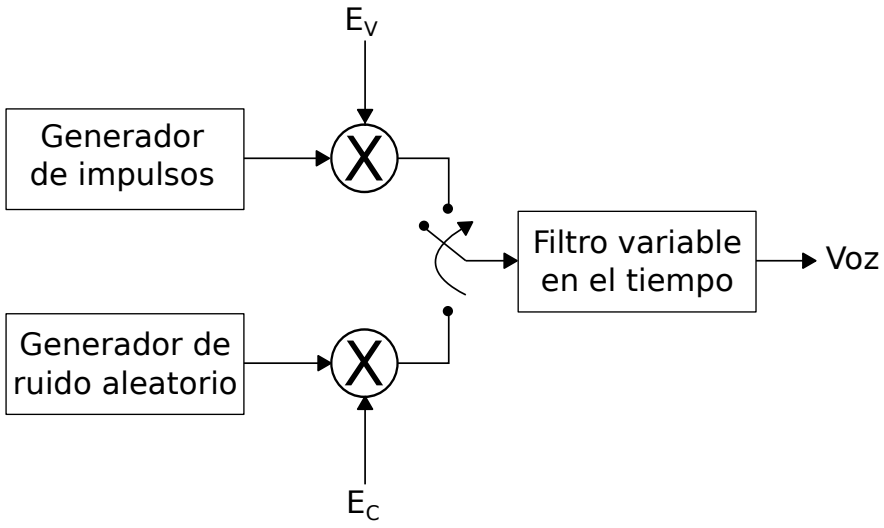


Figura 2.2: Modelo de producción de voz.

Este modelo tiene dos entradas, una para señales sonoras (vocales) E_V y otra para las señales no sonoras (consonantes) E_C. Para las señales sonoras la excitación es un tren de impulsos, mientras que para las señales no sonoras la excitación es ruido aleatorio. La combinación de estas dos señales modela el funcionamiento de la glotis.

A continuación, la señal de entrada pasa por un filtro que modela el funcionamiento del tracto vocal para obtener la señal de voz.

En este modelo es posible considerar que la señal de voz para fonemas sonoros se genera mediante la convolución mostrada en la Ecuación 2.2.

$$s(t) = e(t) * h(t) \tag{2.2}$$

En ella, la entrada al sistema es el tren de pulsos glóticos $e(t)$ y $h(t)$ es la respuesta al impulso del tracto vocal.

Cuando se pasa al dominio frecuencial mediante la transformada de Fourier, se observa que el espectro en frecuencia de la señal de voz, se corresponde con el producto de la transformada de Fourier de la señal de entrada por la respuesta en frecuencia del filtro.

$$S(\omega) = E(\omega)H(\omega) \tag{2.3}$$

2.2.2. Análisis Cepstral

Se define el cepstral de una señal $s(t)$ como la transformada inversa de Fourier del módulo del espectro en escala logarítmica de esa señal, es decir:

$$c(t) = F^{-1}[\log(S(\omega))] \tag{2.4}$$

Desarrollando el ceptral para $S(\omega)$ se obtiene:

$$c(t) = F^{-1}[\log|E(\omega)|] + F^{-1}[\log|H(\omega)|] \tag{2.5}$$

$$c(t) = c_e(t) + c_h(t) \tag{2.6}$$

De la Ecuación 2.6 se concluye que el cepstral de una señal es la suma del cepstral de la señal de entrada y del cepstral de la respuesta al impulso del filtro.

Generalmente, la señal del pulso glótico varía muy lentamente en relación con la respuesta en frecuencia del filtro que modela el funcionamiento del tracto vocal. Al realizar la primera transformación de Fourier se puede observar como $e(t)$ es modulada por $h(t)$. Por ello, al realizar la segunda transformación después de haber aplicado el logaritmo al módulo del espectro, se queda en las primeras muestras cepstrales la información relativa a la respuesta en frecuencia del filtro que modela el tracto vocal.

La mayor parte de la información del locutor se encuentra en las cuerdas vocales, mientras que la información de la palabra pronunciada se encuentra en las características del tracto vocal. Por ello, en el reconocimiento automático del habla, lo que interesa son las características del tracto vocal y se utilizan las componentes ceptrales bajas. En cambio, en el reconocimiento automático de locutores se utilizarán las componentes cepstrales altas, que contienen las características de la glotis.

Para el reconocimiento del habla es usual considerar de 10 a 12 coeficientes cepstrales (los 10 ó 12 primeros) obtenidos sobre una ventana temporal de unos 20 ó 30 milisegundos de duración. Para obtener los cepstrales de una señal de voz, se puede utilizar el algoritmo de extracción definido en el estándar ETSI 201 108 [4]. En la Figura 2.3 se muestra el diagrama de bloques de este algoritmo de extracción y a continuación se explica detalladamente cada uno de estos bloques.

Abreviaturas

ADC	*Conversión analógico - digital*
OC	*Compensación de Offset*
F	*División en tramos*
PE	*Pre - énfasis*
W	*Ventanas de análisis*
FFT	*Transformada rápida de Fourier*
MF	*Filtrado de la señal*
LOG	*Transformación logarítmica*
DCT	*Transformada de coseno discreta*

Figura 2.3: Diagrama de bloques del algoritmo de extracción de cepstrales.

Conversión analógico - digital

En primer lugar, se debe definir la frecuencia de muestreo de la digitalización de la señal de voz. Como el rango de inteligibilidad de la voz se encuentra en el espectro de frecuencias entre 300 Hz y 3400 Hz, se requiere una frecuencia de muestreo mínima de 8 KHz. Sin embargo, el estándar permite además frecuencias de muestreo de 11 KHz y 16 KHz.

Compensación de Offset

A continuación, se debe aplicar un filtro de muesca (*notch*) a las muestras de la señal de entrada S_{in} para retirar su valor de continua, produciendo una señal libre de continua S_{of}.

$$S_{of}(n) = S_{in}(n) - S_{in}(n-1) + 0{,}999 S_{of}(n-1) \tag{2.7}$$

División en tramos

La señal libre de continua S_{of} se divide en tramos superpuestos de N muestras. La diferencia entre los puntos de inicio entre tramos consecutivos es de M muestras y define el número de tramos por unidad de tiempo.

Los valores específicos de N y M dependen del valor de la frecuencia de muestreo elegida, como se puede apreciar en la Tabla 2.1. El tamaño de los tramos es 25 ms para 8 y 16 KHz y de 23.27 ms para 11 KHz.

Frecuencia de muestreo (KHz)	8	11	16
Longitud del tramo N (muestras)	200	256	400
Intervalo de cambio M (muestras)	80	110	160

Tabla 2.1: Valores de la longitud del tramo y del intervalo de cambio dependiendo de la frecuencia de muestreo.

Filtro de pre - énfasis

A continuación se aplica un filtro de pre-énfasis a cada tramo de la señal de entrada libre de continua:

$$s_{pe}(n) = s_{of}(n) - 0{,}97 s_{of}(n-1) \tag{2.8}$$

donde, s_{of} y s_{pe} son las señales de entrada y de salida del bloque de pre-énfasis respectivamente.

Ventanas de análisis

A la salida del bloque de pre-énfasis se aplica una ventana de Hamming de longitud N:

$$s_w(n) = \left\{ 0{,}54 - 0{,}46 \cos\left(\frac{2\pi(n-1)}{N-1} \right) \right\} s_{pe}(n), 1 \leq n \leq N \tag{2.9}$$

donde, N es la longitud del marco y s_{pe} y s_w son las señales de entrada y salida del bloque de aplicación de ventanas respectivamente.

Transformada rápida de Fourier (FFT)

Cada ventana de N muestras se completa con ceros hasta 256 muestras para 8 y 11 KHz y hasta 512 para 16 KHz de frecuencia de muestreo. A continuación, se aplica la transformada de Fourier de longitud $FFTL = 256$ ó $FFTL = 512$ para obtener el espectro de la señal:

$$bin_k = \left| \sum_{n=0}^{FFTL-1} s_w(n) e^{-jnk\frac{2\pi}{FFTL}} \right|, k = 0, ..., FFTL - 1 \tag{2.10}$$

donde, s_w es la señal de entrada en el bloque FFT, $FFTL$ es la longitud de la FFT (256 ó 512 muestras), y bin_k es el valor absoluto del vector complejo resultante.

Debido a la simetría del espectro, sólo los términos $bin_{0,...,FFTL/2}$ se entregan a la salida de este bloque.

Filtrado de la señal

Los componentes de baja frecuencia se ignoran, porque las frecuencias útiles se encuentran entre los 64 Hz y la mitad de la frecuencia de muestreo. Esta banda se encuentra dividida en 23 canales equidistantes en el dominio de la frecuencia de Mel.

La elección de la frecuencia de inicio en 64Hz, corresponde al caso donde toda la banda de frecuencia se encuentra dividida en 24 canales y el primer canal es descartado usando cualquiera de las tres posibles frecuencias de muestreo.

Las frecuencias centrales de cada canal en términos de los índices bin de la FFT ($cbin_i$ para el canal i) se calculan como sigue:

$$Mel\{x\} = 2595 \log_{10}\left(1 + \frac{x}{700}\right) \tag{2.11}$$

$$f_{c_i} = Mel^{-1}\left\{ Mel\left\{f_{start}\right\} + \frac{Mel\left\{f_s/2\right\} - Mel\left\{f_{start}\right\}}{23+1} i \right\}, i = 1, ..., 23 \tag{2.12}$$

$$cbin_i = round\left\{ \frac{f_{c_i}}{f_s} FFTL \right\} \tag{2.13}$$

donde, $round(.)$ significa redondear al siguiente entero.

La salida de los filtros de Mel es la suma ponderada de los valores de la magnitud del espectro obtenida con la transformada rápida de Fourier (bin_i) de cada banda. La aplicación de las ventanas triangulares medio superpuestas es como sigue:

$$fbank_k = \sum_{i=cbin_{k-1}}^{cbin_k} \frac{i - cbin_{k-1} + 1}{cbin_k - cbin_{k-1} + 1} bin_i + \sum_{i=cbin_k+1}^{cbin_{k+1}} \left(1 - \frac{i - cbin_k}{cbin_{k+1} - cbin_k + 1}\right) bin_i \tag{2.14}$$

donde, $k = 1, ..., 23$, $cbin_0$ y $cbin_{24}$ denotan los índices bin de la FFT que corresponden a la frecuencia de inicio y a la frecuencia mitad de la de muestreo respectivamente.

$$cbin_0 = round\left\{ \frac{f_{start}}{f_s} FFTL \right\} \tag{2.15}$$

$$cbin_{24} = round\left\{ \frac{f_s/2}{f_s} FFTL \right\} = FFTL/2 \tag{2.16}$$

Transformación logarítmica

A la salida de los filtros de Mel se aplica una función logarítmica a cada canal.

$$f_i = \ln(fbank_i), i = 1, ..., 23 \tag{2.17}$$

Se establece un valor mínimo para la transformación logarítmica, de modo que la salida de este bloque no puede ser menor de -50.

Transformada de coseno discreta

Finalmente, se obtienen los 13 coeficientes cepstrales aplicando una transformación de coseno discreta a la salida de la función logarítmica.

$$C_i = \sum_{j=1}^{23} f_j \cos\left(\frac{\pi i}{23}(j - 0{,}5)\right), 0 \leq i \leq 12 \tag{2.18}$$

2.3. Comunicación en sistemas distribuidos

Un sistema distribuido se define como una colección de computadoras autónomas separadas físicamente y conectadas entre sí por una red de comunicación. En ella cada máquina posee sus propios componentes de hardware y el software adecuado para que el sistema sea visto por los usuarios como un único sistema de computación.

2.3.1. Arquitectura cliente-servidor

La arquitectura cliente-servidor es un modelo de sistema distribuido en el que el trabajo se reparte entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, que le da respuesta, como podemos ver en la representación de la Figura 2.4. La mayoría de las aplicaciones en Internet utilizan la arquitectura cliente-servidor, como por ejemplo los servidores web, los servidores de archivos, los servidores del correo, etc.

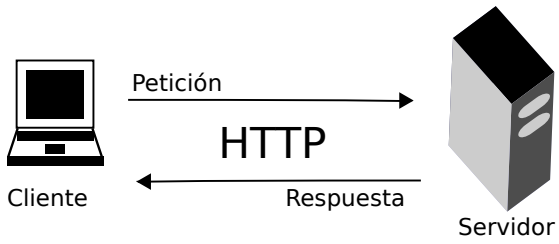


Figura 2.4: Arquitectura cliente-servidor.

Cliente: Es quien inicia las peticiones y espera las respuestas del servidor. Por lo general, puede conectarse a varios servidores a la vez y normalmente interactúa directamente con los usuarios finales mediante una interfaz gráfica de usuario.

Servidor: Al iniciarse espera a que le lleguen las peticiones de los clientes. Tras la recepción de una solicitud, la procesa y luego envía la respuesta al cliente. Por lo general, acepta conexiones

desde un gran número de clientes y no es frecuente que interactúe directamente con los usuarios finales.

Comunicación entre clientes y servidores (Sockets)

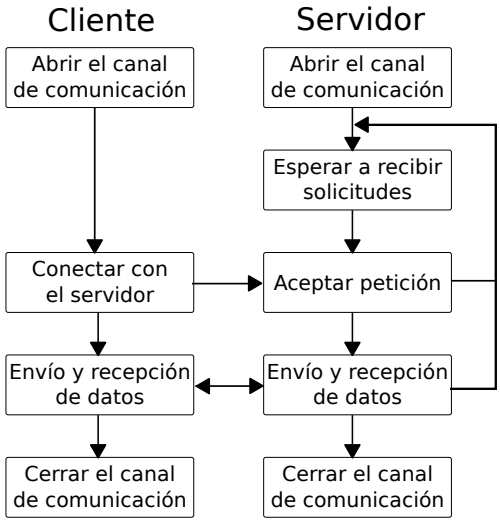


Figura 2.5: Modelo de comunicación cliente-servidor.

A los puntos finales de una comunicación entre dos sistemas que intercambian información a través de una red de comunicaciones se les llama sockets [12]. Un socket queda definido por la dirección IP del dispositivo en el que se encuentra y un número de puerto de 16 bits. Una conexión está determinada por un par de sockets, el del cliente y el del servidor. En la Figura 2.5 se muestra un modelo típico de comunicación por sockets en una arquitectura cliente-servidor.

Hay dos tipos de sockets: socket stream y socket datagram.

Socket stream: Utilizan el protocolo TCP (Transmission Control Protocol), por lo que también son conocidos como sockets orientados a conexión. El utilizar el protocolo TCP implica que antes de enviar la información se debe establecer la conexión entre los dos sockets, y ofrece la ventaja de que incorpora la corrección de errores de forma transparente al programador.

Socket datagram: Utilizan el protocolo UDP (User Datagram Protocol), por lo que también son conocidos como sockets sin conexión. Al utilizar dicho protocolo se puede enviar la información sin necesidad de haber establecido la comunicación entre los sockets. La ventaja de este tipo de sockets es que produce una menor sobrecarga sobre la información transmitida, a cambio de no tener incorporada la corrección de errores. Esto lo hace más rápido y por tanto más interesante en una red en la que se pierdan pocos paquetes y para aplicaciones de tiempo real.

2.4. Dispositivos móviles

Los dispositivos móviles son aparatos de pequeño tamaño, con algunas capacidades de procesamiento, con conexión permanente o intermitente a una red, con memoria limitada y diseñados específicamente para una función, aunque pueden llevar a cabo otras más generales [21].

2.4.1. Clasificación de los dispositivos móviles

Dado el variado número de niveles de funcionalidad asociado con dispositivos móviles, en el 2005, T38 y DuPont Global Mobility Innovation Team propusieron los siguientes estándares para la definición de dispositivos móviles:

Dispositivo móvil de datos limitado: Dispositivos que tienen una pantalla pequeña, principalmente basada en pantalla de tipo texto con servicios de datos generalmente limitados a SMS y acceso WAP.

Dispositivo móvil de datos básico: Dispositivos que tienen una pantalla de mediano tamaño, (entre 120 x 120 y 240 x 240 pixels), menú o navegación basada en iconos por medio de una rueda o cursor, y que ofrecen acceso a e-mails, libreta de direcciones, SMS, y un navegador web básico.

Dispositivo móvil de datos mejorado: Dispositivos que tienen pantallas de medianas a grandes (por encima de los 240 x 120 pixels), navegación de tipo *stylus*, y que ofrecen las mismas características que el *"dispositivo móvil de datos básico"*, más aplicaciones nativas y corporativas usuales, en versión móvil. Este tipo de dispositivos utilizan un sistema operativo como iOS, Windows Phone, Symbian, Android, etc.

2.4.2. Sistemas operativos móviles

Un sistema operativo móvil es un programa que gestiona los procesos básicos de un dispositivo móvil al igual que las computadoras utilizan Windows o Linux entre otros. Sin embargo, los sistemas operativos móviles son bastante más simples y están más orientados a la conectividad inalámbrica, los formatos multimedia para móviles y las diferentes maneras de introducir información en ellos.

En el estudio realizado por Gartner Inc. [7] se muestra una comparativa entre los dispositivos móviles vendidos a nivel mundial en el segundo trimestre del 2011 respecto al mismo periodo del 2010. Presenta los resultados por empresas fabricantes (Figura 2.6) y por sistemas operativos (Figura 2.7). De los resultados de este estudio, podemos concluir que los usuarios prefieren dispositivos móviles con los sistemas operativos Android e iOS, mientras que los dispositivos con Symbian y BlackBerry OS (Research In Motion) cada día son menos apreciados. Observando la gráfica de fabricantes, se confirma la afirmación anterior: Nokia está perdiendo cuota de mercado a pasos agigantados por haber apostado por un sistema operativo móvil que no acaba de gustar a los usuarios, como es Windows Phone de Microsoft, que al igual que Symbian (su anterior sistema operativo) y el Blackberry OS de Research In Motion, están cediendo terreno a Android e iOS.

BlackBerry OS

BlackBerry OS es el sistema operativo utilizado en una línea de teléfonos inteligentes denominados BlackBerry, que integran el servicio de correo electrónico móvil. BlackBerry fue desarrollado por la compañía canadiense Research In Motion (RIM). Aunque incluye las aplicaciones típicas de un teléfono inteligente (libreta de direcciones, calendario, listas de tareas, etc.), es fundamentalmente conocido por su capacidad para enviar y recibir correo electrónico de Internet accediendo a las redes móviles de las compañías de telecomunicaciones que brindan este servicio.

Symbian

Symbian es un sistema operativo móvil que fue producto de la alianza de varias empresas de telefonía móvil, entre las que se encuentran Nokia, Sony Ericsson, Psion, Samsung, Siemens,

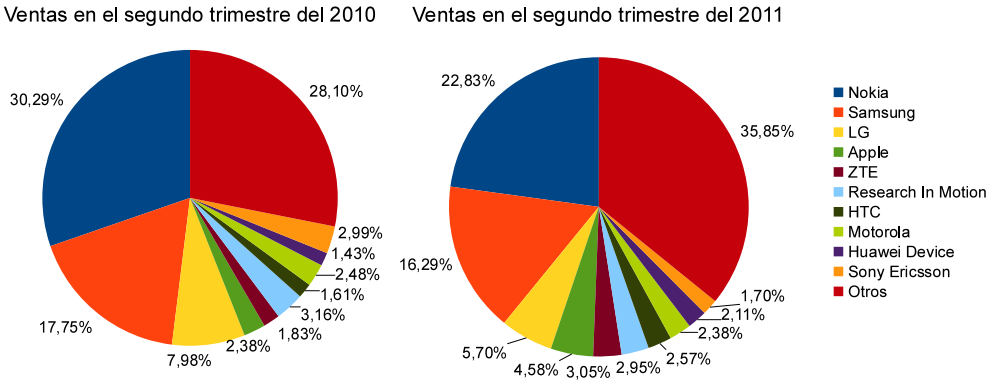


Figura 2.6: Comparativa mundial de ventas de dispositivos móviles por empresas fabricantes.

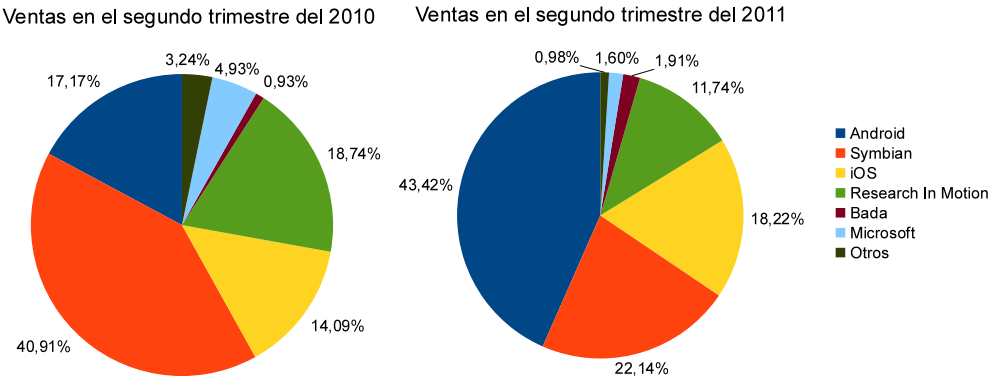


Figura 2.7: Comparativa mundial de ventas de dispositivos móviles por sistema operativo.

Arima, Benq, Fujitsu, Lenovo, LG, Motorola, Mitsubishi Electric, Panasonic, Sharp, etc. Sus orígenes provienen de su antepasado EPOC32, utilizado en PDA's y Handhelds de Psion. El objetivo de esta alianza fue crear un sistema operativo para dispositivos móviles que pudiera competir contra el de Palm y el Windows Phone de Microsoft, y actualmente contra Android de Google Inc., iOS de Apple Inc. y BlackBerry OS.

En 2009 se estableció que la fundación Symbian debía liberar la plataforma Symbian como un proyecto de software libre y gratuito. Sin embargo, en noviembre del 2010 dicha fundación abandonó sus actividades operacionales como resultado de la situación económica global y las condiciones del mercado, pasando la plataforma Symbian a estar bajo la supervisión de Nokia. El código fuente de Symbian se encuentra disponible en la página web *symbian.nokia.com*, donde Nokia aclara que no se trata de una plataforma de software libre bajo la premisa: *Not Open Source, just Open for Business*.

Nokia, que utilizó Symbian en la mayoría de sus dispositivos móviles, anunció en febrero del 2011 que su plan estratégico para competir contra iOS y Android sería utilizar Windows Phone como sistema operativo en sus nuevos dispositivos móviles.

Windows Phone

Windows Phone 7 es un sistema operativo móvil desarrollado por Microsoft, como sucesor de la plataforma Windows Mobile. Está pensado para el mercado de consumo generalista en lugar

del mercado empresarial, por lo que carece de muchas de las funcionalidades que proporciona la versión anterior. Microsoft ha decidido no hacer compatible Windows Phone 7 con Windows Mobile 6, por lo que las aplicaciones existentes no funcionan en Windows Phone 7, haciendo necesario desarrollar nuevas aplicaciones. Con Windows Phone 7, Microsoft ofrece una nueva interfaz de usuario, integra varios servicios en el sistema operativo y planea un estricto control del hardware que implementará el sistema operativo, evitando la fragmentación con la evolución del sistema. Microsoft realizó una importante actualización a finales del 2011, que incluyó Internet Explorer 9 y algunas mejoras que, según Microsoft, hacen que Windows Phone 7 sea realmente competitivo contra los sistemas operativos iOS de Apple y Android de Google. Por el momento, Windows Phone 7 continúa perdiendo cuota de mercado, como se puede apreciar en la Figura 2.7, frente a competidores como Android o iOS.

iOS

iOS (anteriormente denominado iPhone OS) es un sistema operativo móvil de Apple desarrollado originalmente para el iPhone, siendo después utilizado en todos los dispositivos iPhone, iPod Touch e iPad. Es un derivado de Mac OS X, que a su vez está basado en Darwin BSD. El iOS tiene 4 capas de abstracción: la capa del *Núcleo del sistema operativo*, la capa de *Servicios principales*, la capa de *Medios de comunicación* y la capa de *Cocoa Touch*. Todo el sistema se encuentra en la partición */root* del dispositivo y ocupa poco menos de 500 megabytes.

Android

Android es un sistema operativo basado en el núcleo Linux diseñado originalmente para teléfonos inteligentes. Posteriormente se expandió su desarrollo para soportar otros dispositivos tales como tabletas, reproductores MP3, netbook, PC, televisores, lectores de libros electrónicos e incluso se han mostrado microondas y lavadoras funcionando con Android en la Feria Internacional de Electrónica de Consumo de Las Vegas en enero del 2010.

Fue desarrollado inicialmente por Android Inc., una firma comprada por Google en 2005. El anuncio del sistema Android se realizó el 5 de noviembre de 2007 junto con la creación de la Open Handset Alliance, un consorcio de 84 compañías de hardware, software y telecomunicaciones, que han hecho de Android su principal producto. Este conglomerado de compañías se dedica al desarrollo de estándares abiertos para dispositivos móviles, con el objetivo de acelerar la innovación en estos dispositivos y ofrecer a los usuarios una experiencia móvil más rica, barata y mejor.

En el 2008 Google liberó la mayoría del código de Android bajo la licencia Apache, una licencia libre y de código abierto. Android tiene la mayor cuota de mercado desde enero del 2011, con más de un tercio del cómputo global de ventas. En los dos últimos años ha pasado a ser el sistema operativo móvil más utilizado.

CAPÍTULO 3

HERMES

"Canta, Musa, a Hermes, hijo de Zeus y Maya, que tutela Cilene y Arcadia, pródiga en rebaños, raudo mensajero de los inmortales, al que parió Maya, la Ninfa de hermosos bucles, tras haberse unido en amor a Zeus ..."

Homero - Himno a Hermes (Himno IV del volumen Himnos homéricos)

En honor a Hermes, antiguo dios griego mensajero e intérprete, dios de las fronteras y de los viajeros que las cruzan, le dimos su nombre a nuestro sistema de interpretación automática.

En este capítulo explicaremos todos los detalles del sistema implementado. Comenzaremos relatando una modificación respecto al estándar ETSI 201 108 [4], así como nuestras razones para realizar este cambio. A continuación, explicaremos el sistema final describiendo su diseño y arquitectura, la aplicación cliente y el servidor.

3.1. Modificación respecto al estándar ETSI 201 108

Como vimos en el Capítulo 1, Hermes está basado en EuTrans y en el estándar ETSI 201 108. Según éste, el análisis de la voz para la extracción de cepstrales se debe realizar en el dispositivo móvil. En el análisis de la voz para extraer los cepstrales el punto de mayor complejidad corresponde a la realización de la transformada rápida de Fourier (FFT). Este proyecto se implementó en una primera versión de este modo, probando dos librerías diferentes para calcular la FFT:

- La librería libGDX [22] es una librería libre para el desarrollo de juegos multi-plataforma en Java, con algunas partes del código JNI para mejorar el rendimiento. El cálculo de la FFT en nuestro sistema utilizando esta librería resultó muy fácil; sin embargo, en el reconocimiento se produjo una alta tasa de error.
- La librería FFTW3 [6] es conocida por ser la implementación libre más rápida del algoritmo de cálculo de la FFT. Esta librería es la utilizada por iATROS y para poder utilizarla en nuestra aplicación debimos compilarla como código nativo mediante el NDK de Android [11]. Las librerías nativas precisan de un contenedor para poder ser utilizadas desde Java. Para esta librería encontramos el contenedor jfftw3 [13] que le proporciona acceso JNI, desarrollado por Katsutoshi Itoyama, profesor adjunto de la Facultad de Informática de la Universidad de Kyoto.

Después de haber implementado la extracción de cepstrales en la aplicación cliente, las razones por las que decidimos modificar el sistema y realizar el procesado de extracción de cepstrales en el servidor son las siguientes:

- No es posible realizar la extracción de cepstrales si el dispositivo móvil no dispone de unas características mínimas.
- El tiempo necesario para la extracción de los cepstrales depende de las características del dispositivo móvil.
- El fichero de cepstrales en ocasiones llega a ser mayor que el fichero de audio original.

Con lo que en el sistema finalmente implementado las ventajas son:

- El sistema funciona en todos los dispositivos móviles con Android 2.1 o versiones superiores de Android.
- La sensación de retardo en el reconocimiento es la misma para todos los usuarios, independientemente del dispositivo móvil.
- El fichero de audio casi siempre ocupa menos espacio que el de cepstrales con lo que se reduce el tiempo de envío.
- La aplicación cliente resulta más sencilla.

3.2. Arquitectura y diseño

El sistema de interpretación automática Hermes es un sistema distribuido con una arquitectura cliente-servidor (Figura 3.1). Hermes está compuesto por una aplicación cliente que corre en un dispositivo móvil que se comunica mediante sockets con un servidor, de modo que el dispositivo móvil sólo se encarga de la interacción con el usuario. Dicho dispositivo digitaliza la voz del usuario y le muestra la respuesta del sistema, mientras que en el servidor se realizan las tareas que requieren mayor nivel de procesado y memoria: el reconocimiento del habla y la traducción.

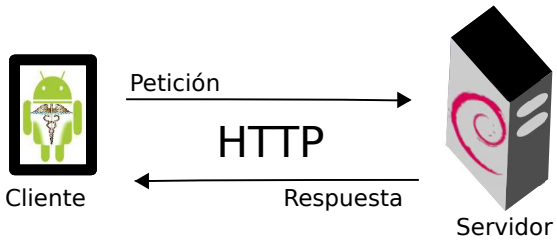


Figura 3.1: Arquitectura del sistema de interpretación automático Hermes.

En la Figura 3.2 se muestra el funcionamiento de nuestro sistema de interpretación automático. La aplicación cliente en el dispositivo móvil se encarga de digitalizar la voz del usuario en un fichero, de enviarlo al servidor y de mostrar la respuesta recibida desde el servidor al usuario. Por otro lado, el servidor se mantiene a la espera de que le llegue una petición de la aplicación cliente. En el momento que recibe una nueva petición, la acepta y el servidor recibe el fichero con la voz digitalizada obtenida desde el dispositivo móvil. A continuación, lanza iATROS dando como entrada el fichero recibido y retransmite la respuesta de iATROS al dispositivo móvil a través del socket.

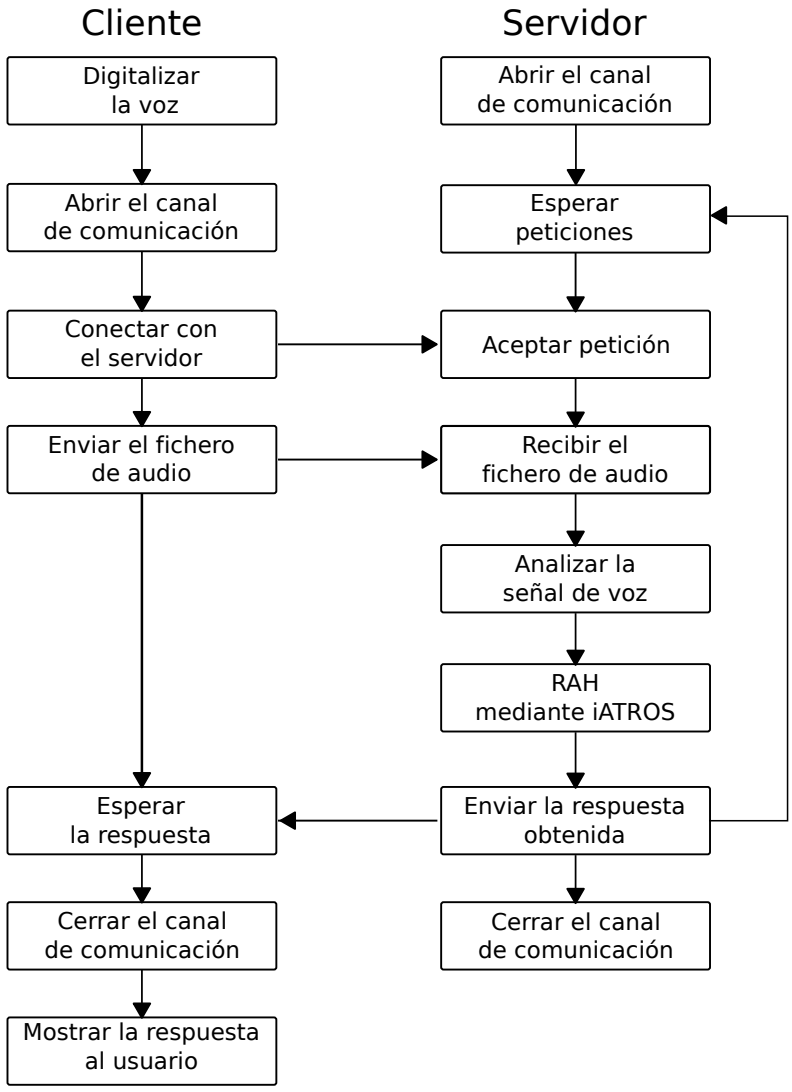


Figura 3.2: Modelo de funcionamiento de Hermes.

3.3. Cliente

La aplicación cliente se ha desarrollado en Android [9, 8, 10] por ser una plataforma libre y por el gran éxito que está teniendo en estos momentos. El dispositivo móvil elegido para el desarrollo de este proyecto ha sido una tableta *Samsung Galaxy Tab*, que cuenta con un procesador de 1 GHz, 16 GB de memoria interna, 512 MB de RAM y Android 2.2. Sin embargo, la aplicación se ha desarrollado utilizando el SDK con la API 7 de Android. Esto significa que esta aplicación puede instalarse en dispositivos móviles con Android a partir de la versión 2.1.

Las aplicaciones en Android están compuestas por cualquiera de los siguientes componentes:

Actividad: Representa la capa de presentación de toda aplicación Android, por ejemplo, una pantalla que el usuario ve. Una aplicación para Android puede tener varias actividades y se puede cambiar entre ellas en tiempo de ejecución de la aplicación.

Servicios: Realizan tareas en segundo plano, sin ofrecer una interfaz gráfica de usuario. Se puede notificar al usuario a través de la función de notificación de Android.

Proveedor de Contenido: Proporciona datos a las aplicaciones; a través de un proveedor de contenido su aplicación puede compartir datos con otras. Android contiene una base de datos SQLite, que puede servir como proveedor de contenidos.

Receptor de Mensajes: Recibe los mensajes del sistema y las solicitudes implícitas; se puede utilizar para responder a condiciones cambiantes en el sistema. Una aplicación puede registrarse como receptor de la difusión de ciertos eventos y se puede iniciar a sí misma si se producen tales acontecimientos.

Nuestra aplicación cliente está compuesta de tres actividades: la actividad principal, la actividad de digitalización de la voz y la actividad de comunicación con el servidor. En la Figura 3.3 se puede ver el grafo de funcionamiento de esta aplicación. Al iniciarse, el usuario se encuentra con la actividad principal, desde la que se pasa a la actividad de digitalización de la voz. Una vez terminada la digitalización se activa la actividad de comunicación con el servidor. Desde la actividad de comunicación se envía el fichero de audio al servidor y se recibe la respuesta de iATROS que se muestra al usuario. Terminado el proceso, la aplicación vuelve a la actividad principal a la espera de una nueva consulta por parte del usuario.

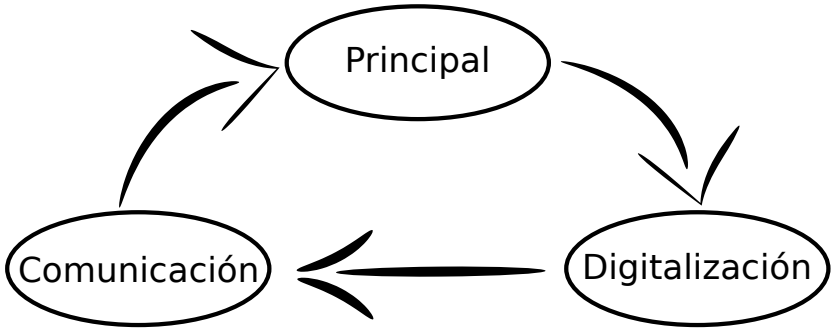


Figura 3.3: Funcionamiento de la aplicación cliente.

A continuación, veremos en detalle cada una de estas tres actividades.

3.3.1. Actividad 1: Principal

Cuando el usuario inicia la aplicación en su dispositivo móvil con Android, se encuentra con la pantalla de inicio mostrada en la Figura 3.4(a). En esta pantalla el usuario encuentra un botón verde con un micrófono en su interior, un botón gris con un altavoz y los textos *"Preparado"* y *"Pulse para comenzar a reconocer"*. El usuario debe pulsar el botón verde con el micrófono justo antes de empezar a decir lo que desea traducir. En la parte baja se encuentran dos cuadros de texto: en el primero se mostrará el texto reconocido por el servidor y en el segundo su traducción. Al ser pulsados, su contenido será leído en castellano e inglés, respectivamente, mediante el sintetizador de voz de Google. El botón gris con el altavoz permite al usuario escuchar el audio capturado por el dispositivo.

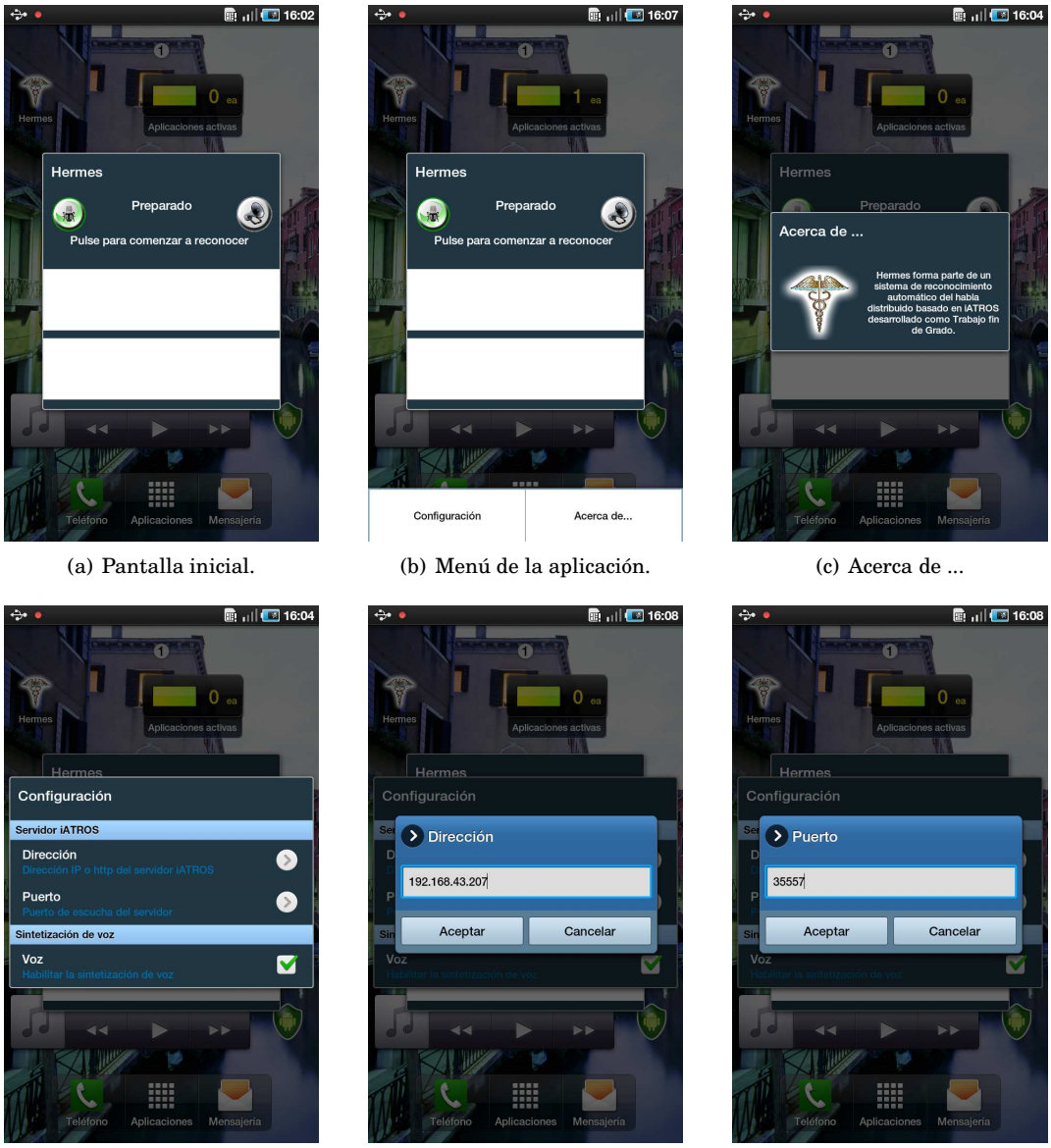
En esta pantalla inicial el usuario tiene a su disposición la configuración de la aplicación en el botón *"Configuración"* y la información básica de este proyecto en el botón *"Acerca de ..."* a los que se accede al pulsar la tecla *"Menú"*, tal y como se puede ver en la Figura 3.4(b).

Al pulsar sobre el botón *"Acerca de ..."* se muestra una breve descripción del proyecto (Figura 3.4(c)), el nombre del autor, de los directores y el lugar y año de realización.

Al pulsar sobre el botón *"Configuración"*, se entra en el menú de configuración que contiene las opciones que los usuarios pueden modificar (Figura 3.4(d)). Éstas son los dos parámetros que hacen referencia a la comunicación con el servidor por sockets (la dirección y el puerto del socket del servidor) y la posibilidad de desactivar la sintetización de voz.

La dirección del servidor (Figura 3.4(e)) se puede introducir como una dirección IP (como podría ser *"192.168.43.207"*) o como una dirección HTTP (como podría ser *"http://iatros.upv.com"*).

El puerto TCP utilizado para escuchar en el servidor (Figura 3.4(f)) debe ser un número comprendido entre 1024 y 49151 (como podría ser *"35557"*), que son los números de puerto libres para el uso en nuestras aplicaciones.



(a) Pantalla inicial. (b) Menú de la aplicación. (c) Acerca de ...

(d) Menú de configuración. (e) Dirección IP del servidor iATROS. (f) Puerto de escucha del servidor iATROS.

Figura 3.4: Actividad principal.

En cuanto el usuario pulse el botón verde con el micrófono, la aplicación pasará a la siguiente actividad: la digitalización de la voz.

3.3.2. Actividad 2: Digitalización de la voz

Para poder analizar la voz en el servidor, como se describe en el estándar ETSI ES 201 108 [4], en esta actividad se digitaliza el sonido que llega al micrófono del dispositivo móvil con una frecuencia de muestreo de $16KHz$, y se almacena en un fichero de audio sin compresión.

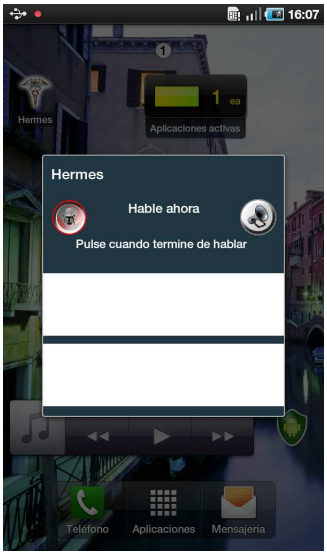


Figura 3.5: Actividad de digitalización de la voz.

Mientras se está digitalizando la voz, el usuario se encuentra con la pantalla de la Figura 3.5 en la que el botón verde al ser pulsado se ha vuelto rojo y el texto le recuerda que debe *"Hablar ahora"* y que debe *"Pulsar cuando termine de hablar"*.

En cuanto el usuario termine de hablar y pulse el botón rojo la aplicación pasará a la siguiente actividad: la comunicación con el servidor.

3.3.3. Actividad 3: Comunicación con el servidor

En esta actividad se establece la comunicación mediante sockets con el servidor. Al servidor se le envía el fichero de sonido obtenido en la actividad anterior y, mientras iATROS está realizando el reconocimiento en el servidor, la aplicación cliente espera a recibir la respuesta, que se muestra al usuario en cuanto se recibe.

Mientras el dispositivo móvil trata de establecer la comunicación mediante sockets con el servidor, aparece una animación con el texto *"Conectando con el servidor iATROS. Por favor espere ..."*, como se puede ver en la Figura 3.6.

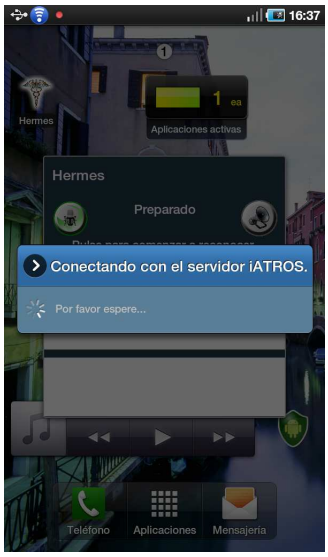
Una vez establecida la comunicación mediante sockets con el servidor se procede a enviar el fichero de sonido. Durante el envío se muestra una barra de progreso con el texto *"Enviando el fichero de sonido al servidor iATROS. Por favor espere ..."*, como se aprecia en la Figura 3.6(b), a fin de mantener informado al usuario sobre el progreso del envío del fichero.

Mientras el servidor iATROS realiza el reconocimiento del fichero de sonido enviado, el dispositivo móvil espera a que el servidor termine con el reconocimiento y le envíe la respuesta. Durante este periodo de tiempo en el dispositivo móvil se muestra una animación con el texto *"Procesando en el servidor iATROS. Por favor espere ..."*, como se puede ver en la Figura 3.6(c).

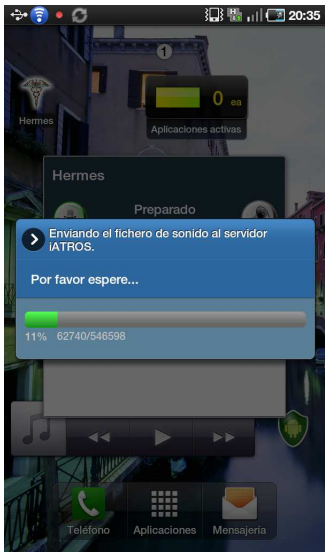
Una vez recibida la respuesta del servidor iATROS, ésta se muestra al usuario. En el primer

cuadro de texto se presenta la transcripción de la frase dicha por el usuario y en el cuadro de texto inferior su traducción al inglés, como se puede ver en la Figura 3.6(d). Al mismo tiempo, se reproduce la traducción utilizando el sintetizador de voz de Google en inglés.

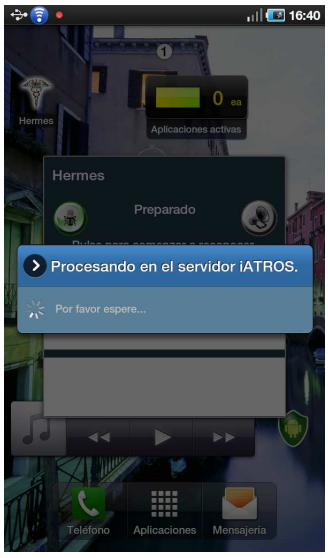
La Figura 3.6(d) muestra un ejemplo de funcionamiento del sistema. En el caso de que un usuario diga la frase *"Quiero una habitación tranquila con vistas al mar."*, el servidor iATROS daría como respuesta la transcripción de la frase dicha y su traducción al inglés: *"I want a quiet room with a view of the sea."*.



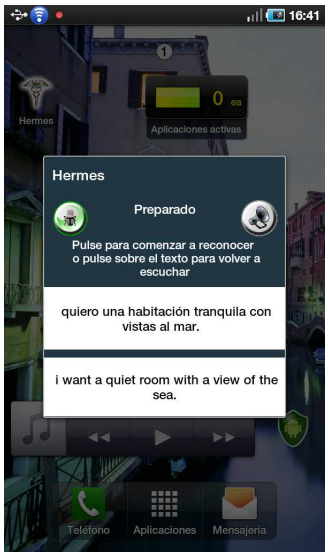
(a) Estableciendo la comunicación con el servidor iATROS.



(b) Envío del fichero de sonido al servidor iATROS.



(c) Procesando en el servidor iATROS.



(d) Muestra de la respuesta al usuario.

Figura 3.6: Actividad de comunicación con el servidor.

A continuación, la aplicación vuelve de la actividad de comunicación con el servidor a la actividad principal, manteniendo en los cuadros de texto la respuesta del servidor. De este modo, el usuario puede seguir leyendo la respuesta, e incluso si pulsa sobre los textos, puede volver a escucharlos hasta que esta respuesta sea sobreescrita por una nueva consulta del usuario, tal y como recuerdan los textos: *"Preparado"* y *"Pulse para comenzar a reconocer o pulse sobre el texto*

para volver a escuchar".

3.4. Servidor

La función del servidor es mantenerse a la espera de las peticiones de los clientes y dar una respuesta en cuanto recibe una petición. Las peticiones de los clientes llegan en forma de un fichero con la voz del usuario digitalizada y la respuesta que esperan los clientes es el resultado del procesado de este fichero con iATROS. El resultado esperado es la transcripción y la traducción al inglés de la frase dicha por el usuario.

Como servidor hemos utilizado una máquina virtual creada con el software de virtualización Virtual Box [17]. A este servidor virtual le hemos instalado el sistema operativo Debian [2] (una distribución basada en GNU/Linux), y el software de reconocimiento del habla iATROS [19]. En el Apéndice A se muestra el proceso de creación y configuración del servidor virtual, y en el Apéndice B se muestra el proceso de instalación de iATROS en dicho servidor.

Una vez creado y configurado el servidor virtual, se desarrolló una aplicación en Java encargada de establecer la comunicación mediante sockets con la aplicación cliente de los dispositivos móviles con Android y de ejecutar el RAH. Para ejecutar esta aplicación en el servidor utilizamos la siguiente orden:

```
$ java -jar Servidor.jar
```

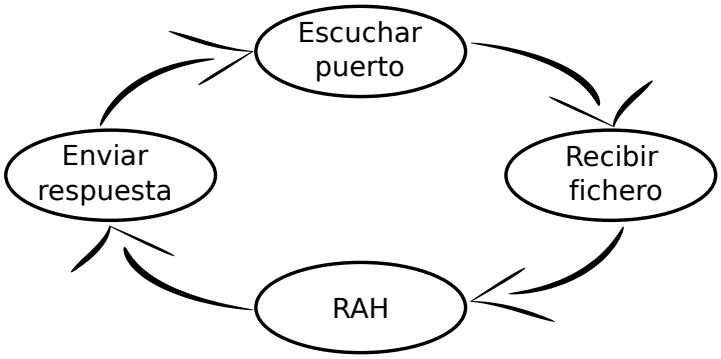


Figura 3.7: Funcionamiento de la aplicación servidor.

En la Figura 3.7 se muestra el grafo de funcionamiento de esta aplicación. Como se puede observar, está compuesta por cuatro fases: la fase de escucha del puerto del socket a la espera de que llegue una petición de un cliente, la fase de recepción del fichero de sonido, la fase de RAH y la fase de envío de la respuesta del reconocimiento al cliente. A continuación veremos con detalle cada una de estas cuatro fases.

3.4.1. Fase 1: Escuchar puerto

Cuando se ejecuta la aplicación el servidor se encuentra en la fase de escucha. En esta fase el servidor está a la escucha del puerto del socket TCP (por ejemplo, el puerto TCP número: *35557*) a la espera de peticiones de los clientes. En el servidor se puede ver el siguiente mensaje que informa de que está listo para atender peticiones:

```
Servidor iATROS preparado. Esperando nuevo cliente.
```

Cuando se recibe una petición de un nuevo cliente, la aplicación establece la comunicación y muestra por pantalla la confirmación:

```
Nuevo cliente aceptado.
```

A continuación, la aplicación del servidor pasa a la fase de recepción del fichero de sonido.

3.4.2. Fase 2: Recibir fichero

En esta fase la aplicación servidor acepta recibir el fichero de sonido, mostrando por la pantalla el nombre del fichero que está recibiendo, así como el identificador del dispositivo con Android que se lo está enviando:

```
Recibiendo el fichero sound.raw, desde el dispositivo: 9774d56d682e549c
```

Una vez completada la recepción del fichero, se muestra el siguiente mensaje de confirmación por la pantalla:

```
Fichero completo!
```

Con el fichero de sonido completo, la aplicación servidor ya puede pasar a la fase de RAH.

3.4.3. Fase 3: Reconocimiento Automático del Habla

En esta fase, la aplicación servidor realiza el análisis de la voz y posteriormente el reconocimiento mediante iATROS del fichero de audio recibido.

Para realizar el análisis de la voz del fichero recibido, la aplicación ejecuta *iatros-speech-cepstral*, como se puede ver en la siguiente orden:

```
iatros-speech-cepstral -c conf.feat -i sound.raw -o sound.cc
```

Donde: *sound.raw* es el fichero de entrada (con el audio obtenido con el dispositivo móvil), *sound.cc* es el fichero de salida (con los cepstrales que representan la voz del fichero de entrada) y *conf.feat* es el fichero de configuración (en el que se definen los valores de todas las variables del análisis de la voz definido en el estándar ETSI 201 108). El contenido del fichero de configuración *conf.feat*[1] se muestra a continuación:

```
# Configuration file for iATROS system
# Generated by iATROS Configurator
# Thu Oct 21 12:32:11 2011

<filter>
TriangularFilters 23
</filter>

<parameters>
FrameSize 410
```

[1] El significado de cada uno de los parámetros de configuración se puede consultar en la ayuda de iATROS.

```
SampleFreq 16000
SubSampleFreq 100
FFTlength 512
FrameShift 160
WindowLen 0.025625
Factor 0.97
WindowAlfa 0.54
CepCoefNumb 13
Channels 1
Bits 16
Frames 32
SecondsSilence 0.5
SilenceThreshold 50
Derivative 2
</parameters>

<buffers>
SizeSignal 100000
SizeCC 1000
</buffers>

<device>
InputDevice default
OutputDevice default
</device>
```

Una vez se ha obtenido el fichero de cepstrales se realiza el RAH con la versión offline de iATROS :

```
iatros-offline -c eutrans.cnf
```

Este proyecto está basado en EuTrans, concretamente en la tarea del *turista*, por lo que no ha sido necesario realizar un entrenamiento de los modelos utilizados por el reconocedor. En el fichero de configuración de EuTrans (*eutrans.cnf*) se definen los modelos sintáctico, léxico y acústico utilizados y todos los parámetros del reconocedor. A continuación se muestra el contenido del fichero *eutrans.cnf* [2]:

```
samples = sound.cc

[decoder]
hmm = models/HMM-Albayzin-128-CC
lexicon = models/expc.lx
lexicon-type = ATROS
grammar = models/exp.tr
grammar-type = FSM
start = <s>
end = </s>
histogram-pruning = 10000
do-acoustic-early-pruning = true
language-separator = |
beam = 400
grammar-scale-factor = 10
word-insertion-penalty = 0
```

[2]El significado de cada uno de los parámetros de configuración se puede consultar en la ayuda de iATROS.

Cuando iATROS encuentra la frase que corresponde con mayor probabilidad a la dicha por el usuario, la aplicación toma la respuesta de iATROS y pasa a la última fase: el envío de esta respuesta a la aplicación cliente.

3.4.4. Fase 4: Enviar respuesta

En esta última fase la respuesta obtenida de iATROS se envía mediante el socket a la aplicación cliente, que durante todo este tiempo ha permanecido a la espera. Esta respuesta, incluye la transcripción en castellano y su traducción al inglés separadas por el símbolo |, como se puede ver en el siguiente ejemplo:

```
Quiero una habitación tranquila con vistas al mar. | I want a quiet room with a view of the sea.
```

Para finalizar, una vez enviada la respuesta al cliente, el servidor vuelve a la fase de escucha del puerto a la espera de nuevas peticiones de los clientes.

```
Servidor iATROS preparado. Esperando nuevo cliente.
```

COMPROBACIÓN DEL SISTEMA

"El testing puede probar la presencia de errores
pero no la ausencia de ellos."

Edsger Dijkstra (1930-2002), físico e informático holandés.

Para comprobar el funcionamiento del sistema de interpretación automática (*Hermes*), se realizaron pruebas instalando la aplicación cliente en distintos dispositivos móviles (ver Tabla 4.1). La aplicación cliente funcionó perfectamente en todos los dispositivos excepto en el "Sony Ericsson Xperia X10 mini pro", en el que a pesar de no dar ningún tipo de error, la aplicación no pudo grabar el fichero de audio aunque disponía de memoria suficiente. A parte de este caso aislado, se comprobó que para los dispositivos con pantallas pequeñas el cuadro de texto inferior donde se muestra la traducción salía cortado. Esto demuestra que si se quisiera que este proyecto se pudiera utilizar de forma genérica en cualquier dispositivo móvil con Android habría que realizar una nueva interfaz gráfica adaptada a estas pantallas.

Marca y modelo	Android	CPU	RAM	Pantalla
Samsung Galaxy Tab 7"	2.2	1 GHz	512 MB	600 x 1024 px en 7"
HTC Wildfire	2.1	528 MHz	384 MB	240 x 320 px en 3.2"
HTC Hero	2.1	528 MHz	288 MB	320 x 480 px en 3.2"
Samsung Galaxy Ace	2.2	800 MHz	278 MB	320 x 480 px en 3.5"
LG Optimus ME P350	2.2	600 MHz	256 MB	240 x 320 px en 2.8"
Sansumg Galaxy S+	2.3	1.4 GHz	512 MB	480 x 800 px 4"
Sony Ericsson Xperia X10 mini pro	2.1	1 GHz	512 MB	320 x 480 px en 3"
HTC Desire HD	2.3	1GHz	768 MB	480 x 800 px en 4,3"

Tabla 4.1: Lista de dispositivos móviles con los que se ha comprobado el sistema

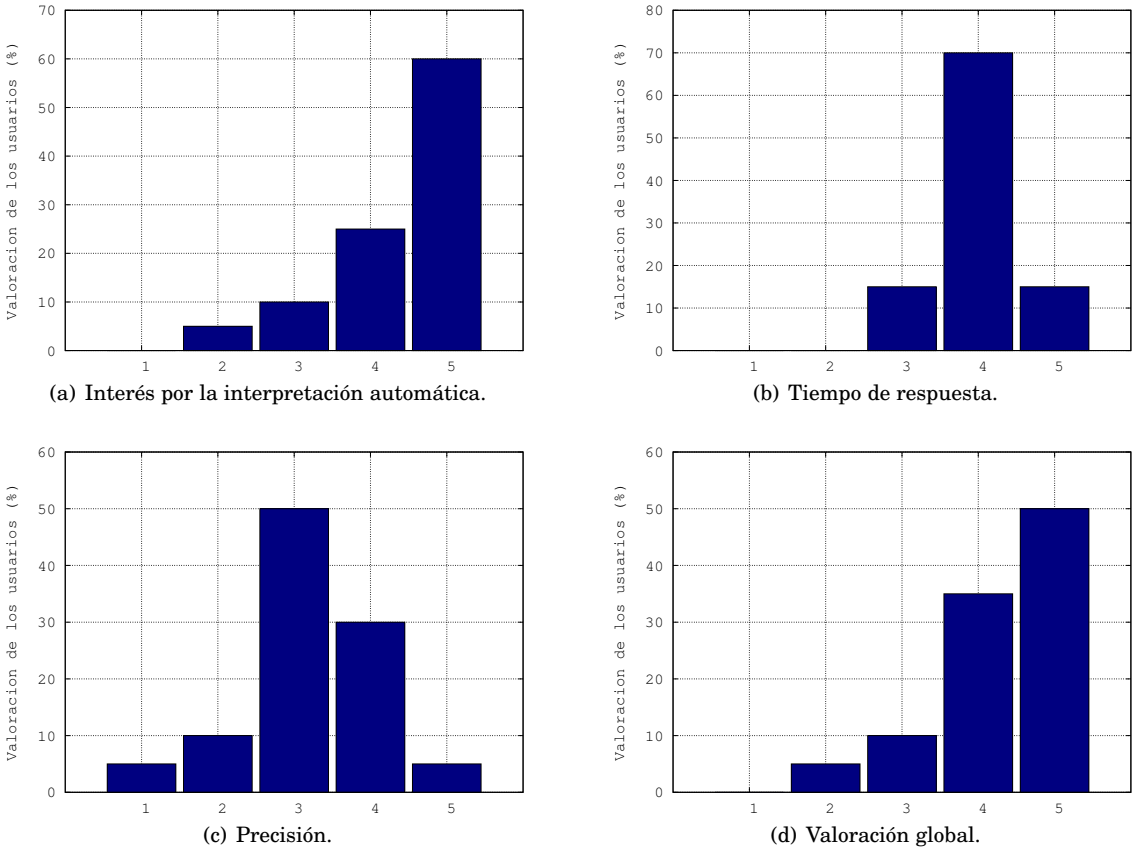
Además de comprobar el sistema utilizando distintos dispositivos móviles, se comprobó con 20 personas, diez hombres y diez mujeres, con un rango de edades comprendido entre los 11 y los 63 años con una gran variedad de ocupaciones (un cartero, una esteticista, un conductor de autobús, una psicóloga, una periodista y una escolar, entre otros).

Los usuarios utilizaron el sistema locutando las frases incluidas en el corpus de prueba mostrado en el Apéndice C y realizaron la encuesta del Apéndice D. La transcripción de sus respuestas se encuentra en el Apéndice E.

El primer objetivo de esta encuesta es obtener la valoración del sistema por parte de los usuarios en cuatro puntos: su interés por este tipo de sistemas, tiempo de espera de la respuesta, precisión en la interpretación y la satisfacción con el sistema. Para evaluar estos cuatro puntos se

pidió a los usuarios que los puntuaran del 1 al 5, donde 1 es "nada" y 5 es "mucho". En la Figura 4.1 se muestran las estadísticas obtenidas de sus respuestas:

La Figura 4.1(a) muestra las estadísticas sobre el interés por los sistemas de interpretación automática. Como podemos ver, a la mayoría de los encuestados les parecen muy interesantes los sistemas de interpretación automática. Centrándonos en nuestro sistema, en cuanto al tiempo de espera de la respuesta, como se puede ver en la Figura 4.1(b) todos los usuarios consideran que es aceptable, pero, también mejorable. Al evaluar la precisión del sistema, en la Figura 4.1(c) se muestra que la mayoría de los usuarios le dan un simple aprobado; por contra, un pequeño porcentaje considera que debe mejorarse para poder aprobar, mientras que un porcentaje menor considera que funciona muy bien. La diferencia de precisión de unos usuarios a otros se debe a que los modelos fueron entrenados utilizando un corpus genérico y el sistema no incluye un módulo de adaptación al locutor. Para terminar con la evaluación del sistema, se les pidió que dieran una puntuación global. En la Figura 4.1(d) se muestra la opinión global de los usuarios. Podemos ver que excepto un pequeño porcentaje que considera que el sistema funciona bastante mal, casi todos los usuarios muestran una elevada satisfacción con él.



(a) Interés por la interpretación automática.

(b) Tiempo de respuesta.

(c) Precisión.

(d) Valoración global.

Figura 4.1: Estadísticas de los resultados obtenidos durante la comprobación del sistema.

El segundo objetivo de la encuesta es averiguar qué aspectos del sistema les han gustado y cuáles consideran que deben mejorarse. La mayoría de los usuarios coincidieron en que lo que más les gusta de este sistema de interpretación automático es su facilidad de uso, algunos añadieron que les gustaba la utilidad que tiene, así como la posibilidad de utilizarlo en cualquier lugar, puesto que con un sistema como éste se sentirían más seguros a la hora de viajar a países angloparlantes.

Los sistemas de RAH y TA aún están lejos de ser perfectos; esto hace que nuestro sistema tampoco lo sea. La mayoría de usuarios opinaron en las encuestas que se debe mejorar el RAH, incluyendo más vocabulario (no reconoce palabras relacionadas con la comida, como "desayuno", "comida" o "cena"), así como incluir un módulo de adaptación al locutor, puesto que hubo algún

usuario al que prácticamente no reconocía. Por último, también hubo quien opinó que se debe mejorar la interfaz gráfica, mejorar el RAH en entornos ruidosos, añadir la detección de silencio para que no sea necesario volver a pulsar el botón al terminar de hablar y permitir al usuario corregir la respuesta del sistema, en el caso de que ésta no sea correcta.

El último objetivo de la encuesta es conocer la opinión de los usuarios sobre la valoración económica. El $30\,\%$ de los encuestados no pagaría por este tipo de servicios, las razones dadas fueron: por falta de costumbre, por no necesitarlo o porque "la cultura no tiene precio". El $70\,\%$ restante que sí pagaría, lo haría de distintas formas y en distintas cantidades: algunos pagarían entre 70 € y 150 € por una tableta en la que se incluyera este servicio (sin depender de ningún servidor), otros lo harían por la descarga del programa para utilizarlo en sus propios dispositivos móviles de 0,49 € a 50 € , y el resto pagaría por el uso del servicio con unas propuestas que van de 0,08 € al minuto a 6 € al mes.

CAPÍTULO 5

CONCLUSIONES Y TRABAJO FUTURO

"Torna quan vingues."

José Enrique Llorens Muñoz

5.1. Conclusiones

Para concluir este proyecto, debemos decir que se ha conseguido realizar el objetivo, que era crear un sistema de interpretación automático. Como vimos en los capítulos anteriores, este sistema de interpretación está compuesto por un servidor y una aplicación cliente.

Para el servidor se ha utilizado el software de virtualización VirtualBox en el que se ha instalado y configurado el sistema operativo Debian y el software de RAH iATROS. Además, se ha desarrollado una aplicación servidor para comunicar con la aplicación cliente en los dispositivos móviles.

La aplicación cliente se ha desarrollado para versiones de Android 2.1 y posteriores, utilizando el entorno de desarrollo Eclipse. En un principio para esta aplicación se desarrollaron cuatro actividades:

- La actividad principal, que es la encargada de interactuar directamente con el usuario, mostrando la interfaz gráfica, con los botones, cuadros de texto y las opciones de menú.
- La actividad de digitalización de la voz, en la que se graba en un fichero de audio sin compresión la señal recibida en el micrófono.
- La actividad de análisis de la voz, que en la versión definitiva del proyecto fue desechada por las razones explicadas anteriormente, derivando esta tarea al servidor en la versión final. En esta actividad se realizaba el análisis de voz del fichero de audio y se obtenían los cepstrales.
- La actividad de comunicación con el servidor; con ella la aplicación cliente establece la comunicación por sockets con el servidor, le envía el fichero de cepstrales y espera a recibir la respuesta de iATROS para mostrarla en los cuadros de texto y volver a la actividad principal.

El entrenamiento de modelos acústicos y de traducción no formaba parte de los objetivos de este proyecto, por lo que no fueron entrenados nuevos modelos y se utilizaron los modelos estimados previamente para la tarea del turista de EuTrans.

5.2. Trabajo futuro

A la conclusión de este trabajo quedaron algunos puntos por resolver que, por su complejidad, consideramos que en ellos podrían basarse futuros proyectos final de grado para estudiantes de ingeniería.

Sobre el servidor quedaron como tareas abiertas: la ampliación de uso a otras tareas e idiomas, la adición de un nuevo módulo de adaptación al locutor para mejorar la experiencia del usuario y la modificación de iATROS para añadir un servicio web, de modo que reconocimiento y traducción funcionen en streaming, reduciendo considerablemente el tiempo de espera de la respuesta por parte del usuario. En el caso de la aplicación cliente: la portabilidad a otros sistemas operativos móviles y el análisis de la señal acústica en los dispositivos móviles para detectar automáticamente el silencio y, con ello, el final de la frase, haciendo innecesario pulsar de nuevo el botón.

Bibliografía

[1] J. Amengual, A. Castaño, A. Castellanos, V. Jiménez, D. Llorens, A. Marzal, F. Prat, J. Vilar, J. Benedi, F. Casacuberta, et al. (2000). The EuTrans Spoken Language Translation System. *Machine Translation*, 15(1):75–103.

[2] Debian. Debian – The Universal Operating System (2011). <http://www.debian.org/> (Consultado: 1 de junio del 2011).

[3] J. Diaz, A. Rubio, A. Peinado, E. Segarra, N. Prieto, and F. Casacuberta. Development of Task Oriented Spanish Speech Corpora. In *Proceedings of EUROSPEECH*, volume 93, 1993.

[4] ETSI. 201 108 v1. 1.2 Speech Processing, Transmission and Quality aspects (STQ); Distributed speech recognition; Frontend feature extraction algorithm; Compression algorithms, 2000.

[5] G. Forney Jr (1973). The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278.

[6] M. Frigo and S. G. Johnson. FFTW (2011). <http://www.fftw.org> (Consultado: 20 de septiembre del 2011).

[7] Gartner. Gartner Says Sales of Mobile Devices in Second Quarter of 2011 Grew 16.5 Percent Year-on-Year; Smartphone Sales Grew 74 Percent (2011). <http://www.gartner.com/it/page.jsp?id=1764714> (Consultado: 4 de noviembre del 2011).

[8] J. T. Gironés (2011). *El gran libro de Android*. Madrid: Marcombo.

[9] Google. Android (2011). <http://www.android.com/> (Consultado: 6 de junio del 2011).

[10] Google. Android Developers (2011). <http://developer.android.com/index.html> (Consultado: 5 de septiembre del 2011).

[11] Google. Android NDK (2011). <http://developer.android.com/sdk/ndk/index.html> (Consultado: 9 de septiembre del 2011).

[12] D. Graser (2001). *Cisco Networking Academy Program: Second Year Companion Guide*. Cisco Press.

[13] K. Itoyama. jfftw3 (2009). <http://sourceforge.jp/projects/freshmeat_jfftw3/releases/> (Consultado: 10 de octubre del 2011).

[14] M. Luján-Mares, V. Tamarit, V. Alabau, C. Martínez-Hinarejos, M. Pastor, A. Sanchis, and A. Toselli (2008). iATROS: A speech and handwritting recognition system. *V Jornadas en Tecnologías del Habla (VJTH'2008)*, pp 75–78.

[15] C. Manning and H. Schütze (1999). *Foundations of statistical natural language processing*. MIT Press.

[16] P. Moreno, B. Raj, E. Gouvêa, and R. Stern. Multivariate-Gaussian-based cepstral normalization for robust speech recognition. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, volume 1, pp 137–140. IEEE, 1995.

[17] Oracle. Virtual Box (2011). <https://www.virtualbox.org/> (Consultado: 1 de junio del 2011).

[18] M. Pastor, A. Sanchis, F. Casacuberta, and E. Vidal. EuTrans: a Speech-to-Speech Translator Prototype. In *Proceedings of EuroSpeech*, pp 2385–2389, 2001.

[19] PRHLT. iATROS (2011). <http://prhlt.iti.es/page/projects/multimodal/idoc/iatros> (Consultado: 10 de julio del 2011).

[20] P. Suppes (1970). Probabilistic grammars for natural languages. *Synthese*, 22(1):95–116.

[21] Wikipedia. Dispositivo móvil — Wikipedia, La enciclopedia libre (2011). <http://es.wikipedia.org/w/index.php?title=Dispositivo_m%C3%B3vil&oldid=52769721> (Consultado: 20 de octubre del 2011).

[22] M. Zechner. libGDX (2011). <http://libgdx.badlogicgames.com/> (Consultado: 20 de agosto del 2011).

APÉNDICE A

CREACIÓN DE UN SERVIDOR VIRTUAL

"A crash reduces
your expensive computer
to a simple stone."

Computer Haiku Poem from Internet

Un servidor virtual es un software que emula a un servidor que puede ejecutar programas y ofrecer servicios como si fuese real. Esto permite tener en un mismo ordenador varios servidores con distintas configuraciones y diferentes sistemas operativos.

En este proyecto, se optó por utilizar la virtualización para evitar hacer pruebas directamente en la máquina física. Se creó un servidor virtual basado en Debian [2] mediante Virtual Box [17], siguiendo los siguientes pasos:

1. Descarga de Virtual Box y Debian.
2. Creación de la máquina virtual.
3. Instalación de Debian en la máquina virtual.
4. Instalación de ssh.
5. Reenvío de puertos en Virtual Box.

A.1. Descarga de Virtual Box y Debian

Desde la página web *http://www.virtualbox.org/wiki/Downloads* descargamos el software de virtualización Virtual Box para nuestro sistema operativo. Virtual Box se encuentra disponible para Linux, Windows, Mac OS y Solaris.

Desde la página web *http://www.debian.org/releases/* descargamos la última versión de Debian. Desde esta página se tiene la opción de descargar las imágenes ISO de CD o DVD de la última versión disponible en sus tres vertientes: estable, en pruebas e inestable para la arquitectura de nuestro sistema. Para este proyecto descargamos tan sólo la imagen ISO del primer CD de la versión estable para la arquitectura x86.

A.2. Creación de la máquina virtual

Una vez descargada la última versión de Virtual Box, la instalamos en nuestro sistema anfitrión y la ejecutamos. En Virtual Box procedemos a crear una nueva máquina virtual mediante el asistente que nos guiará en cada paso:

1. Le asignamos el nombre *Servidor_iATROS* a la máquina virtual e indicamos que el sistema operativo que vamos a instalar es de tipo *Linux* y versión *Debian*.
2. Le otorgamos *512 Mb* de memoria RAM.
3. Le creamos un nuevo disco duro de arranque virtual de tipo *expansión dinámica* de *10 Gb*.

A.3. Instalación de Debian en la máquina virtual

Una vez creada la máquina virtual, la seleccionamos y entramos en su configuración para montar una unidad de CD/DVD ROM a partir de la imagen ISO del CD de instalación de Debian que hemos descargado anteriormente.

Al iniciar la máquina virtual *Servidor_iATROS*, ésta arrancará desde el CD de Debian que hemos montado, mostrando una pantalla con las distintas opciones de instalación. Seleccionamos la primera opción *install* y procedemos a instalar el sistema siguiendo los siguientes pasos:

1. Seleccionamos el idioma, el país y el teclado deseados.
2. Le damos un nombre a la máquina *Servidor_iATROS* y un dominio *virtual*.
3. Elegimos el tipo de particionamiento del disco duro como guiado utilizando todo el disco y escribimos los cambios ejecutados en él.
4. Introducimos la contraseña de administrador.
5. Creamos un nuevo usuario, definiendo su nombre *user* y su contraseña *password*.
6. En la selección de programas, marcamos *Servidor web* y *Sistema estándar* dejando desmarcado *Entorno de escritorio*, ya que no instalaremos el entorno gráfico para emular fielmente un servidor real.
7. Instalamos GRUB.
8. Al finalizar la instalación, desmontamos la unidad (en el menú Dispositivos, opción Desmontar CD/DVD ROM) y pulsamos aceptar para reiniciar.
9. Al reiniciar actualizamos el sistema mediante *apt-get*.

```
$ apt-get update && apt-get upgrade && apt-get dist-upgrade
```

Una vez actualizado el servidor, si tecleamos *uname -a* obtendremos la información del sistema.

```
$ uname -a
Linux debian 2.6.32-5-686 #1 SMP Tue Sep 13 22:14:59 UTC 2011 i686 GNU/Linux
```

A.4. Instalación de SSH

Trabajaremos con el servidor a distancia mediante el protocolo SSH. Como el protocolo SSH no se instala por defecto, debemos instalarlo mediante la orden:

```
$ apt-get install openssh-server
```

Para comprobar que se ha instalado correctamente y que está funcionando, intentamos conectarnos con la siguiente orden:

```
$ ssh localhost
```

Si nos permite la conexión, es que todo funciona correctamente.

A.5. Redireccionamiento de puertos en Virtual Box

Por defecto, la configuración de red de Virtual Box viene definida como NAT, lo que significa que la máquina virtual está inaccesible desde el exterior mientras no se establezca una ruta desde un puerto local de la máquina física (anfitrión) a un puerto de la máquina virtual (invitado).

Sin embargo, nosotros queremos entrar en la máquina virtual desde fuera, por un puerto asignado al socket utilizado para la comunicación con la aplicación cliente y por SSH para poder realizar tareas de administración.

Para hacer esto, abriremos dos puertos en la máquina física: el 2222 para SSH y el 35557 para el socket. Los enrutaremos a los puertos 22 para ssh de la máquina virtual y al 35557 para iATROS.

Para realizar el redireccionamiento de los puertos, accedemos a la configuración de red de la máquina virtual como se puede ver en la Figura A.1 y apretamos en el botón *Reenvío de puertos*.

En la nueva pantalla, añadimos un reenvío del puerto TCP de la máquina física 2222 al 22 de la máquina virtual para SSH, y otro del puerto TCP de la máquina física 35557 al 35557 de la máquina virtual para iATROS, quedando como se puede ver en la Figura A.2

Una vez creado y configurado el servidor virtual cada vez que necesitemos realizar tareas de administración nos conectaremos al servidor virtual desde la máquina física mediante SSH, utilizando la orden:

```
$ ssh -p 2222 user@localhost
```

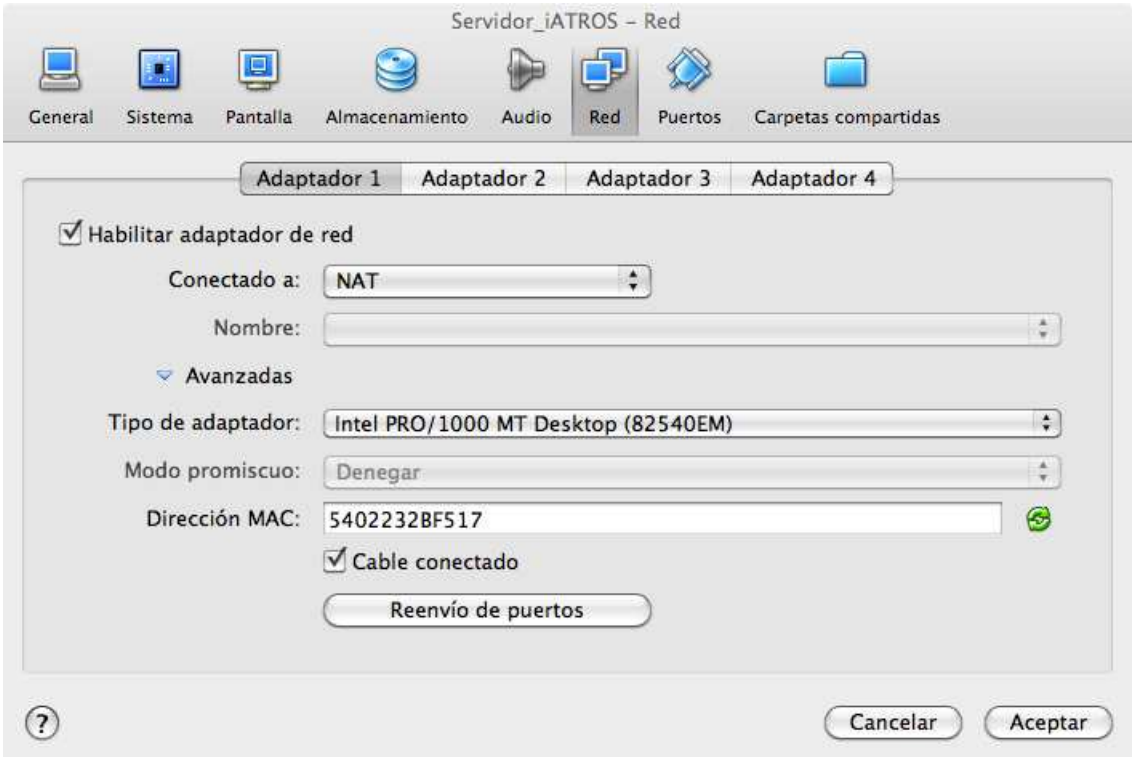


Figura A.1: Configuración de red.

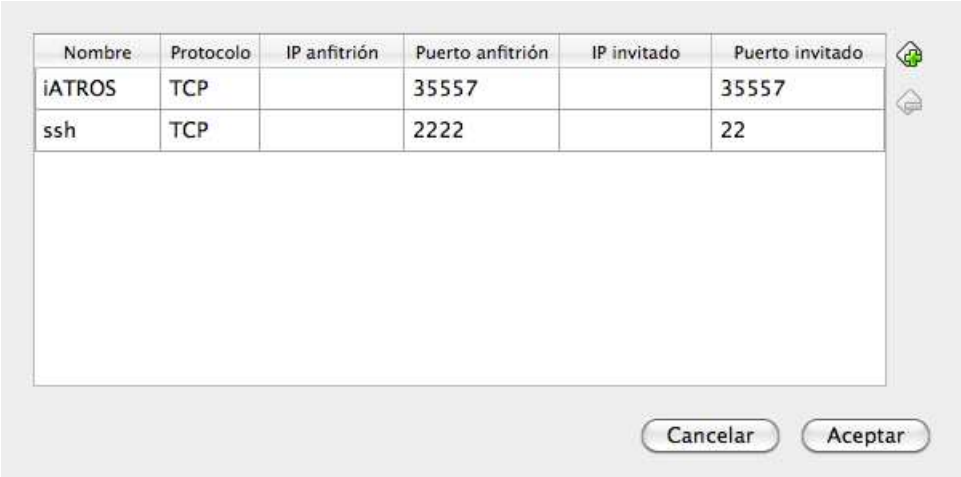


Figura A.2: Reenvío de puertos.

INSTALACIÓN DE iATROS

"La palabra es limitada
y no puede nombrar lo innombrable."

La elocuencia del silencio. Cuento clásico de la India

iATROS está preparado para ser fácilmente compilado en un entorno estándar de Linux. El código fuente de iATROS se proporciona con un archivo para ser fácilmente compilado utilizando la herramienta cmake. El código de iATROS y sus módulos deben ser descargados en el directorio 'local', donde los programas y las librerías se instalarán. Las librerías y los archivos binarios pueden ser instalados en el sistema (se necesitarán privilegios de administrador para esto) creando el sistema iATROS en el directorio de sistema /usr.

Para instalar iATROS en un sistema Debian se deben seguir los siguientes pasos:

1. Descarga del código fuente.
2. Preparación del sistema para la compilación.
3. Configuración.
4. Compilación e instalación.
5. Actualización de las variables de entorno.

A continuación, veremos cada uno de estos pasos en detalle.

B.1. Descarga del código fuente

En la página web del grupo de investigación PRHLT se encuentra disponible el código fuente de iATROS [19]. Procedemos a descargarlo mediante **wget** desde la carpeta de descargas */home/user/Descargas*.

```
$ wget http://prhlt.iti.es/projects/multimodal/idoc/iatros/files/iatros-v1.0.tgz
```

Una vez descargado el fichero *iatros-v1.0.tgz* que contiene el código fuente de iATROS, lo descomprimimos utilizando en descompresor **tar**.

```
$ tar xzvf iatros-v1.0.tgz
```

Al ejecutar la orden anterior se creará una nueva carpeta */home/user/Descargas/iatros-v1.0* que contendrá en su interior todo el código fuente de iATROS.

B.2. Preparación del sistema para la compilación

Para la compilación de iATROS son necesarias las herramientas, subversion, gcc, g++, make, cmake, flex y bison, por lo que si nuestro sistema no dispone de ellas, debemos instalarlas mediante *apt-get*.

```
$ apt-get install subversion gcc g++ make cmake flex bison
```

Los modulos *htr* y *speech* requieren las librerías *fftw3* y *asound2* en sus versiones de desarrollo.

```
$ apt-get install libfftw3-dev libasound2-dev
```

B.3. Configuración

Antes de compilar iATROS debemos configurar el entorno de compilación. Para ello debemos crear el directorio de instalación y desde éste ejecutar el script de configuración **configure** que preparará la compilación de iATROS.

```
$ mkdir /home/user/iatros-v1.0
$ cd /home/user/iatros-v1.0
$ ../Descargas/iatros-v1.0/configure
```

B.4. Compilación e instalación

A continuación, se ejecuta la orden **make** para compilar iATROS, y **make install** para instalarlo en el sistema.

```
$ make
$ make install
```

B.5. Actualización de las variables de entorno

Por último, como se ha instalado iATROS en un directorio no estándar, se deben actualizar las variables de entorno *PATH* y *LD_LIBRARY_PATH*, añadiendo las siguientes líneas al fichero *.bashrc*:

```
export IATROS_PATH=/home/user/iatros-v1.0
export PATH=$PATH:$IATROS_PATH/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$IATROS_PATH/lib
```

APÉNDICE C

CONJUNTO DE PRUEBA

"El lenguaje le da forma al mundo.
La palabra es el primer ejercicio del poder."

Vicente Romano

En este apéndice se muestra el conjunto de frases utilizadas para comprobar el funcionamiento y rendimiento del sistema de interpretación automática implementado.

1. Mi nombre es Pablo Alted.
2. Hice una reserva a nombre de Asunción Bellver.
3. ¿Podrían explicarme la factura de la habitación cero quince?
4. Mi nombre es Fernando Cabo.
5. He reservado una habitación hasta mañana a nombre de la señora Rosa Royo.
6. Tengo hecha una reserva a nombre de Jaime Suárez.
7. Por favor, ¿le importaría mostrarnos alguna habitación individual con teléfono?
8. Por favor, una habitación tranquila individual a nombre de Pablo Castillo.
9. Creo que se produjo un error en mi cuenta.
10. He hecho una reserva a nombre de Elvira García.
11. En la habitación hace mucho calor.
12. ¿Están incluidos en la cuenta los gastos?.
13. Tengo reservada una habitación doble a nombre de Manuel Ortega.
14. Tengo reservada una habitación a nombre del señor Paches.
15. Me llamo Inmaculada Arroyo.
16. Por favor, tengo la reserva de una habitación a nombre del señor Arenas.
17. Tengo hecha una reserva a nombre de Rafael Vázquez.
18. ¿Me llama a un taxi para la habitación doscientos doce?
19. Mi nombre es Ricardo Bacete.

20. Hice la reserva de una habitación doble a nombre de Susana Velasco.
21. Por favor, tengo hecha una reserva a nombre de Amelia Ferrando.
22. Por favor, quiero que me bajen los bultos a mi habitación.
23. Por favor, ¿le importaría mostrarme alguna habitación doble y tranquila?
24. Hemos reservado una habitación a nombre de la señora Silvia Mingot.
25. He reservado una habitación a nombre de la señora Amparo Cabedo.
26. Me llamo Gloria Serrano.
27. Por favor, tengo una reserva a nombre de Silvia Nadal.
28. Tengo una reserva a nombre de Rosalía Nebot.
29. Tengo hecha una reserva a nombre de Gerardo Quereda.
30. ¿Podría cambiarme a otra habitación más tranquila?
31. Llámenos a un taxi para la habitación doscientos veintitrés.
32. Son demasiado caras.
33. Por favor, deseo una habitación tranquila individual a nombre de Daniel Salsas.
34. Por favor, ¿nos repasa la cuenta de la habitación doce?
35. Por favor, he reservado una habitación a nombre de Carmelo Lobo.
36. Por favor, una habitación tranquila doble a nombre del señor Monsonis.
37. ¿Están anotados en la cuenta todos los gastos?
38. Hice una reserva a nombre de Miguel Ibáñez.
39. Una habitación tranquila que tenga buena vista, televisión y teléfono.
40. Tengo reservadas dos habitaciones con teléfono y televisión.
41. ¿Qué precio tiene una habitación individual para diez semanas?
42. Reservé una habitación individual para un día a nombre de Jacinto Ortega.
43. Tengo una reserva a nombre de Marta Orenga.
44. Por favor, hice una reserva a nombre de Silvia Arnau.
45. ¿Puedo pagar con cheques?

ENCUESTA DE SATISFACCIÓN

"Those are my principles,
and if you don't like them...
well, I have others."

Groucho Marx

En este apéndice se muestra la encuesta realizada a los usuarios que probaron el sistema, para conocer su grado de satisfacción y obtener una evaluación.

1. Información personal.
   a) Marca y modelo del dispositivo móvil:
   b) Versión de Android:
   c) Sexo (Hombre/Mujer):
   d) Edad:
   e) Ocupación:
2. Interés por los sistemas de interpretación automática *(En una escala del 1 al 5, donde 5 es "muy interesante" y 1 es "nada interesante").*
   - ¿Cómo de interesante es un sistema de interpretación automática para usted?
3. Tiempo de espera *(En una escala del 1 al 5, donde 5 es "muy rápido" y 1 es "muy lento").*
   - ¿Cómo le ha parecido el tiempo de espera de la respuesta?
4. Precisión *(En una escala del 1 al 5, donde 5 es "muy preciso" y 1 es "nada preciso").*
   - ¿Cómo de preciso le ha parecido el sistema?
5. Valoración global *(En una escala del 1 al 5, donde 5 es "muy satisfecho" y 1 es "nada satisfecho").*
   - ¿Cuál es su grado de satisfacción con el uso de Hermes?
6. Características.
   a) ¿Qué características del producto le han gustado?
      - Facilidad de uso, está de moda, otros (por favor, especifique):
   b) ¿Qué características considera que deben mejorarse?
7. Valoración económica.
   a) ¿Pagaría por este servicio de interpretación automática?
   b) ¿Cuál considera que sería el precio justo?

RESPUESTAS DE LOS USUARIOS

"Yo no procuro conocer las preguntas;
procuro conocer las respuestas."

Confucio (551 - 479 a. n. e.), filósofo chino.

En este apéndice se muestra la transcripción de las respuestas de los usuarios que participaron en la comprobación del funcionamiento del sistema de interpretación automático *Hermes*.

1a: Samsung Galaxy Ace	*1b:* 2.2	*1c:* Hombre	*1d:* 32 años	*1e:* Técnico electrónico
2: 3 *3:* 4	*4:* 3	*5:* 2	*6a:* Facilidad	*6b:* Traducción
7a: No	*7b:* La cultura no tiene precio.			

Tabla E.1: Respuestas del usuario número 1

1a: Samsung Galaxy S+	*1b:* 2.3	*1c:* Hombre	*1d:* 32 años	*1e:* Operario de fábrica
2: 4 *3:* 4	*4:* 3	*5:* 4	*6a:* Facilidad	*6b:* Reconocimiento
7a: No más de 0,80 €	*7b:* 0,49 € por la descarga del programa			

Tabla E.2: Respuestas del usuario número 2

1a: HTC Wildfire	*1b:* 2.1	*1c:* Mujer	*1d:* 31 años	*1e:* Periodista
2: 4 *3:* 4	*4:* 4	*5:* 4	*6a:* Facilidad y ubicuidad	*6b:* Interfaz gráfica
7a: Sí	*7b:* 3 € por la descarga del programa			

Tabla E.3: Respuestas del usuario número 3

1a: HTC Hero	*1b:* 2.1	*1c:* Hombre	*1d:* 35 años	*1e:* Técnico Informático
2: 5 *3:* 4	*4:* 4	*5:* 4	*6a:* Facilidad	*6b:*
7a: No	*7b:* 0 €			

Tabla E.4: Respuestas del usuario número 4

1a: Samsung Galaxy Tab	*1b:* 2.2	*1c:* Mujer	*1d:* 30 años	*1e:* Auxiliar de clínica
2: 5 *3:* 4	*4:* 3	*5:* 5	*6a:* Facilidad	*6b:* Más vocabulario
7a: Sí	*7b:* 150 € por el sistema completo en una tableta			

Tabla E.5: Respuestas del usuario número 5

1a: Samsung Galaxy Tab	*1b:* 2.2	*1c:* Mujer	*1d:* 26 años	*1e:* Policía
2: 5 *3:* 4	*4:* 1	*5:* 4	*6a:* Su utilidad en el trabajo	*6b:* Adap. al locutor
7a: Sí	*7b:* 100 € por el sistema completo.			

Tabla E.6: Respuestas del usuario número 6

1a: Samsung Galaxy Ace	*1b:* 2.2	*1c:* Hombre	*1d:* 31 años	*1e:* Camarero
2: 5 *3:* 4	*4:* 4	*5:* 5	*6a:* Facilidad	*6b:* Reconocimiento
7a: Sí	*7b:* 5 € por la descarga del programa.			

Tabla E.7: Respuestas del usuario número 7

1a: Samsung Galaxy Tab	*1b:* 2.2	*1c:* Mujer	*1d:* 42 años	*1e:* Auxiliar de clínica
2: 5 *3:* 4	*4:* 5	*5:* 5	*6a:* Seguridad al viajar	*6b:* Vocabulario
7a: Sí	*7b:* 100 € por el sistema completo.			

Tabla E.8: Respuestas del usuario número 8

1a: Samsung Galaxy Tab	*1b:* 2.2	*1c:* Mujer	*1d:* 11 años	*1e:* Escolar
2: 5 *3:* 4	*4:* 3	*5:* 5	*6a:* Facilidad	*6b:* RAH en entornos ruidosos
7a: Sí	*7b:* 75 € por el sistema completo.			

Tabla E.9: Respuestas del usuario número 9

1a: HTC desire HD	*1b:* 2.3.5	*1c:* Hombre	*1d:* 26 años	*1e:* Estudiante de Doc. Informática
2: 5 *3:* 4	*4:* 3	*5:* 4	*6a:*	*6b:* Detección del silencio.
7a: No	*7b:* 0 €			

Tabla E.10: Respuestas del usuario número 10

1a: Sony Ericsson Xperia mini pro	*1b:* 2.1	*1c:* Mujer	*1d:* 30 años	*1e:* Psicóloga	
2: 5 *3:* 4	*4:* 3	*5:* 4	*6a:* Facilidad	*6b:*	
7a: Sí	*7b:* Pago por servicio, 0,08 € por minuto.				

Tabla E.11: Respuestas del usuario número 11

1a: LG Optimus ME P350	*1b:* 2.2	*1c:* Hombre	*1d:* 63 años	*1e:* Cartero
2: 4 *3:* 3	*4:* 3	*5:* 5	*6a:* Facilidad	*6b:* Más vocabulario
7a: Sí	*7b:* NS/NC			

Tabla E.12: Respuestas del usuario número 12

1a: Samsung Galaxy Tab	*1b:* 2.2	*1c:* Mujer	*1d:* 60 años	*1e:* Sastre
2: 4 *3:* 3	*4:* 3	*5:* 5	*6a:* Facilidad	*6b:* Más vocabulario
7a: Sí	*7b:* NS/NC.			

Tabla E.13: Respuestas del usuario número 13

1a: Samsung Galaxy Ace	*1b:* 2.2	*1c:* Mujer	*1d:* 31 años	*1e:* Camarera
2: 5 *3:* 3	*4:* 3	*5:* 5	*6a:* Facilidad	*6b:* Reconocimiento
7a: Sí	*7b:* Pago por servicio, 6 € al mes.			

Tabla E.14: Respuestas del usuario número 14

1a: Samsung Galaxy Tab	*1b:* 2.2	*1c:* Mujer	*1d:* 24 años	*1e:* Esteticista
2: 5 *3:* 4	*4:* 4	*5:* 5	*6a:* Facilidad	*6b:* Ampliación a otras tareas
7a: Sí	*7b:* 50 € por la descarga.			

Tabla E.15: Respuestas del usuario número 15

1a: Samsung Galaxy Tab	*1b:* 2.2	*1c:* Hombre	*1d:* 32 años	*1e:* Conductor
2: 5 *3:* 4	*4:* 3	*5:* 4	*6a:* Ubicuidad	*6b:* Permitir corregir
7a: Sí	*7b:* Pago por servicio, 5 € al mes.			

Tabla E.16: Respuestas del usuario número 16

1a: Samsung Galaxy Tab	*1b:* 2.2	*1c:* Mujer	*1d:* 21 años	*1e:* Camarera
2: 5 *3:* 4	*4:* 4	*5:* 5	*6a:* Facilidad	*6b:* Ampliación a otros idiomas
7a: Sí	*7b:* 70 € por el sistema completo.			

Tabla E.17: Respuestas del usuario número 17

1a: Samsung Galaxy Tab	*1b:* 2.2	*1c:* Hombre	*1d:* 27 años	*1e:* Técnico electrónico
2: 3 *3:* 5	*4:* 2	*5:* 3	*6a:* Facilidad	*6b:* Más vocabulario.
7a: No	*7b:* No lo utilizaría.			

Tabla E.18: Respuestas del usuario número 18

1a: Samsung Galaxy Tab	*1b:* 2.2	*1c:* Hombre	*1d:* 24 años	*1e:* Técnico forestal
2: 2 *3:* 5	*4:* 2	*5:* 3	*6a:* Facilidad	*6b:* Mejorar el RAH.
7a: No	*7b:* No tengo acostumbre de pagar por este tipo de servicios.			

Tabla E.19: Respuestas del usuario número 19

1a: Samsung Galaxy Tab	*1b:* 2.2	*1c:* Hombre	*1d:* 23 años	*1e:* Albañil
2: 4 *3:* 5	*4:* 4	*5:* 5	*6a:*	*6b:* Ampliación a otras tareas.
7a: No	*7b:* No lo necesito.			

Tabla E.20: Respuestas del usuario número 20