Ingeniería de Telecomunicación

PROYECTO FIN DE CARRERA

Desarrollo de un Portal de Voz de Atención al Ciudadano mediante VoiceXML

Autora: María García Jiménez

Tutor/Director: David Griol Barres

Julio de 2011

Título: Desarrollo de un Portal de Voz de Atención al Ciudadano mediante VoiceXML
Autor: María García Jiménez
Director: David Griol Barres

EL TRIBUNAL

Presidente: _____

Vocal: _____

Secretario: _____

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día __ de _____ de 20__ en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Agradecimientos

En primer lugar le quería agradecer a mi tutor, David Griol, por darme la oportunidad de involucrarme en el mundo de los sistemas de diálogo, pero sobre todo por su dedicación y por su profesionalidad en la dirección del Proyecto, además de haberme guiado y animado en cada momento. Le agradezco todo cuanto he aprendido de él.

Gracias a mis compañeros de la Universidad, por todos los momentos vividos estos últimos años. Sé que algunos de ellos serán amigos para siempre.

Finalmente, se lo quiero dedicar a mis padres, a mis tías Tere y Delia y a mi hermana Eva, agradeciéndoles haberme apoyado en cada instante, y a Fran, por toda su ayuda y sus consejos, y por supuesto, por estar a mi lado, animándome siempre. Gracias a ellos he logrado llegar al final con fuerza e ilusión.

A todos ellos, GRACIAS.

Resumen

El objetivo principal de este Proyecto Final de Carrera es el desarrollo de un portal de voz de información municipal utilizando el lenguaje de programación VoiceXML. A través del portal desarrollado, los usuarios pueden interactuar telefónicamente mediante voz o teclado para acceder a información y a un conjunto de servicios del municipio de Alcorcón.

El portal de voz ofrece al ciudadano distintas funcionalidades: consulta de información sobre el Ayuntamiento (equipo de gobierno, concejalías, etc.), información de la ciudad (historia, datos geográficos y demográficos, accesos a la ciudad, páginas amarillas, cartelera, noticias, eventos, información meteorológica, etc.), realización de diversas gestiones y trámites (comprobación de listados y expedientes personales, reserva de instalaciones municipales y reserva de citas), realización de encuestas, acceso al buzón del ciudadano para dejar mensajes de sugerencias o reclamaciones y ser transferido a la centralita del Ayuntamiento para ser atendido por un tele-operador.

Gracias al sistema desarrollado se facilitan estos servicios de atención ciudadana de forma automatizada, permitiendo al ciudadano acceder a ellos durante las 24 horas del día de una forma natural y en entornos en los que utilizando los interfaces tradicionales de acceso, como el teclado o el ratón, no sería posible. Adicionalmente, se facilita este acceso a personas con discapacidades motoras o visuales, contribuyendo a eliminar barreras de acceso y posibilitar un mundo tecnológico más accesible.

La aplicación se apoya además en otra tecnologías adicionales, como son la utilización de bases de datos (MySQL), servidores web y de VoiceXML (x10Hosting y Voxeo Evolution) y uso de diferentes lenguajes de programación (SQL, PHP, HTML), que la hacen más dinámica y flexible, aumentando su calidad y eficiencia.

El proyecto se complementa con un estudio detallado y análisis de los sistemas de diálogo y de la aplicación del estándar VoiceXML para su desarrollo, llevado a cabo con el objetivo de aplicar todo su potencial para completar con éxito este Proyecto Final de Carrera.

Palabras clave: sistemas de diálogo, interacción oral, portal de voz, VoiceXML, Voxeo.

Abstract

The main objective of this Bachelor Project is to develop a voice portal to provide municipal information using the VoiceXML programming language. By means of the developed portal, users can interact via voice or the telephone keypad to access information and a set of services from Alcorcón (Madrid, Spain).

The developed voice portal offers citizens different functions: consult information about the City Council (Government Team, Councils, etc.), know city information (history, geographic and demographic data, access to the city, yellow pages, movie show times, news, events, weather, etc.), carry out several steps and procedures (check lists and personal files, book municipal facilities or make an appointment), complete surveys, access citizen's mailbox to leave messages for suggestions and complaints, and be transferred to the City Council to be attended by a tele-operator.

Thanks to the developed application, these services are automatically provided, allowing a 24 hour a day access to information in a more natural way and in environments in which using the traditional access interfaces, such as keyboard or mouse, this access would not be possible. In addition, the application facilitates the access for people with visual or motor disabilities, helping them to eliminate access barriers and enabling a more accessible technology world.

The voice portal integrates additional technologies, such as the use of databases (MySQL), web and VoiceXML servers (x10Hosting and Voxeo Evolution), and several programming languages (SQL, PHP, HTML), which make it more dynamic and flexible and increase its quality and efficiency.

The project has been extended with a detailed study and analysis of spoken dialog systems and the application of the VoiceXML standard for their implementation. This study has been very useful to learn and apply the full potential of these technologies for the complete development of the voice portal.

Keywords: spoken dialog systems, oral interaction, voice portal, VoiceXML, Voxeo.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

En este primer capítulo se presentan los antecedentes del Proyecto Final de Carrera, se establecen los objetivos que se definieron para el mismo y se describen los aspectos generales relativos a las fases de desarrollo, la planificación temporal y los medios empleados para su consecución. En último lugar, se detalla la estructura de capítulos que conforma el presente documento.

1.1 Antecedentes

En la actualidad, debido al gran desarrollo tecnológico, los ordenadores y dispositivos telefónicos se han convertido en una parte esencial de nuestras vidas. Gracias a ellos accedemos a información y a funcionalidades que hace unos años eran impensables. Se quiere poder acceder a estos servicios en cualquier lugar y momento, y de una forma fácil y eficiente. Por este motivo, se busca el desarrollo de interfaces que establezcan medios de comunicación entre nosotros y estas máquinas.

Ya que la voz es el medio más natural e intuitivo para interactuar y comunicarse, las aplicaciones basadas en sistemas de diálogo [MCT04] se han convertido en una de las principales opciones para dotar a los dispositivos electrónicos de capacidad de comunicación.

Estos programas informáticos tienen como principal finalidad interactuar con los usuarios oralmente o de forma multimodal para proporcionarles un determinado servicio, como, por ejemplo, información y reserva de viajes de avión o tren, información meteorológica, centralitas telefónicas o control de dispositivos domóticos.

En los últimos años, los servicios de voz están viviendo una gran evolución y expansión, y su implantación es cada día más frecuente, debido en parte a la explosión de la telefonía móvil que los ha impulsado hasta su nivel de desarrollo actual, y sobre todo por las numerosas ventajas que ofrecen: mayor rapidez, efectividad y facilidad a la hora de realizar tareas de forma automática.

Según los resultados de la encuesta publicada por el INE sobre "Equipamiento y uso de tecnologías de la información y comunicación en los hogares - 2010" [ENC10], el 99,4% de los hogares dispone de teléfono (fijo o móvil), siendo un 80,3% los hogares españoles que disponen de teléfono fijo y un 94,6% los que disponen de teléfono móvil. Respecto a viviendas con acceso a Internet, el 59,1% de los hogares españoles tiene conexión a la Red. Por otro lado, el porcentaje de viviendas equipadas con algún tipo de ordenador sigue creciendo en los hogares españoles alcanzado actualmente el 68,7%.

Estos datos ponen de manifiesto que la balanza se decanta hacia la utilización del teléfono y el acceso por voz antes que a otros medios, ya que la mayoría de los hogares españoles disponen de teléfono fijo o móvil, pudiendo incluso utilizar el acceso telefónico para la consulta oral de la información y servicios en la Red mediante los sistemas de diálogo. El número de accesos a la Red está en continuo crecimiento, al igual que el porcentaje de viviendas equipadas con algún tipo de ordenador, contribuyendo a que la utilización de tecnologías que apuesten por aplicaciones basadas en interfaces orales tengan una posición cada vez más relevante en el futuro tecnológico.

Para la implementación de los sistemas de diálogo, el W3C [W3C] propone el estándar VoiceXML (Voice Extensible Markup Language) [VOICEXML]. Este lenguaje de programación permite la interacción persona-máquina a partir de elementos como traducción de texto-a-voz, reproducción de audio grabado, reconocimiento de voz, reconocimiento de tonos DTMF (Dual-Tone Multi-Frequency), grabación de entrada hablada, control de flujo de diálogo y funciones de telefonía (transferencia de llamada, desconexión, etc.).

VoiceXML es una tecnología independiente de la plataforma que permite la portabilidad y transferencia de datos entre aplicaciones heterogéneas. Si se utiliza de manera conjunta con otros estándares, proporciona una base sólida para el desarrollo de sistemas de diálogo.

Por tanto, es esta tecnología el principal objeto de análisis y estudio del presente Proyecto Final de Carrera, en el que además se hace uso de ella para implementar un sistema de diálogo completo para un dominio específico: un portal de voz municipal para el Ayuntamiento de Alcorcón.

En la fase inicial de planificación del proyecto, se recabó información general referente a la situación actual en la prestación de servicios a los ciudadanos por parte de los Ayuntamientos. La principal conclusión obtenida es que éstos deberían tener servicios de atención ciudadana que cumplan las funciones de difundir información, orientar y

asesorar al ciudadano, recibir y tramitar las sugerencias y quejas, así como realizar registros y coordinar gestiones y trámites.

Una gran ventaja es poder acceder a estos servicios de forma no presencial. Gracias a los avances tecnológicos, se han creado y han ido evolucionando distintos canales que proporcionan estas prestaciones. La página web es uno de los medios más utilizados para municipios que disponen de presupuesto e infraestructuras que lo permitan, pero el canal más común es la línea telefónica de atención al ciudadano.

La Ley 11/2007 sobre el acceso electrónico de los ciudadanos a los Servicios Públicos [LAECSP], incide en el *fomento de múltiples canales de acceso a la información* como una de las principales obligaciones de los Ayuntamientos, y reconoce explícitamente el *derecho de los ciudadanos a relacionarse con las Administraciones Públicas por medios electrónicos*.

El primer pensamiento de todos, incluida la propia Administración, se centra en la utilización de Internet para realizar tramitaciones de forma electrónica. No obstante, aunque el articulado se centra en Internet, la misma ley especifica la inclusión de otros canales de acceso como la telefonía:

- "Canal es cualquier medio de difusión de contenidos o servicios incluyendo el canal presencial, el telefónico y el electrónico y otros que puedan existir".
- "Documento electrónico hace referencia a información de cualquier naturaleza en forma electrónica, archivada en soporte electrónico (lo que incluye el audio y vídeo)".
- "Medio electrónico es cualquier mecanismo que permita producir, almacenar o transmitir documentos [...] incluyendo cualquier red de comunicación como Internet, telefonía fija o móvil u otras (TDT)".

Por tanto, un portal de voz municipal cumple con esta Ley, constituyendo una herramienta que facilita a los ciudadanos el acceso oral a los servicios públicos de su ciudad a través del teléfono en cualquier instante y lugar.

Si se añadiera este nuevo sistema implementado a la infraestructura tecnológica del Ayuntamiento, se incrementaría la disponibilidad de estas centralitas, se ayudaría a reducir los turnos de los tele-operadores y sus costes asociados, se incrementarían las horas de servicio a 24x7, se disminuiría la tasa de llamadas perdidas, se proporcionaría una tasa de redireccionamiento mejor solucionada, se mejoraría la flexibilidad para responder a las necesidades del cliente o a picos de llamadas y se controlarían automáticamente las llamadas recibidas.

A todas estas ventajas hay que añadir que una interfaz oral posibilita la comunicación en casos y lugares en los que con otros medios tradicionales como el teclado o el ratón no sería posible como, por ejemplo, entornos reducidos o aplicaciones accesibles desde vehículos, además de facilitar el acceso a personas con discapacidades visuales o motoras.

En cuanto a la existencia de portales de voz municipales en España, las pocas aplicaciones existentes actualmente ofrecen únicamente el acceso a una grabación para la

elección de una determinada área o sección entre un conjunto reducido, a la que el usuario es redirigido para ser atendido personalmente por un operador del Ayuntamiento (por ejemplo, el portal desarrollado para el Cabildo de Gran Canaria).

Actualmente, *Ambiser Innovaciones* (*http://www.ambiser.es/*), consultora enfocada en el desarrollo de proyectos para las Administraciones Públicas, está implantando portales de voz municipales de dos tipos:

- **Informativos:** se proporciona mediante voz la información solicitada por el usuario del portal:
  - Información del tiempo.
  - Información de farmacias de guardia.
  - Información de horarios.
- **Personalizados:** el usuario realiza una autenticación en el portal y accede a información personal o realiza tareas personales:
  - Solicitud de cita en un servicio municipal.
  - Reserva de una pista deportiva.
  - Quejas / Sugerencias.
  - Inscripciones.
  - Votaciones / Encuestas.

Ejemplos de estos portales de voz, todavía no disponibles, se encuentran en los municipios de Valdepeñas (ver Figura 1.1), Arroyo de la Encomienda, Ávila y Leganés.

Adicionalmente, existen otros tipos de acceso oral a la información municipal. Por ejemplo, en la página Web del Ayuntamiento de Santander (http://portal.ayto-santander.es/portal/page/portal/inet_santander/general/accesibilidad) se puede descargar la aplicación *VoxWebPC* [VOXWEBPC], que permite al usuario navegar oralmente por las páginas que integran el portal del Ayuntamiento, además de poder escuchar el contenido de estas páginas web con una voz sintetizada. Otro ejemplo del uso de estas tecnologías es el portal web del Ayuntamiento de Zaragoza (http://www.zaragoza.es/ciudad/servicios/voz.htm), que ofrece la tecnología *Dixerit* [DIXERIT] en las secciones de:

- Primera Página,
- Noticias Municipales,
- Tablón municipal,
- Agenda Zaragoza,
- Zaragoza Sin Barreras.

Un sistema similar al caso de Zaragoza es el implantado en Alicante (http://www.alicante.es), donde la Concejalía de Modernización de Estructuras Municipales del Ayuntamiento de Alicante ha puesto en marcha la tecnología *ReadSpeaker* [READSPEAKER] en todas las páginas en castellano y valenciano de su portal web. Tanto *ReadSpeaker* como *Dixerit* convierten los contenidos de la web en voz digitalizada de alta calidad.

Figura 1.1. Portal de voz del municipio de Valdepeñas.

El portal de voz desarrollado en este Proyecto Final de Carrera es pionero en ofrecer una extensa y completa oferta de servicios municipales accesibles a través de un número de teléfono.

Mediante el portal de voz de Alcorcón se crea una nueva vía de comunicación útil, eficiente, fácil de utilizar y accesible, de forma que se contribuye a mejorar el acceso a la información, ofrecer la posibilidad de consultar los servicios e incrementar las posibilidades de realización de trámites y gestiones por parte de los ciudadanos del municipio.

1.2 Objetivos

Tal y como se ha descrito en la sección anterior, el objetivo fundamental del Proyecto Final de Carrera es el desarrollo de un sistema de diálogo en el que los usuarios puedan interactuar mediante voz y/o señalización DTMF para acceder a diferentes servicios municipales de Alcorcón.

Utilizando las inmensas posibilidades y potencial de los sistemas de diálogo, se ha implementado una aplicación útil, que en el día a día pueda ser utilizada por los usuarios para resolver sus requerimientos acerca del municipio.

De esta forma, los ciudadanos pueden, por ejemplo, utilizar el sistema desarrollado para obtener información sobre el Ayuntamiento o la ciudad, noticias, eventos e información meteorológica, realizar diferentes gestiones y trámites, rellenar encuestas, usar el buzón del ciudadano y ser atendidos por un tele-operador.

Alcorcón actualmente dispone de una página web y un único teléfono de centralita. Si se añadiera este nuevo sistema implementado, se mejoraría el servicio de atención ciudadana aumentando la disponibilidad, la flexibilidad, el control de las llamadas y reduciendo costes y llamadas perdidas.

En base a este objetivo principal, se definieron los siguientes objetivos parciales:

- Llevar a cabo un estudio completo de los sistemas de diálogo, desde el concepto de interfaz oral a su arquitectura de módulos, pasando por los distintos criterios para su clasificación y finalmente analizando las aplicaciones y ejemplos concretos existentes en la actualidad para poder así aplicar todas sus posibilidades en el desarrollo de la nueva aplicación.

- Completar un estudio detallado del estándar VoiceXML, así como de los demás lenguajes necesarios para implementar todas las funcionalidades del portal de voz, con el fin de conseguir que la comunicación con el usuario sea lo más natural posible.

- Permitir el uso de esta aplicación en entornos en los que no podrían ser utilizados un ordenador con teclado y ratón o no existir el acceso a Internet, por ejemplo, en el entorno de un automóvil.

- Facilitar el uso de estos servicios a usuarios que desconocen tanto el uso de un ordenador o de Internet, por ejemplo, personas ancianas que están acostumbradas únicamente al uso del teléfono.

- Facilitar el acceso a las funcionalidades ofrecidas en esta aplicación a personas con discapacidades motoras o visuales, o a personas mayores con capacidades disminuidas, contribuyendo a hacer el mundo tecnológico un poco más accesible para ellas.

1.3 Fases de desarrollo

Para la realización del presente apartado y del siguiente (Apartado 1.4 - Planificación temporal) se han aplicado los conocimientos adquiridos en la asignatura Proyectos de Ingeniería.

En primer lugar se establecen las fases de desarrollo y las tareas en las que se divide el proyecto. El presente Proyecto se ha dividido en 3 principales fases de desarrollo, que se describen a continuación.

Fase 1: Planificación.

- **Estudio de los sistemas de diálogo**: aproximación a los sistemas de diálogo y estado actual de la investigación en este campo.

- **Planificación y análisis de requisitos del portal de voz**: determinación y selección de las funcionalidades que proporcionan los Sistemas de Diálogo para realizar la aplicación del portal de voz.

- **Estudio de las tecnologías necesarias**: estudio de la plataforma Voxeo, del estándar VoiceXML, del lenguaje de programación PHP, del gestor de base de datos,

así como implementación de ejemplos prácticos para la asimilación de estas tecnologías.

Fase 2: Desarrollo.

- **Análisis y diseño inicial**: división de las distintas funcionalidades en los diferentes módulos y submódulos del sistema.

- **Implementación del sistema**: Desarrollo de todos los módulos y submódulos, así como de las páginas y enlaces necesarios para facilitar la interacción entre ellos.

- **Pruebas unitarias**: estudio, configuración y realización de pruebas funcionales para cada módulo.

- **Pruebas de integración y de sistema**: proceso de pruebas del sistema completo hasta alcanzar una versión completamente estable.

- **Evaluación de la aplicación**: estudio de los sistemas de evaluación para interfaces orales, realización de las preguntas de la encuesta de evaluación, implementación de la página *HTML* con el cuestionario, recogida y análisis de resultados.

Fase 3: Documentación.

- **Memoria del Proyecto Final de Carrera**: Redacción del presente documento de memoria del Proyecto de Final de Carrera.

- **Preparación de la presentación**.

Para la determinación de las tareas se construye en un *diagrama WBS* (*Work Breakdown Structure*). Se trata de una estructura en árbol en la que se ordenan las tareas y subtareas con arreglo a un determinado criterio. La principal utilidad de este tipo de diagrama es proporcionar un método sistemático para confeccionar la lista de tareas.

El *WBS* utilizado para el Proyecto se muestra en la Figura 1.2. Tal y como puede observarse, para el Proyecto se han definido tres familias de tareas, que constituyen el primer nivel de jerarquía a partir del cual se han desarrollado las 10 tareas que componen el proyecto (tareas con índice de 1 a 10). Estas 10 tareas constituyen el segundo y último nivel. Las tareas que pertenecen a una misma familia aparecen formando una columna debajo del nombre de la familia.

Sólo el último nivel (en este caso el nivel 2) representa tareas reales y los niveles superiores se pueden considerar como fases que resumen el proyecto. Una fase, o grupo de tareas, es en el lenguaje de *Project*, una tarea resumen.

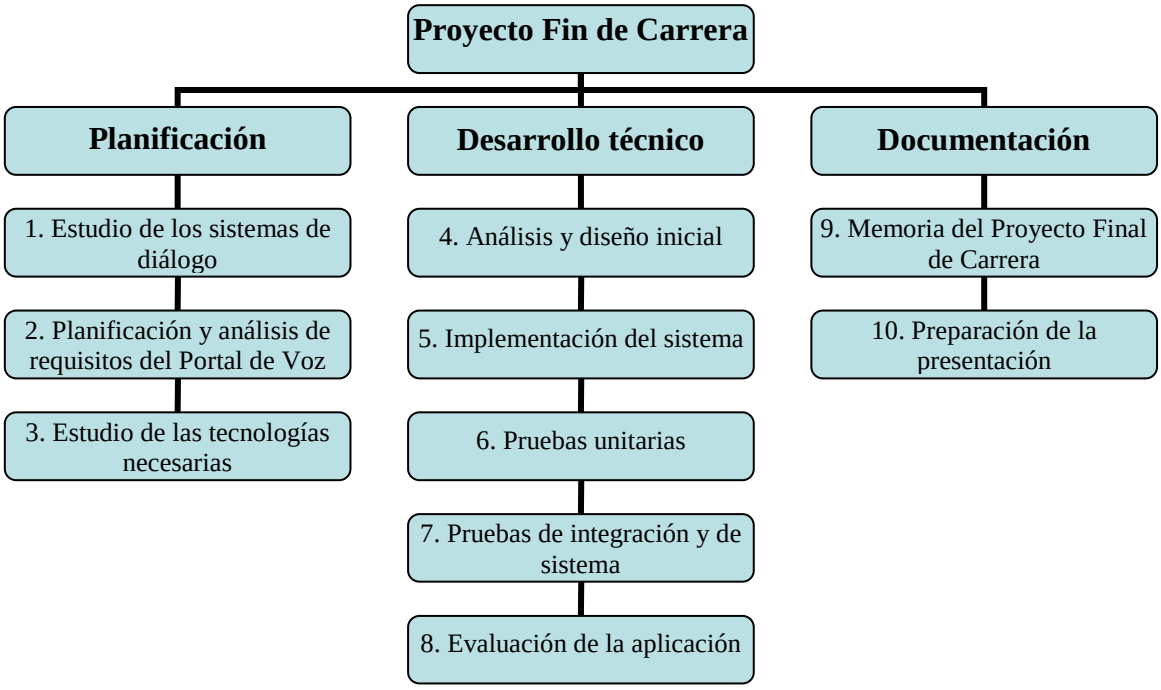


Figura 1.2. Diagrama WBS representando las tareas definidas para el proyecto

1.4 Planificación temporal

Una vez establecidas las fases y tareas en las que se ha dividido el proyecto (Apartado 1.3 - Fases de desarrollo), se ha realizado la planificación temporal de las fases utilizando como herramienta de ayuda un *diagrama de Gantt*. Este diagrama vincula las tareas a un calendario permitiendo de esta forma un seguimiento detallado del estado de avance de cada tarea.

Para la realización del *diagrama de Gantt*, además de incluir las 10 tareas divididas en tareas resumen que se han estudiado en el *diagrama WBS*, se han añadido un hito de inicio de proyecto y un hito de fin de proyecto. Los hitos son tareas de duración nula que se utilizan para marcar puntos de control de avance el proyecto.

A continuación se introducen las duraciones de cada tarea, considerando una jornada de trabajo de 8 horas y 7 días laborales a la semana, y las relaciones de precedencia de cada tarea, que fijan la restricción de no empezar una cierta tarea si su tarea precedente no ha finalizado.

El *diagrama de Gantt* resultante de aplicar este procedimiento de planificación se puede observar en la Figura 1.3. Con este diagrama se muestra una visión global de la planificación del proyecto, de sus distintas fases y de su duración.

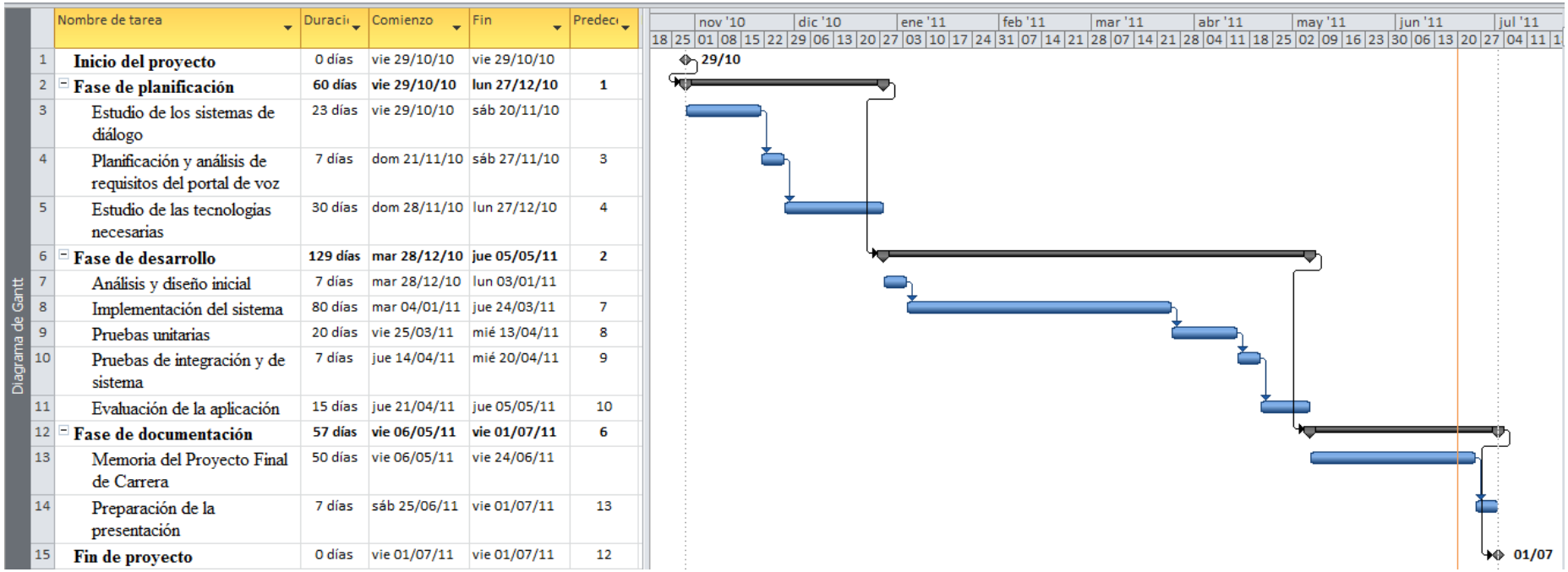


Figura 1.3. Diagrama de Gantt de la planificación temporal del Proyecto Final de Carrera

1.5 Medios y documentación utilizados

Los medios empleados para la realización del Proyecto Final de Carrera han sido los siguientes:

- Dispositivos Hardware:
  - Ordenador portátil.
  - Auriculares con micrófono.
  - Periféricos habituales (teclado, ratón).
  - Servidor web x10Hosting.

- Aplicaciones Software:
  - Plataforma Voxeo.
  - Skype.
  - Paquete Microsoft Office 2007.
  - Editor de programación JVoxEdit.
  - Editor de programación Notepad ++.

En el apartado Presupuesto de esta memoria se adjunta información referente al coste de estos medios.

En cuanto a la documentación examinada, se han empleado numerosos libros y artículos en el ámbito de las interfaces de diálogo y sobre los lenguajes de computación oral, además de diversos manuales de los distintos lenguajes de programación utilizados. Toda esta documentación se encuentra detallada en el apartado de Bibliografía.

1.6 Estructura de la memoria

Se incluye a continuación un breve resumen del contenido de cada capítulo de la memoria, con el objetivo de facilitar la lectura de la memoria:

Capítulo 1: Introducción y objetivos. Este capítulo inicial establece el propósito y los objetivos de todo el contenido del Proyecto Final de Carrera. Además, incluye las fases de desarrollo, la planificación temporal, los medios empleados y la estructura de la memoria.

Capítulo 2: Estado del Arte. En este capítulo se hace un estudio completo de los sistemas de diálogo, se analiza en detalle el lenguaje estándar VoiceXML y se presenta la plataforma utilizada para la implementación de la aplicación, Voxeo.

Capítulo 3: Descripción general de la aplicación. En este capítulo se proporciona una visión global del portal de voz desarrollado, se analizan las tecnologías empleadas y se detallan las operaciones más comunes tratadas a lo largo de su implementación.

Capítulo 4: Descripción detallada de los módulos del sistema. En esta sección se describe detalladamente cada uno de los módulos de los que consta el portal de voz desarrollado para la atención de los ciudadanos: módulo Inicio, módulo Información, módulo Gestiones y Trámites, módulo Encuesta, módulo Buzón del ciudadano y módulo Tele-Operador. Para cada uno de ellos se explica sus funcionalidades, arquitectura y escenarios de uso.

Capítulo 5: Conclusiones y trabajo futuro. Se exponen las principales ideas, cuestiones y conclusiones derivadas de la realización del proyecto, así como las posibles líneas de investigación que a partir de este proyecto se podrían generar.

Capítulo 6: Evaluación de la aplicación. En este capítulo se razona la importancia de la evaluación de los sistemas de diálogo y, a través de las valoraciones subjetivas de los usuarios recogidas mediante un cuestionario, se lleva a cabo la evaluación del portal de voz desarrollado.

Presupuesto. Este apartado contiene un análisis de los costes del diseño y desarrollo del proyecto, detallando el coste de personal y del material necesario para llevar a cabo su realización.

Glosario. En este apartado se recopilan los principales términos y conceptos técnicos utilizados en la memoria, con el objetivo de facilitar su comprensión al lector.

Bibliografía. En este apartado se reflejan las citas bibliográficas que se han consultado para la realización tanto del proyecto como de la memoria.

Capítulo 2

Estado del arte

En este capítulo se analiza el contexto en el que se enmarca este Proyecto Final de Carrera. En primer lugar, se presentan los sistemas del diálogo, explicando los módulos que componen su arquitectura, examinando los distintos criterios para su clasificación y describiendo aplicaciones concretas de sistemas de diálogo existentes en la actualidad. A continuación, se hace un estudio completo del lenguaje estándar de programación VoiceXML, definido como el estándar para el acceso oral a la información en Internet. Finalmente, se presenta de forma detallada la implementación del estándar mediante la plataforma Voxeo: su configuración, su funcionamiento y las herramientas que ofrece. Se concluye el capítulo enumerando el resto de implementaciones más significativas del estándar VoiceXML que existen actualmente.

2.1 Los sistemas de diálogo

2.1.1 Introducción a los sistemas de diálogo

Los sistemas de diálogo basados en procesamiento del habla son sistemas informáticos que reciben como entrada frases del lenguaje natural expresadas de forma oral y generan como salida frases del lenguaje natural expresadas asimismo de forma oral [LOP05].

La finalidad de estos sistemas es emular el comportamiento inteligente de un ser humano en un diálogo con otra persona, con el objetivo de que se realice una cierta tarea [GRI07].

Un sistema de diálogo ideal reconocería el habla espontánea, comprendería enunciados sin restricciones de contenido, proporcionaría respuestas con sentido, gramaticalmente bien formadas y pragmáticamente adecuadas, respondería con voz completamente natural y sería multimodal [LLI06].

Actualmente, los sistemas de diálogo están limitados debido a encontrarse sujetos a las restricciones del reconocimiento automático del habla, estar la comprensión y respuesta limitados a dominios específicos, necesitar estrategias de verificación, estar condicionados por la naturalidad del habla sintetizada y existir el problema del diálogo que surge de forma espontánea.

El abanico de posibilidades en cuanto a entorno y tareas en los que se pueden aplicar estos sistemas es muy amplio, ya que permiten el acceso a servicios y control de máquinas por vía telefónica. La posibilidad de convertir cualquier línea de la red telefónica en un punto de acceso a los servicios les proporciona un gran valor económico y social: banca telefónica, centros de atención de llamadas, etc.

También, los portales de voz representan una vía complementaria al acceso WAP, para unir Internet y la telefonía móvil, dos ámbitos que en los últimos años han experimentado un crecimiento exponencial.

Además, con estos sistemas se posibilita la comunicación en ciertos casos en los que con las interfaces tradicionales como el teclado, el ratón o la pantalla no sería posible. Ejemplos concretos son el acceso a las máquinas (y mediante ellas a la información) de personas que sufren discapacidades visuales o motoras graves, aplicaciones en entornos reducidos o aplicaciones accesibles desde vehículos, donde el conductor no puede utilizar la vista para tareas ajenas a la conducción (marcación de un número de teléfono, control de los elementos accesorios del vehículo, utilización de teleservicios tales como el encaminamiento en ruta, la información sobre tráfico o sobre zonas de aparcamiento, etc.).

2.1.2 Arquitectura de un sistema de diálogo

El esquema utilizado para el desarrollo de sistemas de diálogo suele englobar una serie de módulos genéricos que deben coordinarse para responder a los requerimientos del usuario [LLI06] [HUR+05].

La Figura 2.1 muestra la arquitectura de un sistema de diálogo basado en el procesamiento del habla.

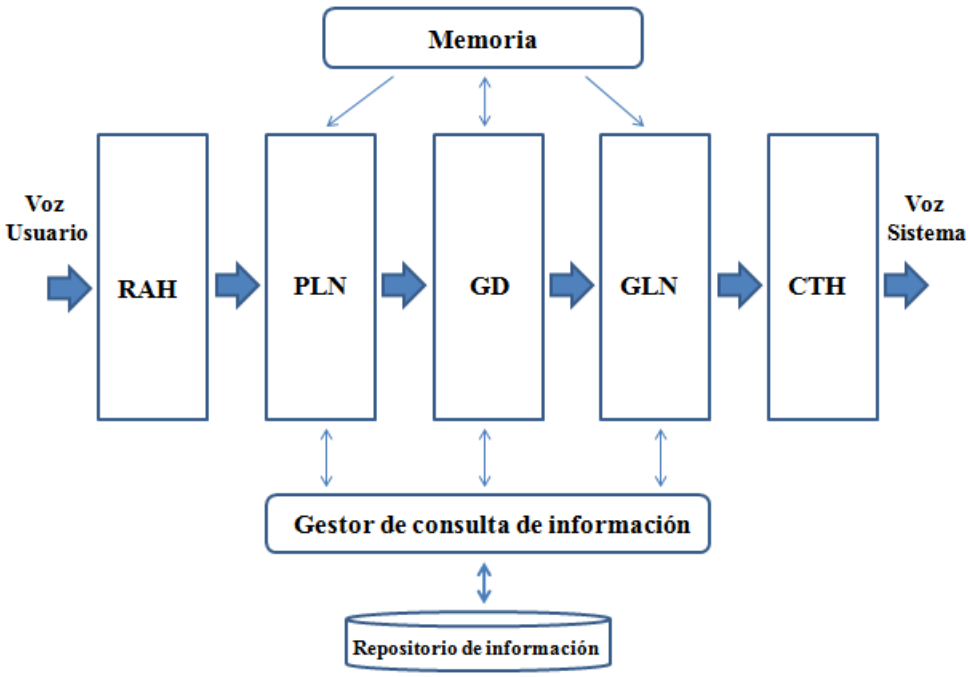


Figura 2.1. Arquitectura modular de un sistema de diálogo

La descripción de los módulos y sus principales características se explican a continuación:

Módulo de Reconocimiento Automático del Habla (RAH): Procesa la voz de usuario y la transforma en la secuencia de palabras reconocidas con mayor probabilidad.

Módulo de Procesamiento del Lenguaje Natural (PLN): Extrae el significado de las palabras reconocidas en el módulo anterior, obteniendo la representación semántica (significado) de la frase y expresándolo en términos de un lenguaje especificado para la tarea.

Gestor del diálogo (GD): Decide qué paso debe dar el sistema tras cada intervención del usuario. Puede decirse que este es el módulo fundamental del sistema, pues su finalidad es lograr que la interacción con el usuario sea lo más cómoda e "inteligente" posible. Para ello, se basa en la interpretación semántica de la petición del usuario, la historia del proceso de diálogo, la información disponible en ese punto, el estado actual del sistema, la información obtenida de la base de datos, la estrategia definida, etc.

Módulo de Generación del Lenguaje Natural (GLN): Tiene como función la generación de una frase, gramaticalmente correcta y en un lenguaje lo más cercano posible al lenguaje natural, que transmita el mensaje generado por el gestor de diálogo.

Módulo de Conversión Texto Habla (CTH): Transforma la frase de respuesta en señal de audio.

Módulo de memoria: Almacena las representaciones semánticas obtenidas a lo largo de la interacción así como las frases previamente generadas por el sistema, proporcionando esta información histórica a los módulos de procesamiento del lenguaje natural, gestor del diálogo y generación del lenguaje natural. De esta forma, el sistema puede resolver las referencias anafóricas existentes en las frases pronunciadas por los usuarios, puede conocer qué frases ha expresado el usuario previamente, y puede utilizar información contextual (mediante el uso de anáforas y elipsis) durante la generación de las frases.

Módulo gestor de consulta de información de la aplicación: Se encarga de generar las consultas necesarias al repositorio de información, procesarlas y proporcionar la información obtenida al módulo gestor del diálogo.

Repositorio de información: Se trata de bases de datos o páginas web donde se almacena la información necesaria para el sistema de diálogo y que será consultada por el gestor de consulta de información.

2.1.3 Clasificación de los sistemas de diálogo

Los sistemas de diálogo se pueden clasificar según distintos criterios [GRI07]:

- **Iniciativa del diálogo**

Sistemas de diálogo guiados por el sistema: La interacción se realiza mediante turnos concretos y fijados entre pregunta y respuesta. Se restringen las iniciativas del usuario.

Sistemas de iniciativa mixta: Aceptan las interrupciones y negociaciones por parte del usuario, reparten de forma equilibrada el turno de palabra e incorporan mecanismos de detección de incoherencias gramaticales.

Sistemas de diálogo guiados por el usuario: Es el usuario el que lleva la iniciativa de diálogo en cualquier instante de la interacción, permitiéndose al usuario que utilice lenguaje natural y no delimitando los mensajes del sistema (por ejemplo, ¿en qué puedo ayudarle?).

- **Dependencia del hablante**

Dependiente del hablante: Un sistema dependiente del hablante está desarrollado para funcionar con un sólo hablante. Estos sistemas, normalmente, son más fáciles de implementar, más baratos y más precisos, pero no tan flexibles como los sistemas adaptables al hablante o los sistemas independientes del hablante.

Independiente del hablante: Un sistema independiente del hablante está desarrollado para funcionar para cualquier hablante de un determinado tipo (por ejemplo, inglés americano). Estos sistemas son los más complicados de desarrollar, los más caros y la precisión es menor que la de los sistemas dependientes del hablante. Sin embargo, son más flexibles.

- **Tipo de comunicación**

Unimodal (oral): Sólo se utiliza el habla.

Multimodal: Se utilizan varios canales de comunicación. Por ejemplo, dispositivos de entrada como el habla, el teclado, el ratón, el micrófono, la cámara, una pantalla o un PDA y canales de salida para proporcionar información como voz, texto, gráficos o imágenes.

- **Tipo de idioma**

Monolingüe: Permiten interacción sólo mediante un idioma.

Multilingüe: Permiten interacción mediante varios idiomas. Por ejemplo, castellano e inglés.

- **Tipo de discurso**

Reconocimiento de palabras aisladas: En el reconocimiento de palabras aisladas el patrón acústico de cada palabra del vocabulario se almacena como una secuencia temporal de características derivadas, usando LPC, análisis de banco de filtros o alguna otra técnica de análisis del lenguaje. El reconocimiento se lleva a cabo comparando el patrón acústico de la palabra a reconocer con los patrones almacenados, seleccionando la palabra que mejor encaje con la palabra desconocida.

Reconocimiento de palabras conectadas: En el reconocimiento de palabras conectadas, la entrada hablada es una secuencia de palabras de un vocabulario específico, y el reconocimiento se lleva a cabo basándose en la coincidencia de palabras de referencia aisladas. Ejemplos de esto son las cadenas de dígitos conectados donde el vocabulario es un conjunto de 10 dígitos, o el reconocimiento de letras conectadas, donde el vocabulario es el conjunto formado por el abecedario. No hay que confundir esto con el reconocimiento continuo de voz, donde el reconocimiento se basa en unidades lingüísticas denominadas fonemas, sílabas, etc…, lo que supone separar la voz en estas unidades y etiquetarlas subsecuentemente.

Reconocimiento continuo: Un sistema de reconocimiento continuo funciona sobre un lenguaje en el que las palabras están conectadas, es decir, no están separadas por pausas. El lenguaje continuo es más difícil de tratar debido a la variedad de efectos. Primero, es difícil encontrar el comienzo y el final de las palabras. Otro problema es la coarticulación. La generación de cada fonema se ve afectada por la generación de los fonemas adyacentes, y de modo parecido, el comienzo y final de las palabras se ven afectados por las palabras que les preceden y suceden. El reconocimiento del lenguaje continuo también se ve condicionado por la frecuencia de habla (un discurso rápido suele ser más difícil).

Reconocimiento discreto: Un sistema de reconocimiento discreto funciona con palabras simples, necesitando de una pausa entre la dicción de cada palabra. Esta es la forma más sencilla de reconocimiento para llevar a cabo ya que los puntos de finalización son más fáciles de encontrar y la pronunciación de una palabra no afecta

a las demás. De este modo, y ya que las ocurrencias de las palabras son más consistentes, es más fácil reconocerlas.

- **Adaptación**

Sistemas adaptativos: El sistema es capaz de aprender nuevas estrategias comunicativas en función del comportamiento del usuario.

Una línea de investigación actual consiste en la incorporación de las emociones a los sistemas de diálogo. Para ello, el análisis del comportamiento del usuario no debe incluir únicamente las emociones básicas sino también todos sus estados emocionales y cambios en el comportamiento. Por ejemplo, si se detecta que el usuario puede estar enfadado, se inician estrategias de recuperación o se transfiere la llamada a un operador humano, o si se detecta algo similar a la alegría, se aprovecha para ofrecer un nuevo producto.

- **Dominio de la aplicación**

Clasificación según la tarea que realiza el sistema. Este criterio se estudiará con detalle en el apartado 2.1.4.

2.1.4 Aplicaciones – Ejemplos

Actualmente existen multitud de aplicaciones de sistemas de diálogo. En este apartado se van a analizar ejemplos de estas aplicaciones de voz clasificándolas según su utilidad, la dificultad de la interacción y las tareas que realizan. De este modo se pueden diferenciar varios niveles de menor a mayor complejidad [Gri07] [Cal08].

Centralita telefónica

Las aplicaciones de voz más sencillas son aquellas en las que se recibe una llamada y se transfiere automáticamente esa llamada a otro número, como es el caso de las centralitas telefónicas que ponen en contacto al usuario con la persona adecuada:

- Atos [ATOS] – Centralita telefónica. España

Consulta de información

Los sistemas de diálogo que se utilizan para obtener información son aplicaciones en las que los datos que se consultan se encuentran en un repositorio de información, como puede ser una base de datos o página web a los que accederá el sistema de diálogo.

Estos sistemas se utilizan en diversos ámbitos, que podemos comprobar en los siguientes ejemplos de aplicación:

- Júpiter [JUPITER] – Predicción meteorológica por teléfono. EE.UU.
- Dihana [DIHANA] – Información de viajes en tren. España.

- Pegasus [PEGASUS] – Información de viajes en avión. EE.UU.
- Conquest [CONQUEST] – Información de horarios en congresos. EE.UU.

Además, hay sistemas que ofrecen la posibilidad de gestionar datos pertenecientes al propio usuario, como sucede en los sistemas que permiten la consulta del correo electrónico a través del teléfono:

- Athosmail [ATHOSMAIL] – Lectura de correos electrónicos utilizando teléfonos móviles. EE.UU.

Reservas

Existen otros sistemas que además de proporcionar información realizan tareas algo más complejas, como son reservas a través de la aplicación de voz:

- DARPA Communicator [DARPA] – Interfaz conversacional inteligente para la reserva de billetes de avión. EE.UU.
- Mercury [MERCURY] – Interfaz conversacional que proporciona información sobre vuelos y permite hacer reservas. EE.UU.

Transacciones

También los sistemas conversacionales pueden llevar a cabo transacciones. Son las aplicaciones relacionadas con el comercio electrónico, como la venta de entradas o de billetes de tren y de avión, y los relacionados con la banca electrónica:

- Cineentradas [CINEENTRADAS] – Venta telefónica de entradas de cine. España.
- Línea BBVA [BBVA] – Sistema que permite solicitar transferencias o domiciliaciones, vender o comprar valores, contratar seguros o realizar aportaciones a planes de pensiones o fondos de inversión. Se comprueba la identidad del cliente mediante una clave personal de acceso. España.

Control remoto

La interacción oral también es útil para el control de dispositivos, especialmente en entornos inteligentes. Un ejemplo de este tipo de sistemas es:

- MIMUS [MIMUS] – Sistema de diálogo multimodal para controlar un hogar inteligente. España.

Tutorización

Los sistemas de diálogo se emplean también en el ámbito de la educación, particularmente con el fin de mejorar habilidades fonéticas y lingüísticas de los usuarios:

- LISTEN [LISTEN] – Tutor de lectura. España.
- VOCALIZA [VOCALIZA] – Sistema para terapias de voz. España.
- ITSPOKE [ITSPOKE] – Tutor para dar información y corregir errores de aprendizaje. EE.UU.

Robótica

Los sistemas de diálogo se integran en las aplicaciones del mundo de la robótica, por ejemplo:

- Cogniron [COGNIRON] – Robot cognitivo. Europa

Compañeros virtuales

Los agentes virtuales y compañeros constituyen la aplicación más compleja de los sistemas de diálogo:

- Ikea. Pregunta a Anna [IKEA] – Compañera relacionada con la tienda IKEA. España.
- Companions [COMPANIONS] – Compañero personalizado multi-modal. Europa.
- Olga [OLGA] – Compañera 3D animada multi-modal. Suecia.
- Collagen [COLLAGEN] – Asistentes conversacionales y agentes colaborativos. EE.UU.

2.2 Voice Extensible Markup Language (VoiceXML)

2.2.1 Introducción al estándar VoiceXML

El lenguaje VoiceXML (Voice Extensible Markup Language) tuvo sus orígenes en 1995. Fue diseñado como un lenguaje de diálogo basado en XML (eXtensible Markup Language) [XML], que pretendía simplificar el proceso de desarrollo de aplicaciones de reconocimiento de voz dentro de un proyecto de AT&T llamado PML (Phone Markup Language).

En 1998, el W3C (Consorcio World Wide Web) organizó una conferencia sobre navegadores por voz. En este momento, AT&T y Lucent tenían diferentes variantes de su PML original, mientras que Motorola e IBM habían desarrollado sus propios lenguajes VoxML y SpeechML respectivamente. Muchos otros asistentes a la conferencia también estaban diseñando lenguajes parecidos, por ejemplo, TalkML de HP y VoiceHTML de PipeBeach.

Debido a esta situación, AT&T, IBM, Lucent y Motorola decidieron formar el Foro de VoiceXML para aunar sus esfuerzos. La misión del Foro VoiceXML era definir un lenguaje estándar que los desarrolladores pudieran utilizar para crear aplicaciones de voz. Eligieron XML como base para ello.

En 2000, el Foro de VoiceXML lanzó VoiceXML 1.0 al público. Poco después, VoiceXML 1.0 se presentó al W3C como base para la creación de un nuevo estándar internacional.

Finalmente, VoiceXML 2.0 es el resultado de ese trabajo basado en la colaboración conjunta de las empresas miembros de W3C, otros grupos de trabajo del W3C y el público. La versión 2.1 aparece como recomendación del W3C en junio de 2007. El working draft de VoiceXML 3.0 fue aprobado en diciembre de 2010.

VoiceXML es un lenguaje diseñado para crear sistemas de diálogo mediante voz que incluye reconocimiento de entradas de voz y de entradas DTMF, funciones de telefonía, salida de voz sintetizada y ficheros de audio, grabación de diálogos y control del flujo del diálogo.

Su objetivo es aprovechar las ventajas de las tecnologías Web para el desarrollo de aplicaciones controladas mediante la voz.

Un sencillo ejemplo de código VoiceXML se muestra en la Figura 2.2.

```
<?xml version="1.0" xml:lang="es-ES" encoding="UTF-8"?>

      <vxml version="2.0">
      <form>
            <block>
                  <prompt>
                        ¡Hola Mundo!
                  </prompt>
            </block>
      </form>
      </vxml>
```

Figura 2.2. Ejemplo “Hola Mundo”

Este ejemplo consta de un formulario simple con un bloque que sintetiza y reproduce al usuario la frase “¡Hola Mundo!”.

La arquitectura definida para una aplicación desarrollada en VoiceXML contiene los componentes que se muestran en la Figura 2.3.

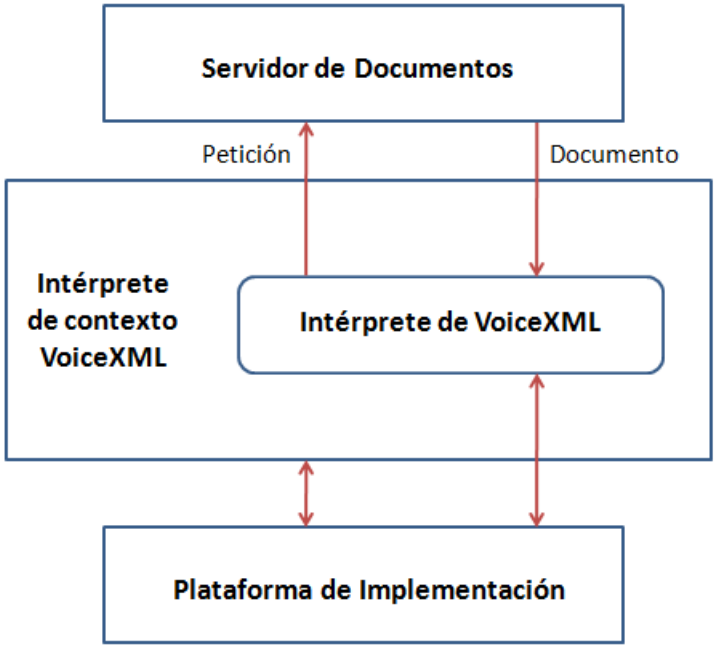


Figura 2.3. Modelo de arquitectura VoiceXML

El servidor de documentos procesa las peticiones del intérprete de VoiceXML a través del intérprete de contexto VoiceXML. En respuesta, el servidor de documentos produce documentos VoiceXML que son procesados por el intérprete de VoiceXML. El intérprete de contexto VoiceXML puede monitorizar entradas de usuario en paralelo con el intérprete de VoiceXML.

La plataforma de implementación está controlada por el intérprete de contexto VoiceXML y por el intérprete de VoiceXML. La plataforma de implementación genera eventos en respuesta a las acciones del usuario y a los eventos del sistema. Algunos de estos eventos son gestionados por el intérprete de VoiceXML, mientras que otros son gestionados por el intérprete de contexto VoiceXML.

El objetivo principal de VoiceXML es dar toda la potencia del desarrollo Web y de la entrega de contenido a los sistemas de diálogo y, de esta forma, liberar a los autores de estas aplicaciones de la programación a bajo nivel y de la administración de recursos.

VoiceXML permite la integración de servicios de voz con servicios de datos mediante el modelo cliente-servidor. Un servicio de voz es visto como una secuencia de diálogos de interacción entre un usuario y una plataforma de implementación. Los diálogos son proporcionados por el servidor de documentos, que puede ser externo a la plataforma de implementación.

El servidor de documentos se encarga de mantener una completa lógica del servicio, el funcionamiento de las bases de datos y las operaciones legales del sistema, además de producir los diálogos.

La entrada de usuario afecta a la interpretación del diálogo y es recogida en *requests* enviadas al servidor de documentos. El servidor de documentos responde con documentos VoiceXML para continuar la sesión del usuario con otros diálogos.

2.2.2 Conceptos básicos de VoiceXML

Un documento VoiceXML puede considerarse como una máquina de estados finitos. El usuario está siempre en un estado, o diálogo, en un determinado momento. Cada diálogo determina el siguiente diálogo al que hacer la transición. Las transiciones se especifican mediante URIs (Uniform Resource Identifier) [URI, IETF 1998], que definen el siguiente documento y diálogo a utilizar.

La ejecución se termina cuando un diálogo no especifica un sucesor, o si tiene un elemento que hace salir explícitamente de la conversación.

Los conceptos básicos referentes a VoiceXML son los siguientes:

- **Diálogos y subdiálogos**

Hay dos tipos de **diálogos**: formularios y menús.

-Los **formularios** definen una interacción que recoge los valores de un conjunto de variables. Cada campo puede especificar una gramática que define las entradas permitidas para ese campo.

-Un **menú** presenta al usuario una amplia gama de opciones y, a continuación, transiciones a otro diálogo basado en la elección elegida.

Un **subdiálogo** es como una llamada a función, que proporciona un mecanismo para invocar una nueva interacción y volver al formulario original. Las instancias de variables, las gramáticas y la información de estado se guardan y están disponibles al volver al documento inicial.

Cuando los subdiálogos son invocados añaden un nuevo contexto de ejecución. El subdiálogo puede ser un nuevo diálogo dentro del documento existente o un nuevo diálogo dentro de un nuevo documento. Los subdiálogos pueden estar compuestos de varios documentos.

- **Sesión**

Una sesión comienza cuando el usuario comienza a interactuar con el intérprete de contexto VoiceXML, continúa mientras los documentos son cargados y procesados, y termina cuando se solicita por el usuario, por un documento o por el intérprete de contexto VoiceXML.

- **Aplicación**

Una aplicación es un conjunto de documentos que comparten el mismo documento raíz de la aplicación. Siempre que el usuario interactúa con un documento en una aplicación, el documento raíz de la aplicación se carga también y permanece cargado

mientras el usuario se encuentra en transición entre documentos dentro de la misma aplicación. Finalmente se descarga cuando el usuario pasa a un documento que no está en la aplicación.

Cuando una aplicación tiene un solo documento, la ejecución del documento comienza por defecto en el primer diálogo. Según se ejecuta cada diálogo, se determina el siguiente diálogo. Cuando el diálogo no especifica un diálogo sucesor, se detiene la ejecución del documento.

Si una aplicación tiene varios documentos, cada documento se ejecuta como una aplicación aislada. En los casos en los que se quiera tener varios documentos para trabajar conjuntamente como una única aplicación, se debe seleccionar un documento como documento raíz de la aplicación y el resto como documentos hoja. Cada documento hoja nombra al documento raíz en su elemento <vxml>.

Cada vez que el intérprete tenga que cargar y ejecutar un documento hoja, primero cargará el documento raíz de la aplicación si no está ya cargado. El documento raíz de la aplicación permanecerá cargado hasta que el intérprete tenga que cargar un documento que pertenezca a otra aplicación. Por lo tanto siempre se mantendrá una de las dos condiciones siguientes durante la interpretación:

- El documento raíz de la aplicación se carga y el usuario está ejecutando en él. No hay ningún documento hoja.
- El documento raíz de la aplicación y un sólo documento hoja son cargados. El usuario está ejecutando en el documento hoja.

Si hay una cadena de subdiálogos definida en documentos separados, puede haber más de un documento hoja cargado aunque la ejecución se realizará sólo en uno de estos documentos.

Cuando un documento hoja hace que se cargue un documento raíz, ninguno de los diálogos en el documento raíz se ejecutan. La ejecución comienza en el documento hoja.

Una de las ventajas de las aplicaciones multidocumento es que las variables y las propiedades del documento raíz están disponibles para su uso por los documentos hoja, de esta forma la información se puede compartir y conservar.

- **Gramáticas**

Cada diálogo puede tener asociadas una o más gramáticas de voz o de DTMF.

En las **aplicaciones dirigidas por ordenador**, las gramáticas están activas sólo cuando el usuario está en ese diálogo.

En las **aplicaciones de iniciativa mixta**, donde el usuario y la máquina se alternan en la determinación de qué hacer a continuación, algunos de los diálogos están marcados para hacer sus gramáticas activas incluso cuando el usuario se encuentra en

otro diálogo en el mismo documento, o en otro documento cargado en la misma aplicación. En esta situación, si el usuario dice algo que coincida con las gramáticas activas del otro diálogo, se ejecuta la transición a ese otro diálogo. La iniciativa mixta agrega flexibilidad y potencia para las aplicaciones de voz.

- **Eventos**

VoiceXML proporciona un mecanismo de rellenado de formularios para el manejo de entradas de usuario "normales" y define un mecanismo para controlar los eventos no cubiertos por este procedimiento.

Los eventos son lanzados por la plataforma debido a una gran variedad de circunstancias, como por ejemplo, cuando el usuario no responde, no responde correctamente, solicita ayuda, etc. El intérprete también produce eventos si encuentra un error semántico en un documento de VoiceXML. El control de eventos comunes puede especificarse en cualquier nivel, y se aplica a todos los niveles inferiores.

- **Enlaces**

Los enlaces especifican transiciones a otros puntos del documento, otro documento dentro de la aplicación, u otro documento de otra aplicación.

2.2.3 Elementos y atributos de VoiceXML

En la Tabla 2.1 se describen los elementos de VoiceXML:

Elemento	Funcionalidad
<assign>	Asigna un valor a una variable.
<audio>	Reproduce un clip de audio dentro de un mensaje.
<block>	Contenedor (no interactivo) de código ejecutable.
<catch>	Captura un evento.
<choice>	Define un elemento (ítem) del menú.
<clear>	Borra uno o más ítems del formulario.
<disconnect>	Desconecta una sesión.
<else>	Elemento utilizado en la estructura <if>.
<elseif>	Elemento utilizado en la estructura <if>.

Elemento	Funcionalidad
<enumerate>	Notación abreviada para enumerar las opciones en un menú.
<error>	Captura un evento error.
<exit>	Finaliza la sesión.
<field>	Declara un campo de entrada en un formulario.
<filled>	Acción a ejecutar cuando se rellenan los campos.
<form>	Cuadro de diálogo para presentar información y recopilar datos.
<goto>	Transición a otro diálogo en el mismo o diferente documento.
<grammar>	Especifica un reconocimiento de voz o una gramática DTMF.
<help>	Captura un evento ayuda.
<if>	Lógica condicional.
<initial>	Declara código inicial antes de entrar en un formulario (iniciativa mixta).
<link>	Especifica una transición válida para todos los diálogos en el ámbito de la aplicación.
<log>	Genera un mensaje de depuración.
<menu>	Cuadro de diálogo para elegir entre varias alternativas.
<meta>	Define un elemento de metadata en formato nombre/valor.
<metadata>	Define información metadata mediante el esquema metadata.
<noinput>	Captura el evento *noinput*.
<nomatch>	Captura el evento *nomatch*.
<object>	Define extensiones a medida.
<option>	Especifica una opción en un <field>.
<param>	Parámetro en <object> o <subdialog>.
<prompt>	Salida de audio para el usuario.
<property>	Configuración de la plataforma de control de la aplicación.
<record>	Graba una muestra de audio.

Elemento	Funcionalidad
<reprompt>	Reproduce un *prompt* cuando un campo es revisitado después de un evento.
<return>	Retorno desde un sub-diálogo.
<script>	Especifica un bloque de código ECMAScript.
<subdialog>	Invoca a un diálogo como un subdiálogo del actual.
<submit>	Presenta valores a otro documento del servidor.
<throw>	Lanza un evento.
<transfer>	Transfiere la llamada a otro destino.
<value>	Inserta el valor de una expresión en un *prompt*.
<var>	Declara una variable.
<vxml>	Elemento de mayor jerarquía en cada documento VoiceXML.

Tabla 2.1: Elementos VoiceXML

En la Tabla 2.2 se explican los atributos del elemento <vxml>.

Atributo	Descripción
version	Versión VoiceXML del documento (obligatorio).
xmlns	Espacio de nombres designado para VoiceXML (obligatorio).
xml:base	URI base para el documento. Todas las referencias relativas dentro del documento lo toman como base.
xml: lang	Identificador de idioma del documento. Si se omite, el valor es un valor predeterminado específico de la plataforma. Se hereda hacia niveles inferiores en la jerarquía del documento.
application	URI del documento raíz de la aplicación.

Tabla 2.2. Atributos Vxml

2.2.4 Constructores del diálogo

2.2.4.1 Formularios

Los formularios son diálogos para presentar información y recoger datos. Los atributos de los formularios se muestran en la Tabla 2.3.

Atributo	Descripción
id	Nombre del formulario. Si se especifica, el formulario se puede referenciar dentro del documento o de otro documento.
scope	Alcance por defecto de las gramáticas del formulario.

Tabla 2.3. Atributos de <form>

- **Interpretación de los formularios**

Los formularios son interpretados por un algoritmo de interpretación de formularios implícito (FIA, Form Interpretation Algorithm). El FIA tiene un bucle principal que selecciona uno a uno los ítems del formulario y los va visitando. El ítem seleccionado es el primero en el documento cuya condición de guarda no es satisfecha.

El FIA finaliza cuando interpreta una sentencia de transferencia de control o con un <exit> implícito al no quedar ítems que seleccionar en el formulario.

- **Ítems de formulario**

Los ítems de formulario son los elementos que pueden ser visitados en el bucle principal del algoritmo de interpretación. Hay dos tipos:

- **Ítems de entrada:** Especifican una variable a recoger del usuario. Estos ítems tienen *prompts* para decirle al usuario qué decir o qué teclear, gramáticas que definen las entradas permitidas y manejadores de eventos que procesan cualquier evento resultante. También pueden tener un elemento <filled> que define la acción a realizar justo después de que la variable del ítem de entrada se haya rellenado. Ítems de entrada son: <*field*>, <*record*>, <*transfer*>, <*object*> y <*subdialog*>.

- **Ítems de control:** Contienen un bloque de código a ejecutar (<*block*>), o controlan la interacción inicial en un formulario de iniciativa mixta (<*initial*>).

- **Variables y condiciones de los ítems de formulario**

Cada ítem del formulario tiene asociado una variable, la cual por defecto está inicializada a *undefined* cuando se entra en el formulario, y una condición de guarda, la cual gobierna si ese ítem puede ser elegido o no por el FIA. Normalmente, a los ítems de entrada se les da nombre y a los de control no. Todos los ítems de formulario tienen los atributos mostrados en la Tabla 2.4.

Atributos	Descripción
name	Nombre de la variable del ítem del formulario con ámbito de diálogo que contendrá el valor del ítem del formulario.
expr	El valor inicial de la variable del ítem del formulario. Su valor por defecto es *undefined*. Si se ha inicializado con un valor, entonces el ítem del formulario no se ejecutará a menos que el valor de esta variable se borre.
cond	Expresión a evaluar junto con la variable del ítem del formulario. Si no se especifica, por defecto será true.

Tabla 2.4. Atributos de los ítems de formulario

- **Formularios dirigidos**

Los formularios dirigidos son los más simples y comunes de los formularios. Los ítems son ejecutados uno por uno en orden secuencial para implementar una interacción dirigida por el ordenador. Las gramáticas sólo están activas en el estado visitado.

En la Figura 2.4 se muestra un ejemplo de este tipo de formularios.

```
<?xml version="1.0"?>
<vxml version="2.0">
      <form id="Numero">
            <field name="num" type ="number">
                  <prompt>Diga un número.</prompt>
            </field>
            <filled>
                  <block>
                  <prompt>
                  Ha dicho usted el <value expr = "num"/>
                  </prompt>
                  </block>
            </filled>
      </form>
</vxml>
```

Figura 2.4. Ejemplo de formulario dirigido

En el ejemplo anterior, el sistema pregunta al usuario por un número y después de que éste lo dice, el sistema repite el número facilitado.

- **Formularios de iniciativa mixta**

En los formularios de iniciativa mixta, ambos, el ordenador y el humano dirigen la conversación. Deben tener uno o más niveles de gramáticas, definiendo unas gramáticas globales para el formulario y otras específicas para cada parte a rellenar del formulario.

Una estructura común combina un elemento <initial>, que solicita una respuesta general, con un elemento <field>, que solicita una información específica.

En la Figura 2.5 se presenta un ejemplo de formulario de iniciativa mixta en el que se solicitan al usuario los valores de provincia y ciudad para la que se desea conocer el tiempo.

```
<form id="informacionMeteorologica">

      <grammar src="ciudadesYprovincias.grxml"/>
      <block>
            <prompt bargein="false">
            Bienvenido al servicio de información meteorológica
            </prompt>
      </block>
      <initial name="inicio">
            <prompt>
            ¿Para qué ciudad y provincia desea conocer el
            tiempo?
            </prompt>
            <noinput>
                  <assign name="inicio" expr="true"/>
            </noinput>
      </initial>
      <field name="provincia">
            <prompt>
                  ¿qué provincia?
            </prompt>
      </field>
      <field name="ciudad">
            <prompt>
                  Por favor diga la ciudad en
                  <value expr="provincia"/>
                  para la que desea conocer el tiempo
            </prompt>
      <filled>
            <submit next="/tiempo" namelist="ciudad provincia"/>
      </filled>
</field>
```

Figura 2.5. Ejemplo de formulario de iniciativa mixta

Se puede controlar el orden de la recogida de campos de diversas formas:

- Asignar un valor a la variable del ítem de tal forma que este ítem no sea seleccionado por el FIA.

- Usar <clear> para poner la variable del ítem a *undefined*, así se fuerza al FIA a revisitar este ítem de nuevo.

- Especificar de forma explícita el siguiente ítem a visitar usando <goto nextitem>. Con este método se fuerza una transferencia inmediata al ítem indicado.

- **Elemento <*filled*>**

El elemento <filled> especifica una acción a realizar cuando se rellena alguna combinación de elementos de entrada. Puede utilizarse de dos formas:

- Como hijo de un elemento <form>: el elemento <filled> puede utilizarse para realizar acciones que ocurren cuando se llena una combinación de uno o más ítems de entrada.

- Como hijo de un ítem de entrada: si el elemento <filled> aparece dentro de un ítem de entrada se especifica una acción a realizar después de que este ítem de entrada se haya rellenado.

Los atributos de <filled> son los mostrados en la Tabla 2.5.

Atributos	Descripción
mode	Puede tomar dos valores: -*any*: la acción se ejecuta cuando cualquiera de los elementos de entrada especificados se rellena por la última entrada del usuario. -*all* (valor por defecto): la acción se ejecuta cuando todos los elementos de entrada mencionados poseen un valor, y al menos uno se ha rellenado por la última entrada del usuario. Un elemento <filled> en un ítem de entrada no puede especificar un modo. Si se especifica un modo, se produciría un error.badfetch por la plataforma.
namelist	Los ítems de entrada se almacenan en esta lista. - Para un <filled> en un form será la lista de nombres de los ítems de entrada del formulario. - Para un <filled> en un ítem de entrada no se puede especificar una lista de nombres, en este caso, la lista de nombres será el nombre del elemento de entrada. Si se especificara el atributo *namelist*, se produciría un *error.badfetch* por la plataforma.

Tabla 2.5. Atributos de <filled>

- **Elemento <link>**

El uso de elementos <link> es una forma de definir enlaces entre distintos puntos de la aplicación ante la ocurrencia de sucesos concretos.

Un elemento <link> puede tener una o más gramáticas con el alcance del elemento que contiene el <link>. Cuando una de estas gramáticas concuerda, se activa el enlace y se producen cualquiera de las dos acciones siguientes:

- Transición a un nuevo documento o diálogo (como <goto>)
- Lanzamiento de un evento (como <throw>).

Un ejemplo del uso del elemento <link> en el que se produce una transición es el de la Figura 2.6.

```
<link next="#siguiente">

      …

</link >
```

Figura 2.6. Ejemplo de elemento <link>

El elemento <link> puede ser hijo de <vxml>, de <form>, o de los ítems de formulario <field> y <initial>. Un *link* en el nivel de <vxml> dispone de gramáticas que están activas en todo el documento. Un *link* en el nivel de <form> dispone de gramáticas activas mientras el usuario está en ese formulario. Si un documento de raíz de la aplicación tiene un *link* de nivel de documento, sus gramáticas están activas independientemente de qué documento de la aplicación se esté ejecutando.

Conceptualmente, el elemento *link* tiene dos partes: “condición” y “acción”. La "condición" es el contenido del elemento del link, es decir, las gramáticas que deben coincidir para que el link se active. La "acción" se especifica mediante los atributos del elemento, es decir, dónde realizar la transición o qué evento lanzar.

Los atributos de <link> se explican en la Tabla 2.6.

Atributo	Descripción
next	URI a donde ir. Este identificador URI es un documento o un diálogo en el documento actual.
expr	Funciona igual que el atributo *next*, excepto cuando el identificador URI es determinado de forma dinámica mediante la evaluación de la expresión ECMAScript dada.
event	Evento a lanzar cuando el usuario coincide con una de las gramáticas del link.
eventexpr	Expresión ECMAScript para evaluar el nombre del evento a lanzar cuando el usuario coincide con una de las gramáticas del link.
message	Cadena de mensaje que proporciona contexto adicional sobre el evento que se produce.
messageexpr	Expresión ECMAScript para evaluar la cadena de mensaje.
dtmf	Secuencia DTMF para el enlace. A diferencia de las gramáticas DTMF, el espacio en blanco es opcional: dtmf = "# 123" es equivalente a dtmf = "1 2 3 #".

Tabla 2.6. Atributos de <link>

Se debe especificar uno de los siguientes atributos: "next", "expr", "event" o "eventexpr"; de lo contrario, se produce un evento *error.badfetch*.

2.2.4.2 Menús

Un menú es una estructura con un solo campo que solicita hacer una elección al usuario y, dependiendo de esta elección, hace transiciones a diferentes lugares. Puede tener su gramática activa cuando el usuario esté ejecutando otro diálogo.

Un ejemplo genérico de un elemento *menu* es el mostrado en la Figura 2.7.

```
< menu id="opciones" dtmf="true">

      <choice next="uril"/>
      <choice next="uri2"/>
</menu>
```

Figura 2.7. Ejemplo genérico del elemento <menu>

- **Elemento *menu***

Los elementos del menú le identifican y determinan el alcance de sus gramáticas. Los atributos de estos elementos se describen en la Tabla 2.7.

Atributos	Descripción
id	Identificador del menú. Permite que el menú sea el destino de un <goto> o un <submit>.
scope	Ámbito de aplicación de la gramática del menú.
dtmf	Cuando se establece a *true*, a las primeras nueve alternativas que no han especificado explícitamente un valor para el atributo *dtmf* se les dan los valores implícitos "1", "2", etc. El valor por defecto es *false*.
accept	Cuando se establece a "*exact*" (valor por defecto), el texto de los elementos de elección en el menú define la frase exacta a ser reconocida. Cuando se establece a "*approximate*", el texto de los elementos de elección define una frase aproximada de reconocimiento.

Tabla 2.7. Atributos <menu>

- **Elemento *choice***

El elemento <choice> tiene varias posibilidades:

- Puede especificar una gramática de voz, definida automáticamente o mediante un elemento <grammar>.
- Puede especificar una gramática DTMF.
- El contenido puede utilizarse para formar la cadena de solicitud en el <enumerate>.

- Puede especificar un evento a lanzar o el URI a donde ir cuando se selecciona una opción.

Los atributos del elemento *choice* son los explicados en la Tabla 2.8.

Atributos	Descripción
dtmf	Secuencia DTMF para esta elección. Es equivalente a una gramática DTMF. A diferencia de las gramáticas DTMF, el espacio en blanco es opcional: dtmf = "#123" es equivalente a dtmf = "1 2 3 #".
accept	Anula la configuración de accept en <menu> para esta elección particular. Cuando se establece a "exact" (valor por defecto), el texto del elemento *choice* define la frase exacta para ser reconocida. Cuando se establece a "aproximate", el texto del elemento *choice* define una frase aproximada de reconocimiento.
next	URI del siguiente cuadro diálogo o documento.
expr	Especifica una expresión a evaluar como URI para la transición.
event	Especifica un evento que será lanzado.
eventexpr	Expresión ECMAScript que evalúa el nombre del evento que se producirá.
message	Cadena de mensaje que proporciona contexto adicional sobre el evento que será lanzado.
messageexpr	Expresión de *ECMAScript* para evaluar la cadena del mensaje.

Tabla 2.8. Atributos <choice>

Se debe especificar uno de los siguientes atributos: "next", "expr", "event" o "eventexpr", de lo contrario, se produce un evento error.badfetch.

Cualquier frase *choice* especifica un conjunto de palabras y frases para escuchar por el usuario. Una frase *choice* está construida con los elementos PCDATA (Parsed Character DATA) contenidos directamente o indirectamente en un elemento <choice> de un <menu>, o en el elemento <option> de un <field>.

Si el atributo *accept* es "exact", el usuario debe decir la frase completa en el mismo orden en que tiene lugar en la frase *choice.*

Si el atributo *accept* es "aproximate", la elección también puede coincidir cuando un usuario diga una subfrase de la expresión.

- **Elemento *enumerate***

El elemento <enumerate> es una descripción generada automáticamente de las opciones disponibles que tiene el usuario. Este elemento especifica una plantilla que se aplica a cada elección en el orden en que aparecen en el menú:

- Si se utiliza sin contenido, se utiliza una plantilla predeterminada por el Intérprete de contexto VoiceXML que enumera todas las opciones.

- Si se utiliza con contenido, el contenido es el especificado en la plantilla. Este especificador puede referirse a dos variables especiales: *_prompt* (prompt elegido) y *_dtmf* (representación normalizada de la secuencia DTMF elegida).

En la Figura 2.8 se presenta un ejemplo del uso de este elemento.

```
menu dtmf="true">

      <property name="inputmodes" value="dtmf"/>
      <prompt> Bienvenido.
            <enumerate>Para
                  <value expr="_prompt"/>, pulse
                  <value expr="_dtmf"/>
            </enumerate>
      </prompt>
      <choice next="#menu">Volver al menú principal</choice>
      <choice next="#salir">Salir</choice>
</menu>
```

Figura 2.8. Ejemplo <enumerate>

El elemento <enumerate> puede utilizarse dentro de los *prompts* y los elementos *catch* asociados con elementos <menu>, y con elementos <field> que contienen elementos <option>. Si se utiliza en cualquier otro lugar se produce un evento *error.semantic*. Por ejemplo, se produciría error si se usa <enumerate> dentro de otro <enumerate>.

2.2.5 Entrada de usuario. Gramáticas

2.2.5.1 Gramáticas de voz

El elemento <grammar> se utiliza para proporcionar una gramática de voz que:

- Especifica el conjunto de palabras que el usuario debe pronunciar para ejecutar una acción o facilitar información.
- Si se trata de una secuencia coincidente, devuelve la correspondiente interpretación semántica.

La etiqueta <grammar> está diseñada para adaptarse a cualquier formato de gramática que cumpla estos dos requisitos.

El W3C define el Speech Recognition Grammar Specification (SRGS) [SGRS] como estándar para el formato de las gramáticas que deben admitir todos los intérpretes de VoiceXML. El formato en el que se encuentren definidas las gramáticas será específico de cada plataforma. Las plataformas VoiceXML deben admitir el formato *XML Form* de la *W3C SRGS* y deberían soportar el *Augmented BNF (ABNF) Form* de la *W3C SRGS*.

Los siguientes elementos de la Tabla 2.9 se definen en el *XML Form* de la *W3C SRGS* y están disponibles en VoiceXML 2.0.

Elemento	Funcionalidad
<grammar>	Elemento raíz de una gramática XML.
<meta>	Declaración del encabezado del contenido meta de un HTTP equivalente.
<metadata>	Declaración del encabezado del contenido de metadatos XML.
<lexicon>	Declaración del encabezado de un léxico de pronunciación.
<rule>	Declara una expansión de regla con nombre para una gramática.
<token>	Define una palabra u otra entidad que puede servir de entrada.
<ruleref>	Hace referencia a una regla definida localmente o externamente.
<item>	Define una expansión con repetición y probabilidad opcional.
< one-of >	Define un conjunto de expansiones de reglas alternativas.
<example>	Elemento contenido dentro de una definición de la regla que proporciona un ejemplo de entrada que coincide con la regla.
<tag>	Define una cadena arbitraria que es incluida en línea en una expansión que puede utilizarse para la interpretación semántica.

Tabla 2.9. Elementos SRGS (*XML Form*)

2.2.5.2 Gramáticas DTMF

El elemento <grammar> puede utilizarse para definir una gramática DTMF que:

- especifica el conjunto de pulsaciones de teclas que un usuario puede utilizar para realizar una acción o facilitar información.
- si coincide con la entrada DTMF, devuelve la interpretación semántica correspondiente.

Para trabajar con este tipo de gramáticas se necesita que las plataformas VoiceXML admitan el formato XML de gramática DTMF.

Una gramática DTMF se distingue de una gramática de voz por el atributo "*mode"* del elemento <grammar>. Las gramáticas DTMF y las gramáticas de voz pueden ser gramáticas de línea o externas y los atributos "type" y "scope" se utilizan de igual forma para ambas.

El ejemplo de la Figura 2.9 es una gramática XML DTMF en línea que acepta como entrada "1 2 3" o "#".

```
<grammar mode="dtmf" version="1.0" root="root">
       <rule id="root" scope="public">
            <one-of>
                  <item> 1 2 3 </item>
                  <item> # </item>
            </one-of>
      </rule>
</grammar>
```

Figura 2.9. Ejemplo gramática XML DTMF

2.2.5.3 Ámbito de las gramáticas

Los ámbitos de las gramáticas dependen del tipo de gramática:

-Las **gramáticas de ítem de entrada** están activas sólo cuando ese ítem de entrada es elegido en la fase de selección del FIA.

-Las **gramáticas de link** tienen el alcance del elemento que contiene el link.

-Las **gramáticas de formulario** tienen por defecto el alcance del diálogo, de tal forma que se activan sólo cuando el usuario está en el formulario.

-Las **gramáticas menú** tienen por defecto el alcance del diálogo y se activan sólo cuando el usuario está en el menú.

A veces se necesita que un formulario tenga algunas gramáticas activas en todo el documento y otras gramáticas activas sólo en el formulario. Una razón para hacer esto es reducir los problemas de superposición de gramáticas. Para implementarlo, a cada elemento individual <grammar> se le puede dar su propio ámbito si este ámbito debe ser diferente que el alcance del propio elemento <form>.

Una gramática en línea se especifica por el contenido del elemento <grammar> y define una gramática entera, como se puede observar en la Figura 2.10.

```
<grammar type="media-type" mode="voice">

gramática de voz en línea …
</grammar>
```

Figura 2.10. Ejemplo de gramática en línea

Para gramáticas en línea, el parámetro *type* especifica el tipo de medio que rige la interpretación del contenido del elemento <grammar>. En la Figura 2.11 se muestra un ejemplo de gramática en línea definida según el formato *XML Form* de la *W3C SRGS*.

```
<grammar mode="voice" xml:lang="en-US" version="1.0"
root="command">

      <!-- Command is an action on an object -->
      <!-- e.g. "open a window" -->
      <rule id="command" scope="public">
            <ruleref uri="#action"/> <ruleref uri="#object"/>
      </rule>
      <rule id="action">
            <one-of>
                  <item> open </item>
                        <item> close </item>
                  <item> delete </item>
                        <item> move </item>
            </one-of>
      </rule>
      <rule id="object">
             <item repeat="0-1">
                        <one-of> <item> the </item> <item> a
            </item> </one-of>
            </item>
            <one-of>
                        <item> window </item>
                        <item> file </item>
                        <item> menu </item>
             </one-of>
      </rule>
</grammar>
```

Figura 2.11. Ejemplo de gramática en línea formato XML

En la Figura 2.12 se muestra el ejemplo equivalente de la gramática en línea anterior pero definida según el formato *ABNF Form* de la *W3C SRGS*:

```
<grammar mode="voice" type="application/srgs">

#ABNF 1.0;
language en-US;
mode voice;
root $command;
      public $command = $action $object;
      $action = open | close | delete | move;
      $object = [the | a] (window | file | menu);
</grammar>
```

Figura 2.12. Ejemplo de gramática en línea formato ABNF

Una gramática externa se especifica mediante un elemento del formulario como se ejemplifica en la Figura 2.13.

```
<grammar src="URI" type="media-type"/>
```

Figura 2.13. Ejemplo de gramática externa

En este caso el parámetro *type* es opcional porque el intérprete de contexto intentará determinarlo de forma dinámica.

Si se define el atributo *src* y hubiera una gramática en línea como contenido del elemento gramática, se produciría un evento *error.badfetch*.

La Figura 2.14 es un ejemplo de una referencia a una gramática externa escrita en el formato XML Form de la W3C SRGS:

```
<grammar type="application/srgs+xml"
src="http://www.grammar.example.com/date.grxml"/>
```

Figura 2.14. Ejemplo de gramática externa formato XML

En la Figura 2.15 se muestra el ejemplo equivalente al anterior pero para una gramática creada utilizando el formato ABNF Form de la W3C SRGS.

```
<grammar type="application/srgs"
src="http://www.grammar.example.com/date.gram"/>
```

Figura 2.15. Ejemplo de gramática externa formato ABNF

- **Jerarquía de las gramáticas a través de pesos**

El peso para la gramática puede ser especificado por el atributo *weight* según se demuestra en la Figura 2.16.

```
<grammar weight="0.6" src="form.grxml"
type="application/srgs+xml"/>
```

Figura 2.16. Ejemplo atributo <weight>

El peso es un valor positivo de punto flotante sin exponenciales. Los formatos permitidos son "n", "n.", ".n" y "n.n", donde "n" es una secuencia de uno o más dígitos.

Un peso de "1.0" es equivalente a no proporcionar ningún peso. Un peso mayor a "1.0" afecta positivamente a la gramática y un peso inferior a "1.0" afecta negativamente a la gramática. Si no se especifica el atributo *weight*, el peso por defecto para cualquier gramática es "1.0".

Un peso no tiene ningún efecto sobre las gramáticas DTMF.

- **Elementos grammar**

Algunos de los atributos de <grammar> son los descritos en la Tabla 2.10.

Atributos	Descripción
src	URI que especifica la ubicación de la gramática y, opcionalmente, si la gramática es externa, un *rulename*.
scope	Si es "*document*", la gramática está activa en todos los diálogos del documento actual. Si es "*dialog*", la gramática está activa a lo largo del formulario actual. Si se omite, la determinación del alcance de la gramática se resuelve mirando el elemento primario.
type	Se puede especificar el tipo de medio de comunicación mediante el atributo *type*. Los tipos de medios de comunicación tentativos para el formato de gramática W3C son "aplicación/srgs + xml" para el formato XML y "aplicación/srgs" para gramáticas ABNF.
weight	Especifica el peso de la gramática.

Tabla 2.10: Atributos <gramar>

Se deben especificar un atributo "*src*" o una gramática en línea, pero no ambos; de lo contrario, se produce un evento error.badfetch.

2.2.5.4 Asignación de los resultados de la interpretación semántica a los formularios VoiceXML

La interpretación semántica devuelta por una gramática SGRS debe asignarse a una o más variables *ECMAScript* de VoiceXML. Este proceso difiere ligeramente según si se trata de resultados de nivel de campo o de nivel de formulario.

Cada ítem de entrada tiene asociado un "*slot name*" que puede utilizarse para extraer parte de la interpretación semántica completa. El "*slot name*" se utiliza durante la fase de procesamiento del FIA para determinar si coincide o no con un elemento de entrada.

Es posible rellenar más de un elemento de entrada para un valor de 'slot' específico si los "slot names" de los elementos de entrada son los mismos.

2.2.6 Salida del sistema. *Prompts*

El elemento <prompt> controla la salida de voz sintetizada y los archivos de audio. Este elemento tiene los atributos descritos en la Tabla 2.11.

Atributos	Descripción
bargein	Controla si un usuario puede interrumpir un *prompt*.
bargeintype	Establece el tipo de *bargein*: 'discurso' o 'hotword'. Se explicará detalladamente en el apartado 4.5.
cond	Expresión que se debe evaluar como '*true*' después de su conversión a *boolean* para que el *prompt* se reproduzca. El valor por defecto es '*true*'.
count	Número que permite emitir diferentes *prompts* si el usuario está haciendo algo repetidamente. Si se omite, por defecto, es "1".
timeout	Tiempo de espera que se utilizará para la siguiente entrada del usuario.
XML: lang	Identificador del idioma para el *prompt*. Si se omite, se utiliza el valor especificado en el atributo de "XML: lang" del documento.
XML: base	Declara el URI base desde el que se resuelven los URIs relativos en el *prompt*.

Tabla 2.11. Atributos de <prompt>

El contenido del elemento <prompt> está definido en el estándar W3C SSML (Speech Synthesis Markup Language) [SSML], que define un lenguaje de marcas para representar los mensajes del sistema mediante la combinación de voz pregrabada y voz sintética.

Los elementos de marcas de voz de la Tabla 2.12 se definen en el estándar SSML y están disponibles en VoiceXML 2.0.

Elemento	Funcionalidad
<audio>	Especifica los archivos de audio y el texto a reproducir.
<break>	Especifica una pausa en la salida de voz.
<desc>	Proporciona en el elemento <audio> una descripción de la fuente de audio que no es voz.
<emphasis>	Indica que el texto adjunto debe ser hablado con énfasis.
<lexicon>	Especifica un léxico de pronunciación para el *prompt*.
<meta>	Especifica las propiedades meta y "http-equiv" para el *prompt*.
<metadata>	Especifica el contenido de metadatos XML para el *prompt*.
<p>	Identifica el texto adjunto como un párrafo que contiene cero o más sentencias
<phoneme>	Especifica una pronunciación fonética para el texto.

Elemento	Funcionalidad
<prosody>	Especifica información prosódica del texto adjunto.
<say-as>	Especifica el tipo de construcción de texto contenido dentro del elemento.
<s>	Identifica el texto adjunto como una frase.
<sub>	Especifica el texto hablado de reemplazo.
<voice>	Especifica las características de la voz para el texto hablado.

Tabla 2.12. Elementos de marcas de voz

VoiceXML permite que los elementos <enumerate> y <value> aparezcan dentro del elemento <prompt>.

La Figura 2.17 muestra diferentes ejemplos de *prompt*.

```
▪ <prompt> Por favor diga su ciudad. </prompt>
▪ <audio src= "diga_su_ciudad.wav"/>
▪ <prompt> Por favor <emphasis>diga</emphasis> su ciudad.</prompt>
```

Figura 2.17. Ejemplos de prompt

Se puede salir de la estructura " <prompt>... </prompt>" si:

- No hay necesidad de especificar atributos adicionales.
- El *prompt* consta exclusivamente de PCDATA (no contiene marcas de voz) o consiste en sólo un elemento <audio> o un elemento <value>.

2.2.6.1 *Prompts* de audio

Los *prompts* pueden consistir en cualquier combinación de archivos pregrabados, flujos de audio o voz sintetizada. Se puede especificar el contenido de audio a través de una URI y también en una variable de audio grabada previamente como se puede comprobar en el ejemplo de la Figura 2.18.

```
<?xml version="1.0"?>
< vxml version="2.0">
      < form id="ejemploAudio">
            < block>
                  <audio src="ficheroGrabado.wav">
            < /block>
      < /form>
< /vxml>
```

Figura 2.18. Ejemplo de *prompt* con audio pregrabado

El elemento de <audio> puede tener contenido alternativo en caso de que la muestra de audio no esté disponible como se observa en el ejemplo de la Figura 2.19.

```
<prompt>
      <audio src= "bienvenido.wav">
      <emphasis> Bienvenido </emphasis> al portal de voz.
      </audio>
</prompt>
```

Figura 2.19. Ejemplo de *prompt* con audio y texto alternativo

En el atributo de <audio> *src* se indica la URI del fichero de audio pregrabada. Esta URI también puede ser especificada a través de una expresión mediante el atributo *expr*. La expresión puede ser una referencia al audio grabado previamente con el ítem <record/> o la evaluación del URI de un recurso de audio.

Se debe especificar uno de los atributos "src" o "expr", de lo contrario, se produce un evento *error.badfetch*.

2.2.6.2 Elemento <value>

El elemento <value> se utiliza para insertar el valor de una expresión en un *prompt*. Su atributo *expr* especifica la expresión a procesar. En la Figura 2.20 se muestra un ejemplo del elemento <value> donde se usa su atributo *expr*.

```
<prompt>
  <value expr="n*n"/> es la raíz cuadrada de <value expr="n"/>.
</prompt>
```

Figura 2.20. Ejemplo del elemento <value>

2.2.6.3 Atributo *bargein*

Si la plataforma de implementación es compatible con *bargein*, el autor de la aplicación decide si un usuario puede interrumpir un *prompt* del sistema usando voz o entrada DTMF. Esto aumenta la velocidad de las conversaciones, pero no se desea siempre.

Si por ejemplo, se necesita que el usuario escuche todo el *prompt* en una advertencia, en un aviso legal o en un anuncio, se debe deshabilitar *bargein* como se ejemplifica en la Figura 2.21.

```
<prompt bargein="false"> <audio src="advertencia.wav"/></prompt>
```

Figura 2.21. Ejemplo atributo *bargein*

Los usuarios pueden interrumpir un *prompt* cuando su atributo *bargein* es *true*, en cambio, deben esperar a la finalización del *prompt* cuando su atributo bargein es *false*.

Cuando *bargein* está activado, puede utilizarse el atributo *bargeintype* para sugerir el tipo de bargein que se llevará a cabo en respuesta a la voz o a la entrada DTMF. Los posibles valores para este atributo son los mostrados en la Tabla 2.13.

Atributo	Descripción
speech	Se detendrá el prompt tan pronto como se detecte voz o entrada DTMF.
hotword	El prompt no se detendrá hasta que se detecte una coincidencia completa de una gramática activa. Las entradas que no coincidan con la gramática serán ignoradas.

Tabla 2.13. Atributos de *bargeintype*

2.2.6.4 Selección de los *prompts*

Los *tapered prompts* son aquellos que pueden cambiar con cada intento. De esta forma, los mensajes de solicitud de información pueden volverse más breves en el supuesto que el usuario se vaya familiarizando con la tarea, los mensajes de ayuda se pueden detallar si el usuario necesita más ayuda, o bien los prompts pueden cambiar con la intención de hacer la interacción más interesante.

El mecanismo de funcionamiento de los *tapered prompts* es el siguiente: cada ítem de entrada, *initial* y *menu* tienen un contador de *prompts* interno que se restablece a uno cada vez que se entra en el formulario o en el menú. Cuando el sistema selecciona uno de estos ítems de entrada, el *prompt* asociado a este ítem incrementará su contador. Un ejemplo del uso de este tipo de *prompts* es el mostrado en la Figura 2.22.

```
<form id="tapered">
  <block>
    <prompt bargein="false">
      Bienvenido al servicio de heladería.
    </prompt>
  </block>
  <field name="sabor">
  <grammar mode="voice" version="1.0" root="root">
    <rule id="root" scope="public">
      <one-of>
        <item>vainilla </item>
        <item>chocolate</item>
        <item>fresa </item>
     </one-of>
    </rule>
   </grammar>
   <prompt count="1"> ¿Cuál es su sabor preferido?</prompt>
   <prompt count="2"> ¿Cuál es su sabor preferido?</prompt>
   <prompt count="3">Diga chocolate, vainilla o fresa.</prompt>
   <help> Lo siento, no hay ayuda disponible.</help>
  </field>
```

```
    </form>
```

Figura 2.22. Ejemplo de *tapered prompts*

2.2.6.5 Atributo *timeout*

El atributo *timeout* especifica el intervalo de silencio permitido mientras se espera por la entrada del usuario después del final del último *prompt*. Si se supera este intervalo, la plataforma produce un evento *noinput*. Cada *prompt* tiene su propio valor de *timeout*.

La razón para permitir que los *timeouts* se especifiquen como atributos de *prompt* es que se puedan utilizar *prompts* escalados. Por ejemplo, pueden darse al usuario ‘x’ segundos para el primer intento de entrada y ‘z’ segundos para el siguiente, como se muestra en el ejemplo de la Figura 2.23.

```
<prompt count="1">
  Por favor, elija un número.
</prompt>
<prompt count="2" timeout="120s">
  Por favor diga un número. Los números posibles son 1, 2 y 3
</prompt>
```

Figura 2.23. Ejemplo de *timeout*

Si hay varios *prompts* esperando un campo de entrada, se utiliza el *timeout* del último *prompt*.

2.2.7 Programación con VoiceXML

2.2.7.1 Definición de variables y expresiones

Las variables de VoiceXML son equivalentes a las variables ECMAScript [ECMASCRIPT]. Las variables de VoiceXML pueden utilizarse en un <script> y las variables definidas en un <script> pueden utilizarse en VoiceXML.

La declaración de una variable utilizando <var> es equivalente a usar una declaración de 'var' en un elemento de <script>. <script> puede aparecer en cualquier lugar donde también pueda aparecer <var>. Las variables de VoiceXML también se declaran en ítems de formulario.

La convención de nombres para variables es como en ECMAScript, pero los nombres que comiencen con el carácter de subrayado ("_") y los nombres que terminen con un signo de dólar ("$") están reservados para uso interno. Las variables de VoiceXML no deben contener palabras reservada de ECMAScript y los nombres deben ser únicos sin incluir puntos. Por ejemplo: "var x.y" es una declaración ilegal en ECMAScript.

Los nombres de las variables que violen las convenciones de nomenclatura o reglas de ECMAScript producen un evento *'error.semantic'*.

- **Declaración de variables**

Las variables se declaran mediante elementos <var> como se puede observar en los ejemplos de la Figura 2.24.

```
▪ <var name="telefono_informacion"/>
▪ <var name="prefijo" expr="91"/>
▪ <var name="ciudad" expr="'Madrid'"/>
```

Figura 2.24. Ejemplos de declaración de variables

Y también se declaran mediante ítems de formulario. Un ejemplo de ello lo podemos ver en la Figura 2.25.

```
<field name="cod_postal">
   <grammar type="application/srgs+xml"
   src="/grammars/codpostal.grxml"/>
   <prompt>¿Cuál es su código postal?</prompt>
</field>
```

Figura 2.25. Ejemplo de declaración de variable mediante ítem de formulario

Las variables deben declararse antes de ser usadas en VoiceXML o en ECMAScript. Las que se declaren sin un valor inicial explícito se inicializan con el valor *undefined* de ECMAScript. El uso de una variable no declarada producirá un *error.semantic*

Las inicializaciones de variables se realizan cuando el algoritmo entra en el formulario y se van produciendo en el orden del documento.

- **Alcance de las variables**

VoiceXML utiliza una jerarquía de ámbitos de ECMAScript para permitir declarar las variables en diferentes niveles dentro de la aplicación.

Por ejemplo, una variable declarada en el ámbito del documento puede hacer referencia a cualquier lugar dentro de ese documento, mientras que una variable local declarada en un elemento sólo está disponible en ese elemento.

Las variables se pueden declarar en los ámbitos que se describen en la Tabla 2.14.

Ámbito	Descripción
sesión	Variables de sólo lectura que pertenecen a toda una sesión de usuario. Son declaradas por el intérprete de contexto.
aplicación	Variables que se declaran con elementos <var> y <script> como hijas del documento raíz <vxml>. Se inicializan cuando se carga el documento raíz de la aplicación. Existen mientras el documento raíz está cargado y son visibles para cualquier documento de la aplicación.
documento	Variables que se declaran con <var> y <script> en un documento concreto que no sea el documento raíz. Se inicializan cuando se carga el documento y son accesibles sólo dentro de ese documento.
diálogo	Variables declaradas como hijas de <form> o <menu>. Son accesibles sólo dentro de ese elemento de diálogo.
anónimo	Cada elemento <block>, <filled>, y <catch> definen un nuevo alcance anónimo en el que se pueden declarar variables sólo visibles en el interior del propio elemento.

Tabla 2.14. Ámbitos de las variables

La Figura 2.26 muestra el diagrama de la jerarquía de ámbitos.

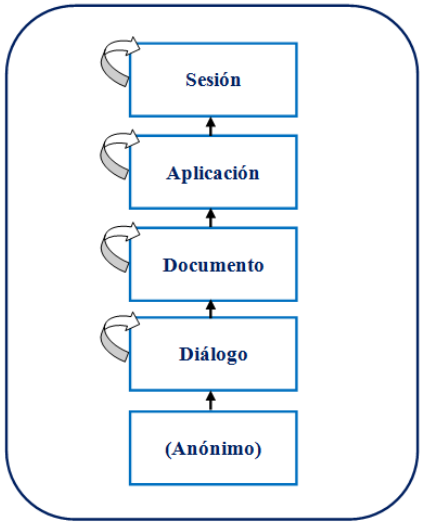


Figura 2.26. Jerarquía del alcance de las variables

No se recomienda utilizar "session", "application", "document" y "dialog" como nombres de variables e ítems de formulario. Aunque no son palabras reservadas, al utilizarlas se ocultan las variables predefinidas con el mismo nombre por reglas de ámbito de ECMAScript utilizadas por VoiceXML.

Existen variables ya declaradas para referenciar aspectos de la sesión y de la aplicación VoiceXML.

- **Referencia a variables**

Las variables son referenciadas con los atributos *cond* y *expr*. Lo podemos observar en el ejemplo de la Figura 2.27.

```
<if cond="ciudad == 'Leganés'">
   <assign name="ciudad" expr="'Madrid'"/>
<elseif cond="city == 'Alcorcón'"/>
   <assign name="ciudad" expr="'Madrid'"/>
<elseif cond="ciudad =='Illescas'"/>
   <assign name="city" expr="'Toledo'"/>
</if>
```

Figura 2.27. Ejemplo de referencia a variables

2.2.7.2 Tratamiento de eventos

La plataforma lanza eventos en los casos en los que el usuario no responde, no responde de forma que la aplicación le entienda, se solicite ayuda, etc. Además, el intérprete produce eventos si encuentra un error semántico en un documento de VoiceXML, o cuando se encuentra con un elemento <throw>. Los eventos se identifican mediante cadenas de caracteres.

Un elemento hereda los *catch elements* de cada uno de sus elementos ancestros. Por ejemplo, si un *field* no contiene el elemento de capturas para *nomatch*, pero su formulario si lo tiene definido, se utilizará el elemento de capturas de *nomatch* del formulario. De esta manera, el comportamiento relativo al tratamiento de eventos puede especificarse en cualquier nivel y se aplica a todos los niveles inferiores.

- **Elemento *throw***

El elemento <throw> produce un evento. Estos eventos pueden ser eventos predefinidos por la plataforma como los de los ejemplos de la Figura 2.28.

```
▪ <throw event="nomatch"/>
▪ <throw event="connection.disconnect.hangup"/>
```

Figura 2.28. Ejemplos de <throw>

O también pueden ser eventos definidos por el usuario, como el de la Figura 2.29.

```
▪ <throw event="com.att.portal.machine"/>
```

Figura 2.29. Ejemplo de evento definido por el usuario

Los atributos de <throw> son los que se describen en la Tabla 2.15.

Atributos	Descripción
event	Evento que se produce.
eventexpr	Expresión ECMAScript que evalúa el nombre del evento que se produce.
message	Cadena de mensaje que proporciona contexto adicional sobre el evento que se produce.
messageexpr	Expresión ECMAScript que evalúa la cadena del mensaje.

Tabla 2.15. Atributos de <throw>

Se debe especificar uno de entre los atributos "*event*" o "*eventexpr*"; de lo contrario, se produce un evento *error.badfetch*.

- **Elemento *catch***

Para capturar y tratar los eventos, independientemente de su origen, se dispone del elemento *catch*. Dentro de este elemento se especificarán las acciones a realizar en caso que el evento sea capturado según la estructura que vemos en la Figura 2.30.

```
<catch event="help">
      Acciones a realizar
</catch>
```

Figura 2.30. Ejemplo de elemento *catch*

El elemento *catch* tiene la variable *_event*, que contiene el nombre del evento que se produjo. Por ejemplo, el código de la Figura 2.31 puede manejar dos tipos de eventos.

```
<catch event="event.x event.y">
  <if cond="_event=='event.x'">
    <!—Para el evento event.x -->
    <audio src="x.wav"/>
  <else/>
    <!—Para el evento event.y -->
    <audio src="y.wav"/>
  </if>
  <!—Continuar con manejadores normales para el resto de
eventos-->
</catch>
```

Figura 2.31. Ejemplo de *catch* usando la variable *_event*

En el ejemplo de la Figura 2.31, la variable *_event* es inspeccionada para seleccionar el audio a reproducir en función del evento que se produjo. Se reproducirá el archivo

x.wav para eventos de event.x y se reproducirá el archivo y.wav para eventos de event.y. El resto del código contendrá contenido ejecutable común a ambos tipos de eventos.

Si un elemento <catch> contiene un elemento <throw> con el mismo evento, puede producirse un bucle infinito como se aprecia en la Figura 2.32.

```
<catch event="help">
   <throw event="help"/>
</catch>
```

Figura 2.32. Ejemplo de bucle infinito por el incorrecto uso de *catch*

La plataforma podría detectar esta situación y generar un error semántico. Los atributos de <catch> son los mostrados en la Tabla 2.16.

Atributos	Descripción
event	Evento o eventos a capturar. Puede especificarse una lista de eventos separados por espacios, que indicará que el elemento <catch> detecta todos los eventos de la lista. En este caso existe un contador para cada evento. Si no se especifica el atributo, todos los eventos deben ser capturados.
count	Contador de las apariciones del evento. El valor por defecto es 1. El recuento permite controlar las veces que aparece el mismo evento.
cond	Expresión que debe ser *true* después de su conversión a *boolean* para que el evento sea capturado. El valor por defecto es *true*.

Tabla 2.16. Atributos de <catch>

- **Notación abreviada**

Los elementos <error>, <help>, <noinput>, y <nomatch> tienen formas abreviadas para los elementos <catch> más habituales.

El elemento de <*error*> es la forma abreviada de <*catch event = "error"*> y atrapa a todos los eventos de tipo error según se puede observar en el ejemplo de la Figura 2.33.

```
<error>
    Ha ocurrido un error – por favor llame de nuevo en unos
    minutos.
    <exit/>
</error>
```

Figura 2.33. Ejemplo de codificación abreviada para el evento <error>

El elemento <*help*> es una abreviatura para <*catch event = "help"*> como se ejemplifica en la Figura 2.34.

```
<help> No hay ayuda disponible.</help>
```

Figura 2.34. Ejemplo de codificación abreviada para el evento <help>

El elemento <*noinput*> abrevia < *catch event = "noinput"* >. Un ejemplo se muestra en la Figura 2.35.

```
<noinput> Lo siento, no he podido oirle.</noinput>
```

Figura 2.35. Ejemplo de codificación abreviada para el evento <noinput>

El elemento de <*nomatch*> es la abreviatura de <*catch event = "nomatch"*> como se puede observan en la Figura 2.36.

```
<nomatch> Lo siento, no le he entendido.</nomatch>
```

Figura 2.36. Ejemplo de codificación abreviada para el evento <nomatch>

Los anteriores elementos tienen los atributos mostrados en la Tabla 2.17.

Atributo	Descripción
count	Número de sucesos (como en <catch>).
cond	Condición opcional para probar si el evento es capturado por este elemento. El valor por defecto es *true*.

Tabla 2.17. Atributos de las abreviaturas de <catch>

- **Elementos *catch* predeterminados**

Se espera que el intérprete proporcione manejadores de catch implícitos por defecto para los eventos *noinput*, *help*, *nomatch*, *cancel*, *exit* y *error* si el autor no los ha especificado.

- **Tipos de eventos**

Hay eventos predefinidos y eventos específicos de la aplicación y de la plataforma. Los eventos también se subdividen en **eventos planos** (casos que se producen normalmente) y **eventos de error** (casos anómalos). La convención de nomenclatura de errores permite múltiples niveles de granularidad.

Los eventos predefinidos son:

- **cancel:** El usuario solicita una cancelación.
- **connection.disconnect.hangup:** El usuario ha colgado.
- **connection.disconnect.transfer:** El usuario ha sido transferido incondicionalmente a otra línea y no volverá.
- **exit:** El usuario ha solicitado salir.
- **help:** El usuario ha pedido ayuda.
- **noinput:** El usuario no ha respondido dentro del intervalo de tiempo de espera.
- **nomatch:** El usuario escribió algo pero no fue reconocido.
- **maxspeechtimeout:** La entrada del usuario es demasiado larga excediendo el valor de *'maxspeechtimeout'*.

Además de errores de transferencia, los errores predefinidos son:

- **error.badfetch:** El contexto del intérprete lanza este evento cuando se produce un error en la captura de un documento y se llega a un lugar donde se requiere ese resultado.
- **error.badfetch.http.*response_code***
- **error.badfetch.protocol.response_code:** En el caso de un fallo de *fetch*, el contexto del intérprete debe utilizar un evento detallado diciendo qué código de respuesta se ha producido.
- **error.semantic:** Se encontró un error de tiempo de ejecución en el documento de VoiceXML.
- **error.noauthorization:** Se produce cuando la aplicación intenta realizar una operación que no está autorizada por la plataforma.
- **error.noresource:** Se ha producido un error en tiempo de ejecución debido a que un recurso de plataforma solicitado no estaba disponible durante la ejecución.
- **error.unsupported.builtin:** La plataforma no admite un tipo de gramática solicitado.
- **error.unsupported.format:** El recurso solicitado tiene un formato que no es compatible con la plataforma.
- **error.unsupported.language:** La plataforma no admite el lenguaje para síntesis o reconocimiento de voz.
- **error.unsupported.objectname:** La plataforma no admite un determinado objeto específico de la plataforma.
- **error.unsupported.(element):** La plataforma no admite un determinado elemento de VoiceXML.

2.2.7.3 Otros elementos ejecutables para el control de flujo

El contenido ejecutable se refiere a un bloque de lógica de procedimiento. Esta lógica aparece en:

- El ítem de formulario <block>.
- Las acciones <filled> en formularios e ítems de entrada.
- Los controladores de eventos (<catch>, <help>, etc.).

Los elementos ejecutables se ejecutan en el orden del documento en su bloque de lógica de procedimiento. Si un elemento ejecutable genera un error, el error se produce

inmediatamente. Los elementos ejecutables posteriores en ese bloque de lógica de procedimiento no se ejecutarán.

- **Elemento *assign***

El elemento <assign> asigna un valor a una variable como se demuestra en la Figura 2.37.

```
▪ <assign name="flavot" expr="'chocolate'"/>
▪ <assign name="ticket.coste" expr="documento.ticket+5"/>
```

Figura 2.37. Ejemplos del elemento <assign>

No se puede hacer una asignación a una variable que no ha sido explícitamente declarada con un elemento de <var> o con una instrucción de *var* dentro de un <script>.

Los atributos de *assign* son los que se muestran en la Tabla 2.18.

Atributos	Descripción
name	Nombre de la variable que es asignado.
expr	Nuevo valor de la variable.

Tabla 2.18. Atributos <assign>

- **Elemento *clear***

El elemento <clear> restablece el valor de una o más variables, incluyendo los elementos del formulario. Un ejemplo del uso de este elemento se ejemplifica en la Figura 2.38.

```
<clear namelist="ciudad codigopostal informacion"/>
```

Figura 2.38. Ejemplo del elemento <clear>

En la Tabla 2.19 se muestra el atributo del elemento *clear*.

Atributo	Descripción
namelist	Lista de variables cuyos valores se van a restablecer. Cuando este atributo no se especifica, se reinician todos los ítems de formulario en el formulario actual.

Tabla 2.19. Atributo <clear>

- **Elementos *if*, *elseif* y *else***

Las sentencias condicionales sirven para dirigir el control de la ejecución de una aplicación. El lenguaje VoiceXML declara tres elementos para la definición de estas sentencias condicionales: <if>, <else> y <elseif>.

La condición se expresa mediante el atributo *cond*, cuya evaluación debe ser un valor booleano. Este elemento es obligatorio en *if* y en *elseif*, pero no puede aparecer en *else*. En caso de ser cierta la condición, es decir que la evaluación de la misma sea true, se ejecutarán las sentencias del elemento correspondiente.

La estructura de estos elementos tiene una sintaxis especial debido a las reglas impuestas por la conformidad con XML:

- El elemento *if* tiene una etiqueta de apertura y otra de cierre, dentro de la cual definirá toda la sentencia condicional.

- Los elementos *elseif* y *else* deben ir dentro de un bloque *if* y son etiquetas cerradas (<elseif.../> y <else/> respectivamente), es decir, estas etiquetas no tienen hijos.

- Un elemento *else*, en caso de aparecer en un bloque condicional, aparecerá una sola vez y será el último elemento hijo de *if*.

En la Figura 2.39 se codifica un ejemplo donde se usan estos tres elementos.

```
<if cond= "total > 1000">
      <prompt> El producto es demasiado caro.</prompt>
<elseif cond=" total == 1000 "/>
      <prompt> El producto es caro.</prompt>
<else/>
      <prompt> El producto no es caro.</prompt>
</if>
```

Figura 2.39. Ejemplo de los elementos *if, elseif y else*

- **Elemento *reprompt***

El FIA espera que el elemento *catch* almacene los *prompts* de forma apropiada según se esté manejando un evento. Por lo tanto, el FIA generalmente no realiza la selección y el encolamiento de *prompts* en la siguiente iteración siguiendo la ejecución de un elemento *catch*.

Sin embargo, el FIA realiza la selección y la formación de colas de mensajes normal después de la ejecución de un elemento *catch* (<catch>, <error>, <help>, <noinput>, <nomatch>) en dos casos:

- Si el elemento *catch* termina mediante la ejecución de un <goto> o <submit> a otro diálogo, o si termina con un <return> de un subdialog. En estos casos, el

nuevo diálogo necesita que se garantice que su *prompt* inicial permanece intacto y no puede ser suprimida o sustituido.
- Si se ejecuta un <reprompt> en el *catch* para solicitar que se reproduzcan los *prompts* posteriores.

En estos dos casos, después de que el FIA selecciona el siguiente ítem de formulario a visitar, realiza el procesamiento normal de los *prompts*, incluyendo la selección, el encolamiento y el incremento del contador de estos *prompts*. El elemento <reprompt> no tiene ningún efecto fuera de un *catch.*

- **Elemento *goto***

El elemento <*goto*> se utiliza para:

- realizar una transición a otro ítem de formulario en el formulario actual,
- realizar una transición a otro diálogo en el documento actual, o
- realizar una transición a otro documento.

Para realizar la transición a otro ítem de formulario, se utiliza el atributo *nextitem,* o el atributo *expritem* en el caso en el que el nombre del elemento de formulario se calcule mediante una expresión ECMAScript. La utilización de estos dos atributos se ejemplifica en la Figura 2.40.

```
▪ <goto nextitem="confirmacion"/>
▪ <goto expritem="(type==12)? 'confirmacion':'anulacion'"/>
```

Figura 2.40. Ejemplo de *goto* en transiciones a otros ítems de formulario

Para ir a otro diálogo en el mismo documento, se utiliza *next* (o *expr*) con sólo un fragmento URI. Ejemplos de este tipo de transición se observan en la Figura 2.41.

```
▪ <goto next="#nombre_otro_dialogo"/>
▪ <goto expr="'#' + 'nombre_otro_dialogo'"/>
```

Figura 2.41. Ejemplo de *goto* en una transición a otro diálogo

Para realizar la transición a otro documento se utiliza *next* (o *expr*) con una URI como se puede apreciar en la Figura 2.42.

```
▪ <goto next="http://example.com/reserva"/>
▪ <goto next="./info_horarios#bibliotecas"/>
```

Figura 2.42. Ejemplo de *goto* en una transición a otro documento

- **Elemento *submit***

El elemento <submit> se utiliza para enviar información al servidor Web de origen y, a continuación, hacer una transición al documento enviado de vuelta en la respuesta. A diferencia de <goto>, permite presentar una lista de variables para el servidor de documentos mediante una solicitud *HTTP get* o *post* [HTTP].

El diálogo al que se quiere hacer el tránsito se especifica mediante la referencia URI en el atributo *next* o *expr* del elemento <submit>, como se puede observar en la Figura 2.43.

```
<submit next="URI_destino" method="post"
 namelist="variable1 variable2 variable3 … " />
```

Figura 2.43. Ejemplo elemento <submit>

Si el diálogo o documento para la transición no es válido (es decir, el diálogo o documento no existe) se lanzará un *error.badfetch*.

Los atributos más comunes de <submit> son los mostrados en la Tabla 2.20.

Atributos	Descripción
next	Referencia URI del documento al que se quieren enviar las variables.
expr	Lo mismo que *next*, excepto que la referencia URI será determinada de forma dinámica mediante la evaluación de la expresión dada de ECMAScript.
namelist	Lista de variables a enviar separadas por espacios en blanco. Este atributo es opcional. Si no se especifica, se enviarán todas la variables declaradas mediante el atributo *name* en el campo del formulario que contenga al <submit>.
method	Método de solicitud: *get* (por defecto) o *post*.
enctype	Tipo de medio de codificación del documento enviado (cuando el valor de *method* es "post").

Tabla 2.20. Atributos <submit>

Se debe especificar exactamente uno de los atributos "next" o "expr", de lo contrario, se producirá un evento *error.badfetch*.

En el ejemplo de la Figura 2.44, al no haberse definido la lista de variables a enviar al documento especificado en *next*, se enviará el parámetro *dni*.

```
<field name= "dni" type="digits">
      <prompt> Introduzca su DNI</prompt>
      <filled>
            <submit method="post" next="recogerDNI.vxml"/>
      </filled>
</field>
```

Figura 2.44. Ejemplo elemento <submit>

- **Elemento *exit***

Devuelve el control al contexto del intérprete, el cual determina qué hacer a continuación. Se codifica según se muestra en la Figura 2.45.

```
<exit/>
```

Figura 2.45. Ejemplo elemento <exit>

Este elemento se diferencia de <return> en que termina todos los documentos cargados, mientras que <return> vuelve de una invocación de <subdialog>.

Si el <subdialog> ha causado un nuevo documento (o aplicación) para ser invocado, entonces <return> hará que ese documento termine, pero la ejecución se reanudará después del <subdialog>. El elemento <exit> no produce un evento "exit".

- **Elemento *return***

El elemento *return* termina la ejecución de un subdiálogo y devuelve el control y los datos al diálogo que hace la llamada.

Los atributos del elemento return se muestran en la Tabla 2.21.

Atributos	Descripción
event	Vuelve y, a continuación, lanza este evento.
eventexpr	Vuelve y, a continuación, inicia el evento al que esta expresión de ECMAScript evalúa.
message	Cadena de mensaje que proporciona contexto adicional sobre el evento que se produce.
messageexpr	Expresión de ECMAScript que evalúa a la cadena de mensaje.
namelist	Nombres de las variables que se devuelven al diálogo que llama. El valor por defecto es no devolver ninguna variable.

Tabla 2.21. Atributos <return>

Debe especificarse exactamente uno de entre los atributos "event", "eventexpr" o "namelist", de lo contrario, se produce un evento *error.badfetch*.

- **Elemento *disconnect***

Hace que el contexto del intérprete se desconecte del usuario. Como resultado, el contexto del intérprete producirá un evento de *connection.disconnect.hangup* y entrará en el estado de procesamiento final. Al procesarse el elemento <*disconnect*> también se vaciará la cola de *prompts*.

- **Elemento *script***

El elemento <script> permite la especificación de un bloque de código de lenguaje de secuencias de comandos de cliente y es análogo al elemento HTML <SCRIPT> [HTML].

En la Figura 2.46 se codifica una secuencia de comandos que calcula un factorial.

```
<script> <![CDATA[
      function factorial(n) {
      return (n <= 1)? 1 : n * factorial(n-1);
      }
      ]]> </script>
<form id="form">
      <field name="fact">
      <grammar type="application/srgs+xml"
      src="/grammars/number.grxml"/>
      <prompt>
            Dígame un número y le diré su factorial.
      </prompt>
      <filled>
      <prompt>
            <value expr="fact"/> tiene como número factorial
            <value expr="factorial(fact)"/>
      </prompt>
      </filled>
      </field>
</form>
```

Figura 2.46. Ejemplo elemento <script>

Un elemento <script> puede aparecer en elementos <vxml> y <form>, o en contenido ejecutable (<filled>, <if>, <block>, <catch>, o las formas cortas de <catch>).

Los atributos más comunes del elemento <script> son los que se muestran en la Tabla 2.22.

Atributos	Descripción
src	URI que especifica la ubicación del script si es externo.
charset	Codificación de caracteres del script designado por *src*. El valor por defecto es UTF-8.

Tabla 2.22. Atributos <script>

Se debe especificar un atributo de entre "src" o un *script* en línea (pero no ambos), de lo contrario, se produce un evento *error.badfetch*.

- **Elemento *log***

El elemento <log> permite que una aplicación genere un registro o mensaje de *debug* que el desarrollador puede usar como ayuda en el desarrollo de la aplicación o análisis de post-ejecución del comportamiento de la aplicación. En la Figura 2.47 se muestra un ejemplo de este elemento.

```
<log> The card number was <value expr="card_num"/> </log>
```

Figura 2.47. Ejemplo del elemento <log>

2.3 Plataforma Voxeo

2.3.1 ¿Qué es Voxeo?

Voxeo Corporation (www.voxeo.com) es una compañía que ha desarrollado la plataforma *VoiceCenter*, la cual permite crear, implementar y probar aplicaciones de voz desarrolladas en VoiceXML de forma telefónica y completamente gratuita.

Además, Voxeo incluye una web estática local para el alojamiento de las aplicaciones de voz, un número de teléfono dedicado para su uso durante la prueba de las aplicaciones, registro y depuración en tiempo real de la aplicación, conectividad de voz sobre IP, foros de soporte donde se puede interactuar con la Corporación Voxeo y con otros miembros de la comunidad y soporte extremo 24x7.

2.3.2 ¿Cómo funciona Voxeo?

Para explicar el funcionamiento de la plataforma de voz Voxeo se va a comparar el modo de operación de un servidor web con el de la red Voxeo.

En la web, los exploradores realizan solicitudes a los servidores web por medio de protocolos de Internet como HTTP. Los servidores web alojan páginas estáticas y aplicaciones dinámicas HTML que se devuelven al explorador. El resultado es una página visual que se ve en un navegador web. Este mecanismo lo podemos observar en la Figura 2.48.

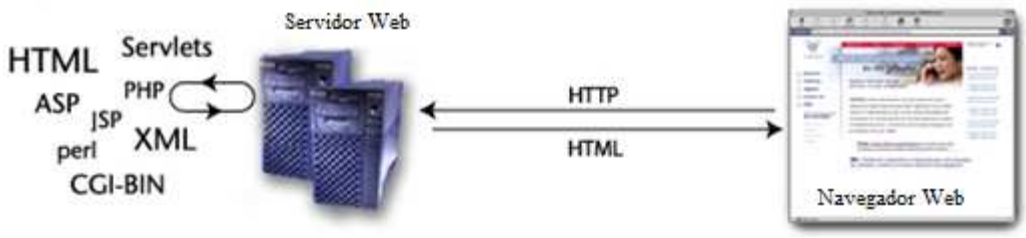


Figura 2.48. Interpretación de una página web

En la red de Voxeo, las aplicaciones de voz siguen el mismo modelo de navegador que las páginas web estándares. La diferencia es que, en vez de HTML, se utilizan lenguajes de marcado para voz (VoiceXML, CCXML [CCXML] o CallXML [CallXML]) para escribir la página. Cuando estas páginas son procesadas por la red de Voxeo, lo que se devuelve es un audio. La Figura 2.49 muestra este procedimiento.

Figura 2.49. Interpretación de una página VoiceXML

2.3.3 Desarrollo de una aplicación VoiceXML con la plataforma Voxeo

En este apartado se explica cómo crear una cuenta de Voxeo, qué herramientas ofrece esta herramienta y cómo configurar de forma detallada una aplicación VoiceXML en ella.

2.3.3.1 Creación de una cuenta de Voxeo

La cuenta de Voxeo incluye todo lo necesario para desarrollar aplicaciones de voz. Para crear una cuenta hay que registrarse en *www.evolution.voxeo.com* según muestra la Figura 2.50.

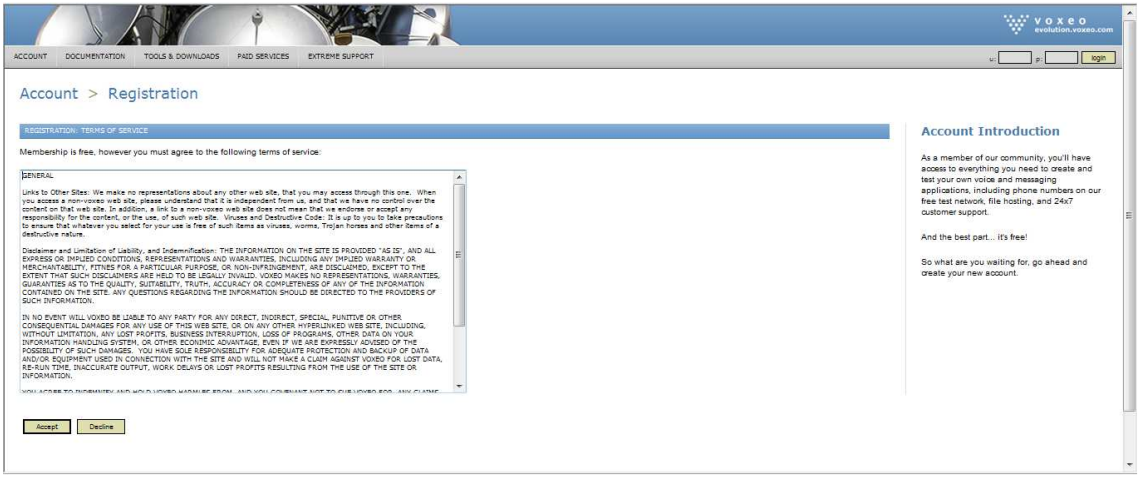


Figura 2.50. Registro en Voxeo

Una vez se haya efectuado el registro, se accede a la plataforma introduciendo los valores *username* y *password*, tal y como se observa en la captura de pantalla de la Figura 2.51.

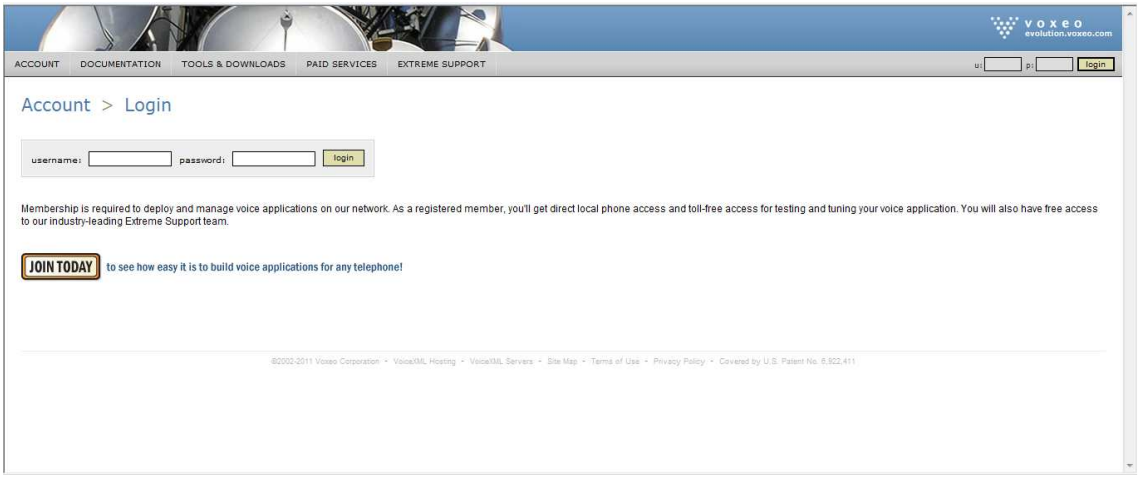


Figura 2.51. Acceso a la cuenta de Voxeo

La vista general al acceder a la cuenta personal de Voxeo es la que se puede observar en la Figura 2.52.

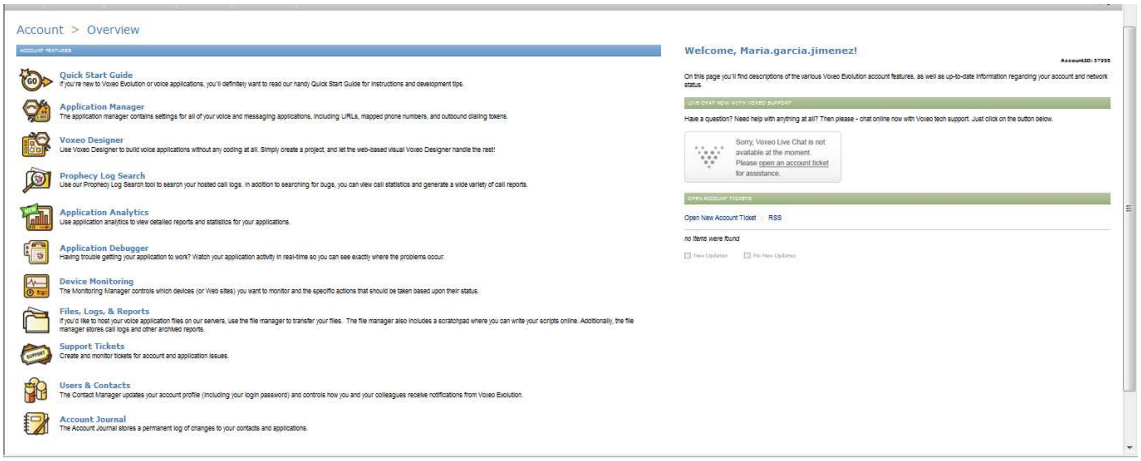


Figura 2.52. Vista general cuenta de Voxeo

Las opciones de herramientas que ofrece Voxeo en la cuenta creada son las siguientes:

- **Quick Start Guide**

Guía rápida de iniciación de la plataforma Voxeo para aquellos usuarios nuevos en Voxeo Evolution o en las aplicaciones de voz.

En esta guía se detallan las herramientas de desarrollo disponibles y las instrucciones a seguir para desarrollar aplicaciones.

- **Application Manager**

La herramienta Application Manager contiene la configuración de las aplicaciones de voz y mensajería, incluyendo URLs, números de teléfono asignados a las aplicaciones, y *tokens* de marcado.

- **Voxeo Designer**

Se utiliza Voxeo Designer para crear aplicaciones de voz sin necesidad de codificación. Simplemente, se crea un proyecto de forma gráfica y Voxeo Designer lo gestiona.

- **Prophecy Log Search**

Herramienta para gestionar los registros de la llamada realizada. Además de facilitar la búsqueda de errores, se pueden ver las estadísticas de la llamada y generar una amplia variedad de informes de esta llamada.

- **Application Analytics**

Aplicación para ver informes detallados y estadísticas de las aplicaciones implementadas.

- **Application Debugger**

Si se producen problemas para conseguir que la aplicación funcione, con Application Debugger se puede ver la actividad de las aplicaciones en tiempo real para comprobar exactamente dónde ocurren estos errores.

- **Device Monitoring**

La herramienta Device Monitoring controla qué dispositivos (o sitios web) se desean supervisar y las acciones específicas que se deben realizar basadas en su estado.

- **Files, Logs, & Reports**

Si se desean alojar los archivos de aplicación de voz en los servidores de Voxeo, se utiliza el administrador de archivos para transferir estos archivos. Este administrador

también incluye un *scratchpad* donde se pueden escribir scripts y además, almacena los registros de llamadas y otros informes de archivado.

- **Support Tickets**

Con la herramienta Support Tickets se crean y controlan los tickets para cuestiones de cuenta y de aplicación.

- **Users & Contacts**

En esta aplicación se gestiona el perfil de la cuenta (incluyendo la contraseña de inicio de sesión) y se controla cómo los usuarios reciben notificaciones desde Voxeo Evolution.

- **Account Journal**

Herramienta que almacena un registro permanente de los cambios de los contactos y aplicaciones de un usuario.

2.3.3.2 Selección de la plataforma de la aplicación de voz

Voxeo ofrece tres plataformas de aplicaciones de voz: CallXML, CCXML y VoiceXML. Las características de cada una de ellas se explican a continuación.

- **Prophecy - CallXML 3.0**

La plataforma CallXML está disponible exclusivamente en Voxeo. CallXML 3.0 agrega la capacidad de utilizar menús de reconocimiento de voz simples dentro de sus aplicaciones, y emplea las operaciones de lógica condicional integrada para extender aún más sus capacidades.

CallXML se adapta a las necesidades de la mayoría de las aplicaciones de telefonía que utilizan entrada DTMF. Si en la aplicación a desarrollar no es necesario el reconocimiento de voz complejo, CallXML puede ser la opción correcta como plataforma.

CallXML 3.0 está disponible para su uso alojada en la red Voxeo, o como instalación de prueba libre para un equipo local.

- **Prophecy - CCXML W3C 1.0**

Voxeo ofrece la primera plataforma compatible con CCXML 1.0. Habiendo distribuido más de 1.000 millones de llamadas desde su introducción comercial en el año 2002, es una buena elección para implementar aplicaciones de conferencia de nueva generación y de enrutamiento de llamadas y garantizar a su vez que permanezcan en la vanguardia de la tecnología.

CCXML 1.0 está disponible para su uso en el servidor de Voxeo o como una instalación de prueba gratis en un equipo local.

- **Prophecy - VoiceXML 2.1**

  VoiceXML 2.1 está disponible como una solución alojada en el propio servidor de Voxeo o en un equipo local. Incluye el reconocedor de voz Voxeo y es el primero y único en el mundo compatible al 100% con la certificación VoiceXML 2.0.

  VoiceXML 2.1 permite desarrollar aplicaciones que no es necesario modificar si se migran a otra plataforma compatible. Además, soporta todas las ampliaciones y mejoras de VoiceXML 2.1, así como los estándares de formato de gramáticas SISR/SRGS. También incluye soporte para los formatos de gramática antiguos GSL y JSGF, ofreciendo una enorme flexibilidad y variedad de opciones para las implementaciones IVR (Interactive Voice Response) activadas por voz.

2.3.3.3 Edición y manejo de ficheros

En primer lugar, se debe elegir un editor de texto en el que se desarrolle el código de la aplicación. Para este Proyecto Final de Carrera se ha utilizado el editor de texto Java VoiceXML, *JVoxEdit* (www.jvoxedit.sourceforge.net/). Se trata de un editor específico para *VoiceXML* con funciones de ayuda, como es un entorno de desarrollo integrado, debido a la complejidad y extensión del código a implementar. Además, es un editor de texto gratuito *open source* distribuido bajo los términos de la Licencia Pública General (GNU).

JVoxEdit está programado en *Java, y* su interfaz de usuario utiliza *Java Swing*, biblioteca gráfica para Java que incluye *widgets* tales como cajas de texto, botones, desplegables y tablas. Este editor es independiente del sistema operativo.

El público al que *JVoxEdit* está destinado principalmente son programadores de la industria de las telecomunicaciones y la tecnología de la información. Su fecha de lanzamiento fue el 10-11-2007 y está registrado el 08-06-2008.

Este editor analiza regularmente los documentos *VoiceXML* y, en tablas específicas, va creando una vista de árbol del documento e informando sobre los errores de sintaxis que se van produciendo.

Las figuras que se muestran en este apartado son capturas de pantalla del editor para documentar de una manera gráfica la interfaz *JVoxEdit*.

En la Figura 2.53 está implementado un ejemplo, *holamundo.vxml*, de forma correcta. En la Figura 2.54 se puede apreciar cómo se produce un error en el código y el editor avisa sobre qué error ha ocurrido y en qué línea se ha detectado el problema.

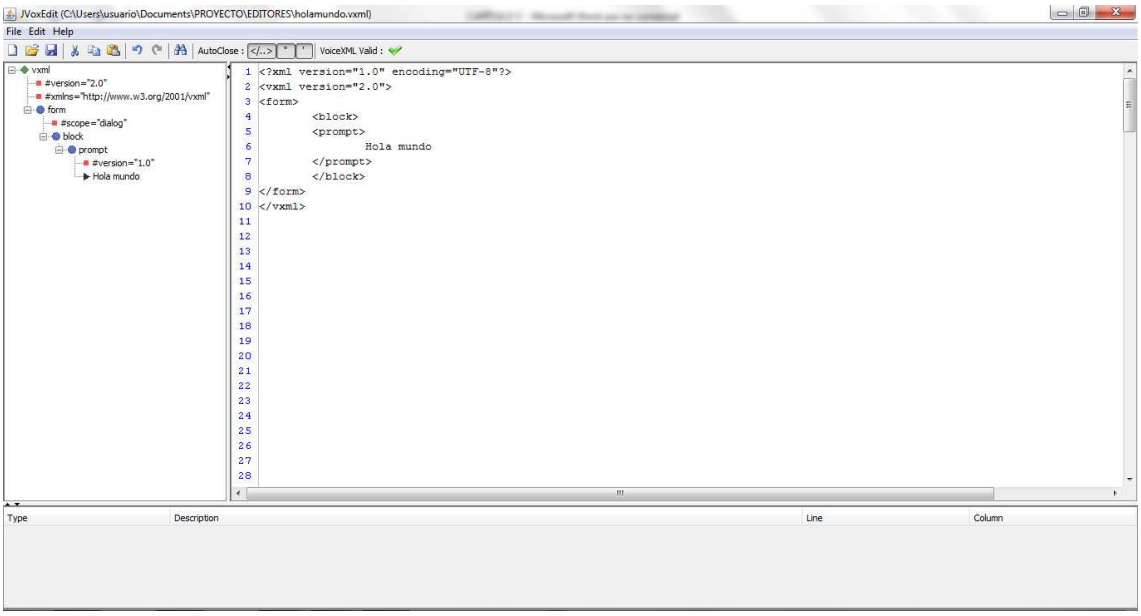


Figura 2.53. Pantalla principal de JVoxEdit con código correcto

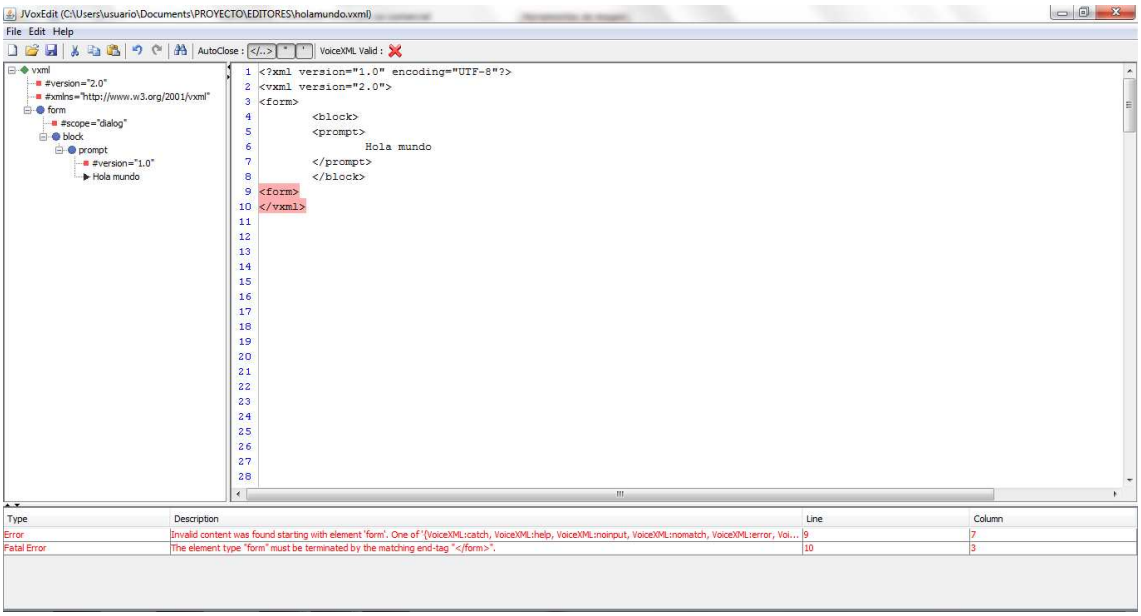


Figura 2.54. Pantalla principal de JVoxEdit con código erróneo

La Figura 2.55 y la Figura 2.56 muestran la pantalla principal del editor con un ejemplo codificado que incluye una gramática.

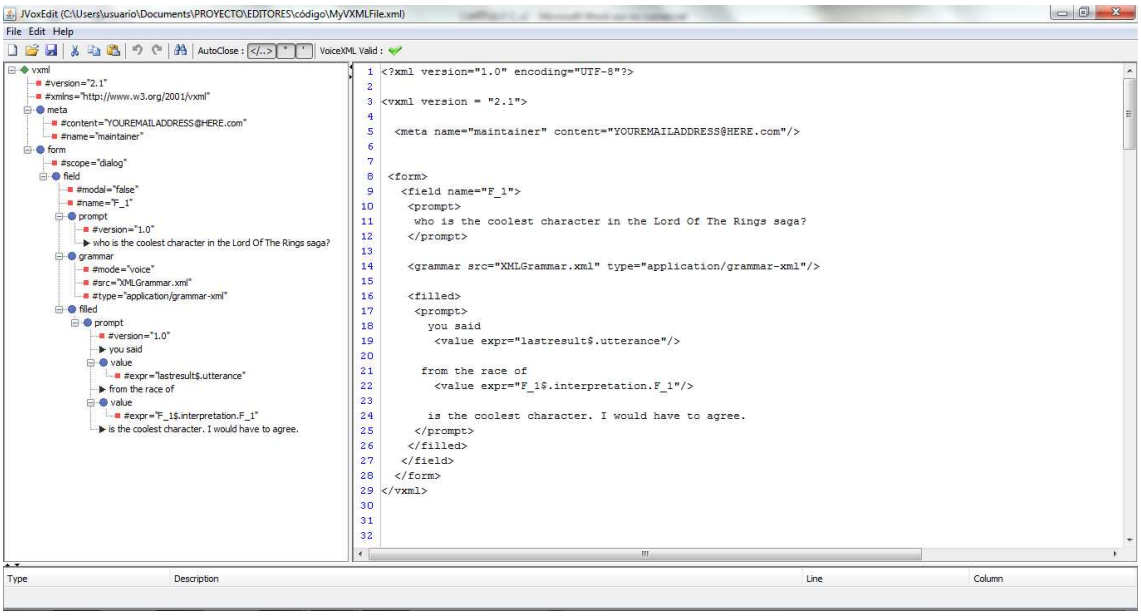


Figura 2.55. Ejemplo de código VoiceXML con gramática asociada

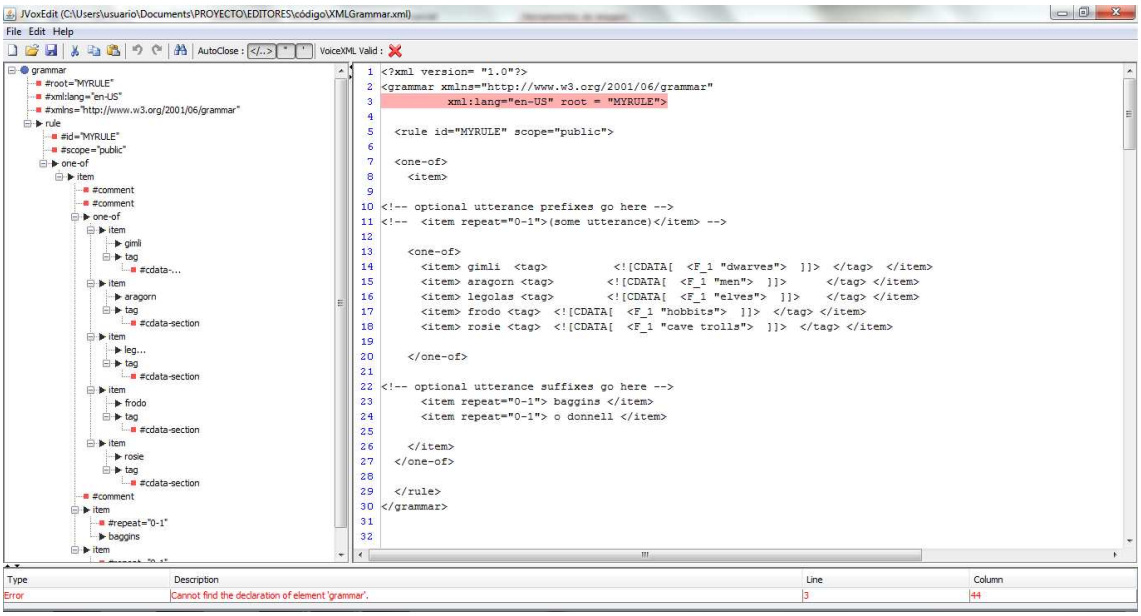


Figura 2.56. Ejemplo de gramática

Como el ejemplo está dividido en dos ficheros, el principal y la gramática asociada, al abrir por separado el fichero con la gramática, el editor nos avisa que ésta hace referencia al fichero principal.

Una vez se tienen los ficheros con el código fuente de la aplicación, se deben alojar en un servidor web accesible público. Si no se tiene acceso a un servidor web, Voxeo ofrece un espacio de hospedaje gratuito para estos archivos. Para ello, simplemente hay que acceder al menú "*Files, Logs, Reports*" en el menú "*Account*" y cargar los archivos en el directorio */root/www*.

La Figura 2.57 y la Figura 2.58 muestran gráficamente los directorios donde se deben alojar los ficheros según se ha explicado anteriormente, y en la Figura 2.59 se observa cómo se ha cargado con éxito el fichero *holamundo.vxml*.

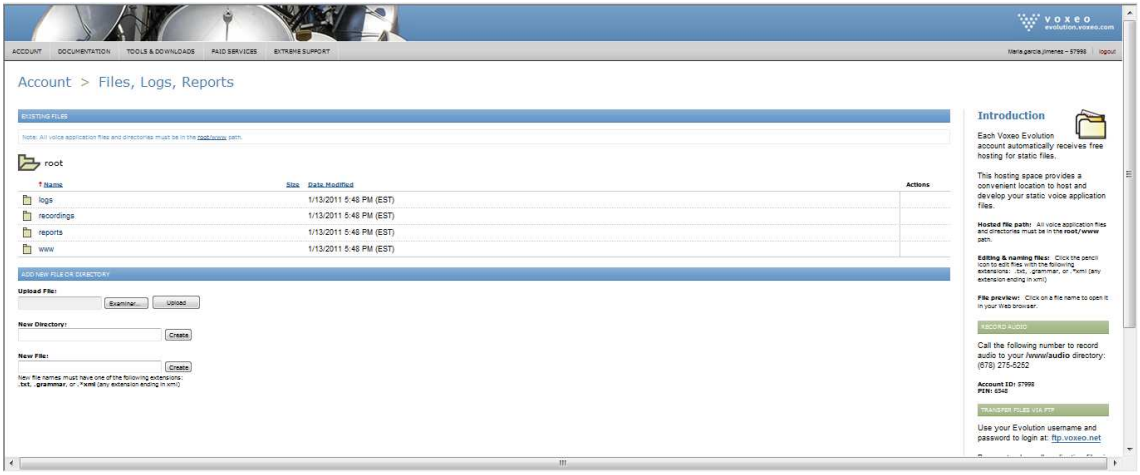


Figura 2.57. Directorio /root

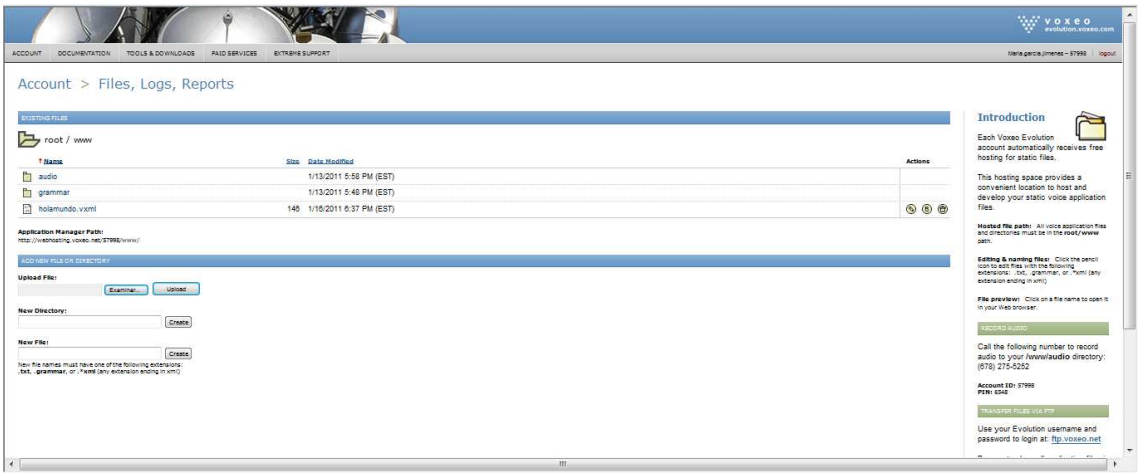


Figura 2.58. Directorio /root/www

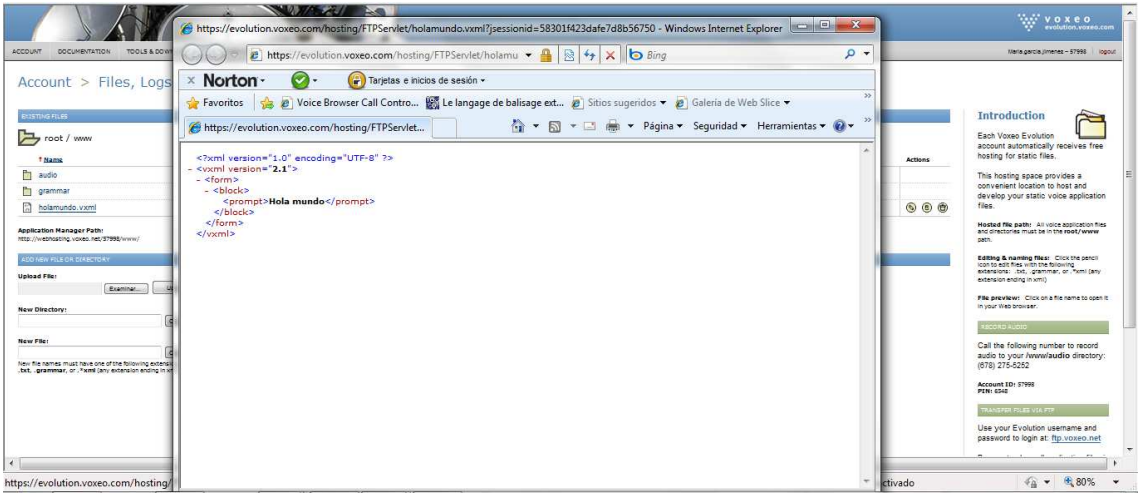


Figura 2.59. Fichero *holamundo.vxml* cargado en el directorio */root/www*

Si la aplicación posee más de un fichero, por ejemplo, si un fichero tiene una gramática asociada en otro fichero distinto, se deben almacenar todos estos ficheros en el directorio */root/www*.

La Figura 2.60 muestra cómo se han cargado dentro del directorio /root/www dos ficheros, *MyVXMLFile.xml* y su gramática asociada *VXMLGrammar.xml*.

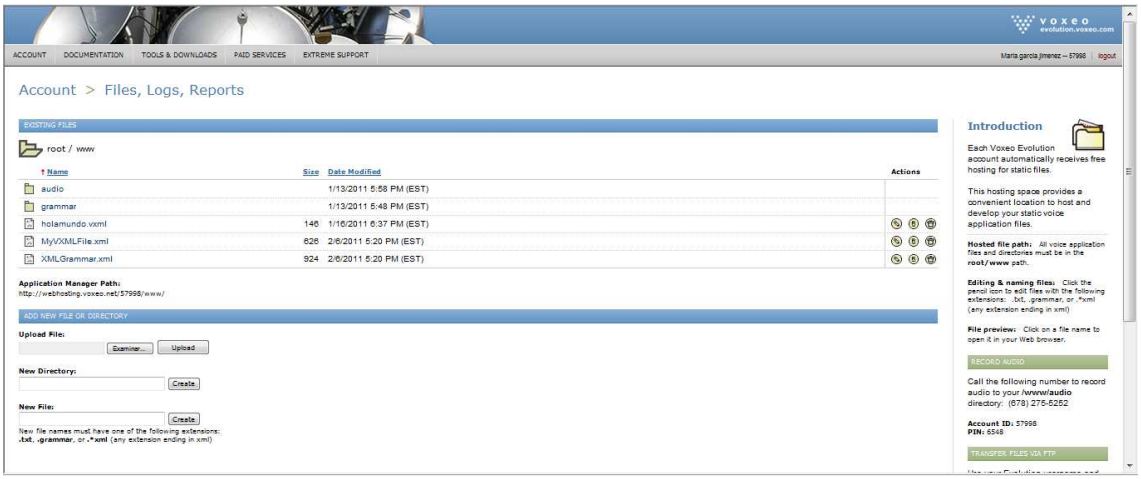


Figura 2.60: Ficheros MyVXMLFile.xml y VXMLGrammar.xml almacenados en el directorio /root/www

La Figura 2.61 muestra el código VoiceXML del fichero MyVXMLFile.xml y la Figura 2.62 muestra el código VoiceXML del fichero VXMLGrammar.xml.

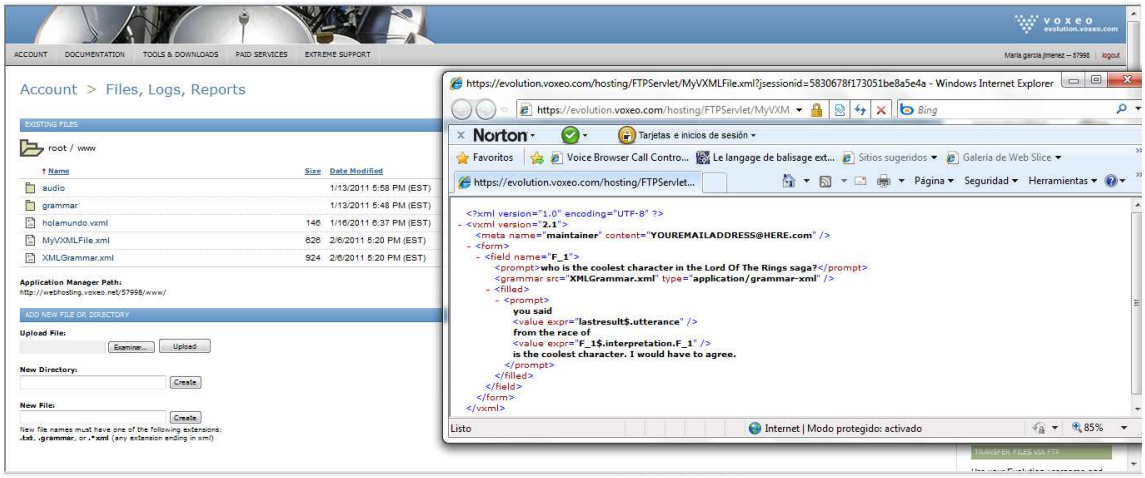


Figura 2.61. Código fichero MyVXMLFile.xml

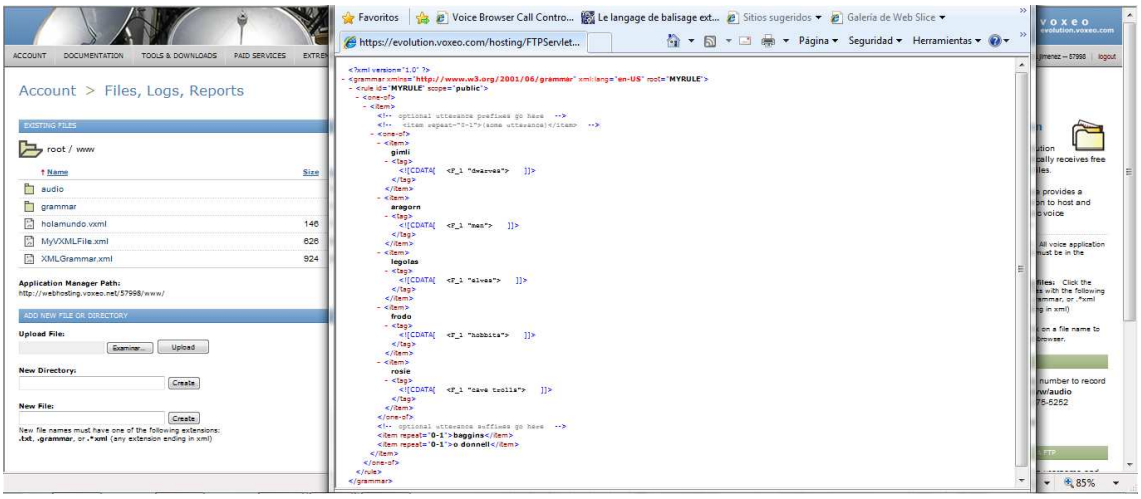


Figura 2.62. Código fichero VXMLGrammar.xml

2.3.3.4 Creación de una aplicación y vinculación a un número de teléfono

Para vincular la aplicación con un número de teléfono, se debe seleccionar "*Application Manager*" en el menú "*Account*". Dentro de este gestor de aplicaciones, se hace *click* en el botón "*Add application*" como muestra la Figura 2.63.

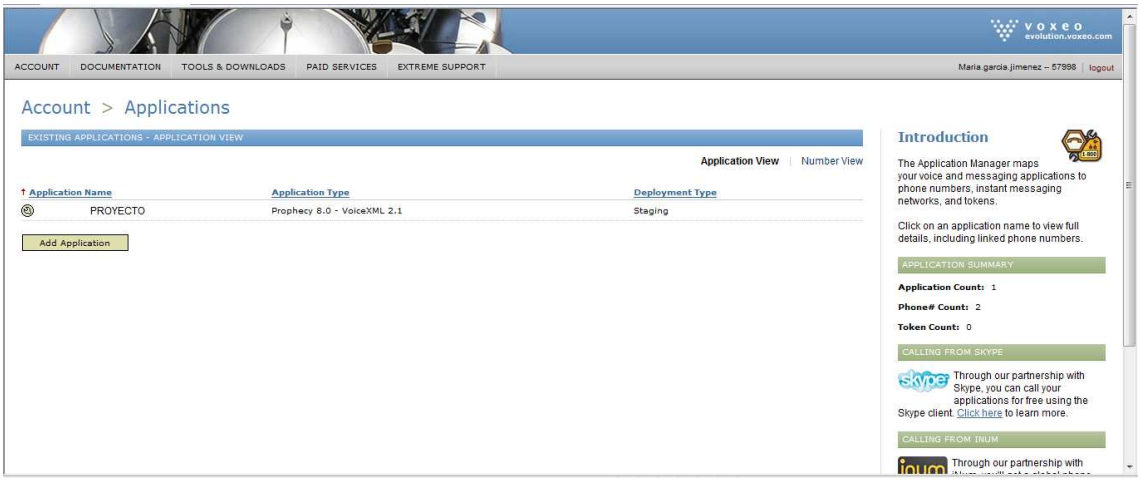


Figura 2.63. "*Application Manager*"

En la siguiente pantalla ("*Create a new application"*), se asigna un nombre a la aplicación, se selecciona el tipo de plataforma de las explicadas en el apartado 2.3.3.2 y se escribe la URL de acceso público para el código de la aplicación de voz en el campo "*Voice URL*". Si se utiliza el espacio de hospedaje gratuito de *Voxeo*, sólo hay que hacer *click* en el vínculo del "*File Manager*" para asignar el archivo que se ha cargado según el apartado anterior.

Finalmente, hay que completar todos los campos necesarios y hacer *click* en el botón "*Create application"*.

En la Figura 2.64 y en la Figura 2.65 se presentan gráficamente los pasos a seguir para la creación de la aplicación ‘PRUEBA’.

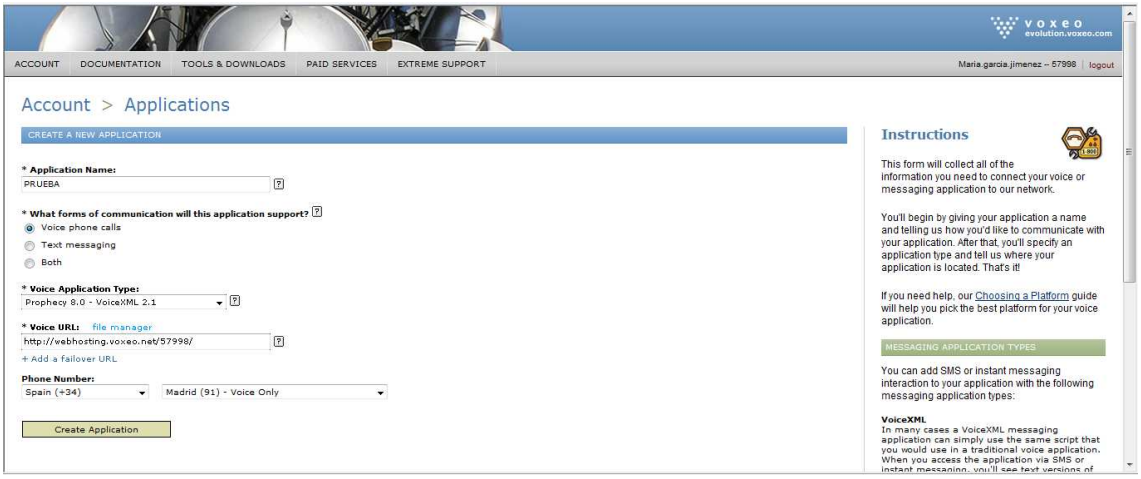


Figura 2.64. “*Create a new application*”

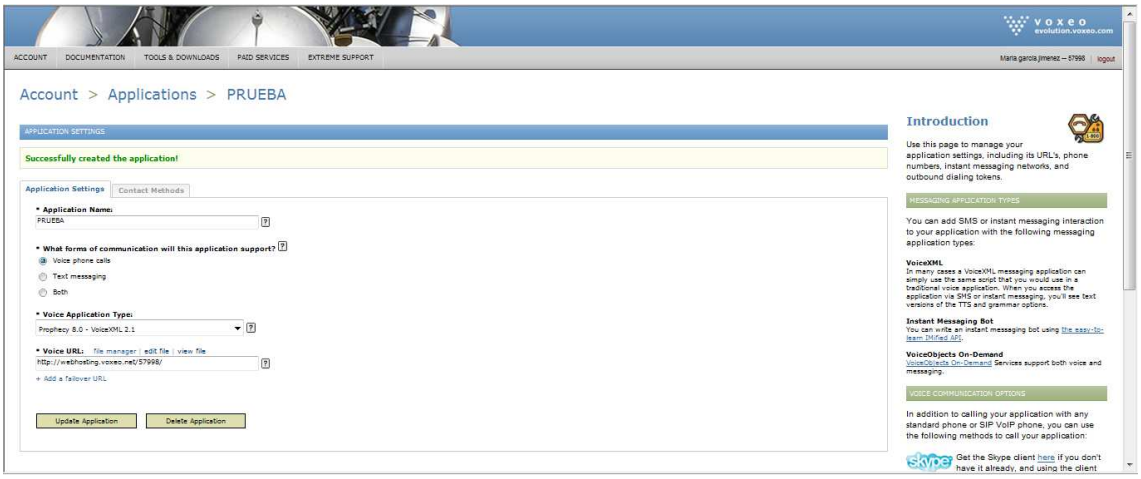


Figura 2.65. Especificar URL del código fuente

En la Figura 2.66 se observa cómo se crea la aplicación ‘Prueba gramática’. En este caso la aplicación tiene dos ficheros. El principal y la gramática asociada. Se debe escribir como dirección URL la correspondiente al fichero principal.

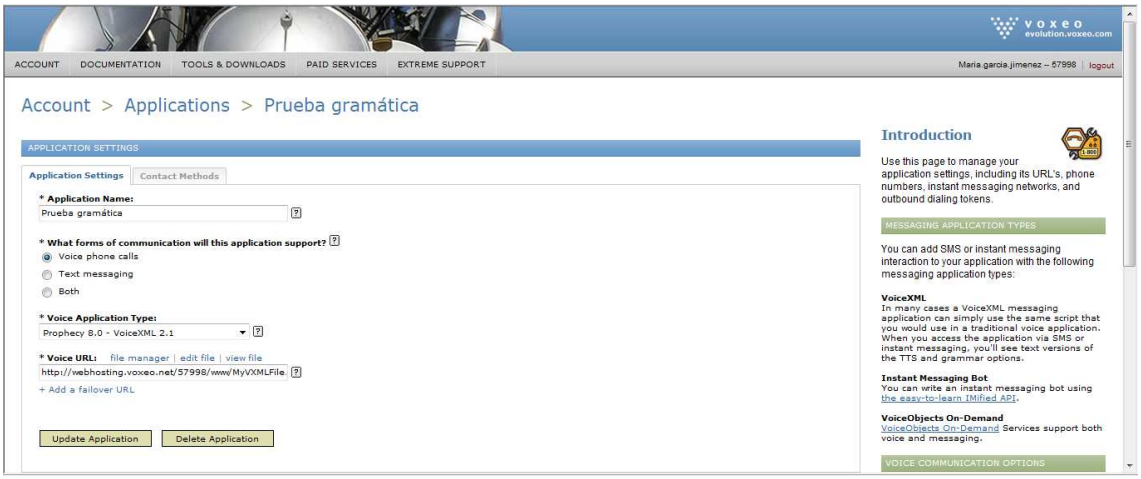


Figura 2.66. Creación aplicación ‘Prueba gramática’

Después de seguir los pasos anteriormente explicados, aparece una nueva pestaña, “*Contact Methods*”. En ella se especifican los números de teléfono asignados para la aplicación que se acaba de crear, tal y como podemos comprobar en la Figura 2.67 para la aplicación ‘PRUEBA’ y en la Figura 2.68 para la aplicación ‘Prueba gramática’.

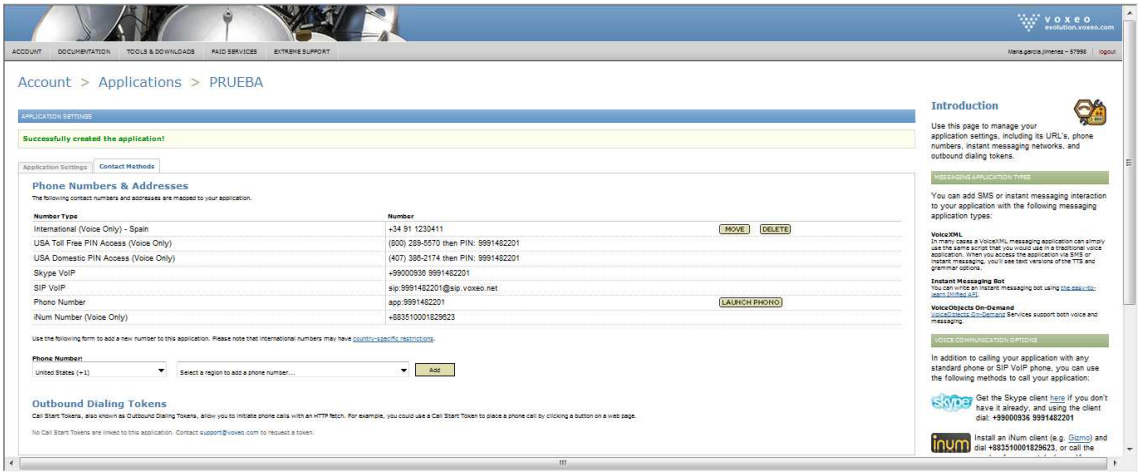


Figura 2.67. “*Contact Methods*” para la aplicación ‘PRUEBA’

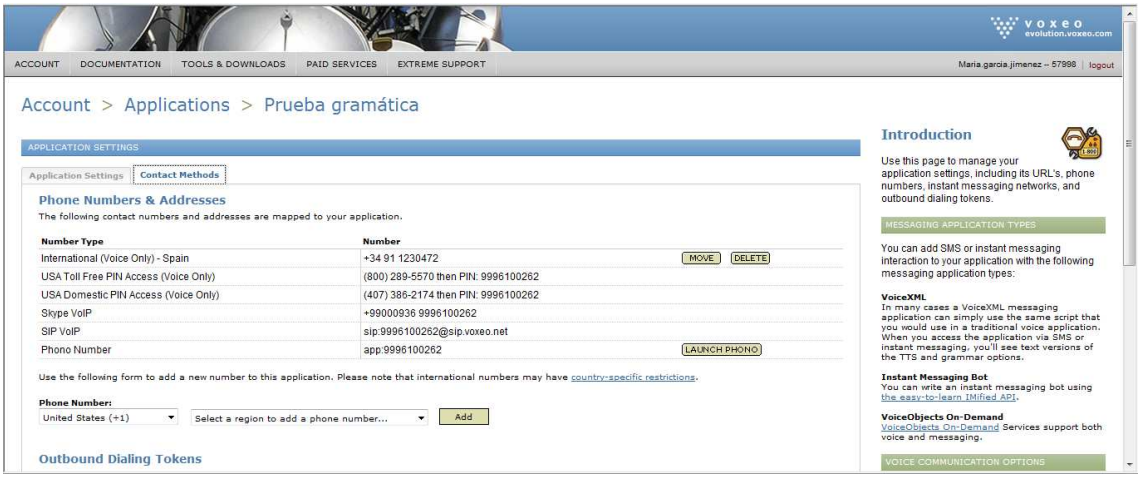


Figura 2.68. “*Contact Methods*” para la aplicación ‘Prueba gramática’

Después de haber creado las aplicaciones se puede acceder a ellas, modificarlas o borrarlas desde la opción *'Existing Applications'* en la pestaña *'Application manager'*, como se muestra en la Figura 2.69.

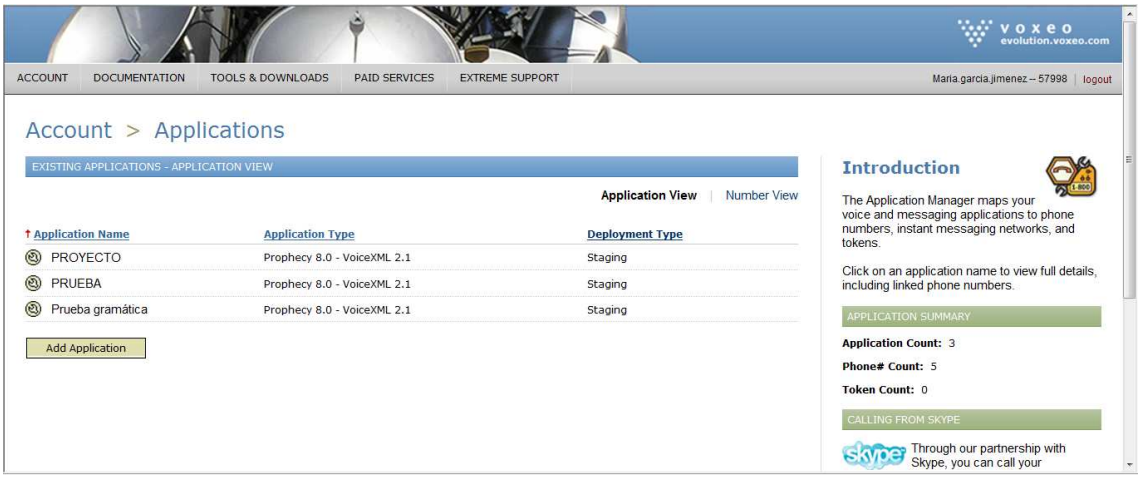


Figura 2.69. Aplicaciones existentes en la plataforma

2.3.3.5 Depurador y llamada a la aplicación

El depurador, "*Application Debugger*" en el menú "*Account*", muestra la actividad de la aplicación en tiempo real, lo que permite monitorizar su progreso y depurar los errores que se puedan producir.

Para observar este estado de la aplicación, una vez que se tiene el depurador abierto, se marca el número de teléfono asignado y se llama a la aplicación. Si se produce algún problema se puede utilizar el soporte incorporado, enviando una copia de los registros del depurador al equipo de soporte de Voxeo.

La aplicación de voz se puede probar llamando, por ejemplo, mediante *Skype* según se muestra en la Figura 2.70.

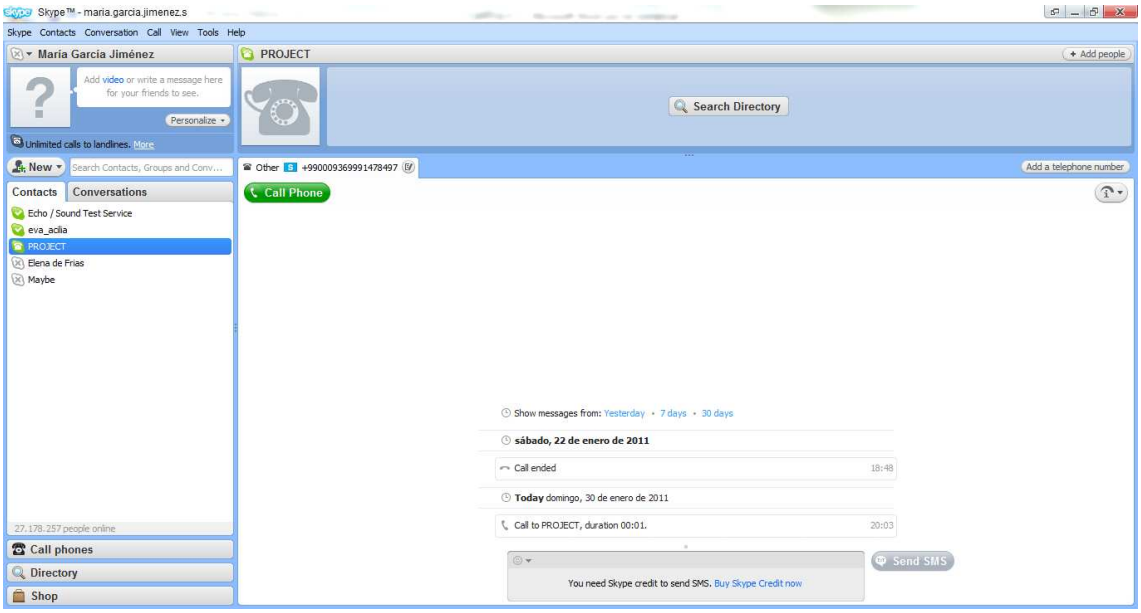


Figura 2.70. Llamada a la aplicación mediante *Skype*

La pantalla que se obtiene en el depurador de Voxeo de la llamada realizada en la pantalla anterior es la que observa en la Figura 2.71.

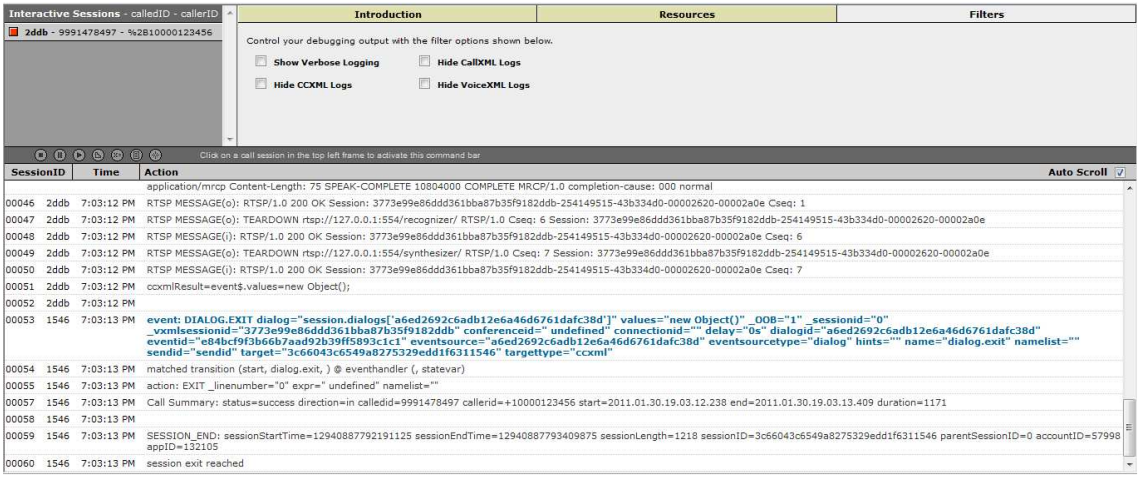


Figura 2.71. Depurador de Voxeo

2.3.4 Otras implementaciones del estándar VoiceXML

Además de la plataforma ofrecida por Voxeo, existen diversas implementaciones de las especificaciones de interfaces vocales del W3C para VoiceXML 2.0 y 2.1. Las más relevantes se describen brevemente a continuación.

- **Bevocal Café** [BEVOCALCAFE] es un entorno de desarrollo que proporciona un intérprete de VoiceXML 2.0 y las herramientas necesarias para depurar y probar aplicaciones. El intérprete de VoiceXML soporta las últimas recomendaciones del W3C, incluyendo mejoras como la verificación del usuario o la utilización de datos en formato XML.

- **Conita** [CONITA] ha implementado un intérprete VoiceXML basado en Open VXI.

- **Eloquant** [ELOQUANT] ha implementado un intérprete de VoiceXML 2.0. Este intérprete soporta las especificaciones de interpretación semántica definidas por el W3C para el reconocimiento de la voz y ofrece hospedaje para aplicaciones vocales.

- **HeyAnita** [HEYANITA] soporta la mayoría de los estándares de voz actuales, incluyendo VoiceXML 2.0. Ofrece el servidor FreeSpeech para el desarrollo, implementación y hospedaje de aplicaciones de voz. Es compatible con los principales sistemas operativos y permite a las empresas elegir el hardware de telefonía y el software de reconocedores y sintetizadores a utilizar.

- **HP** ofrece la plataforma **OpenCall** [OPENCALL], que soporta VoiceXML 2.0 en entornos HP-Unix y Linux para protocolos de telefonía ISUP/SIP/RDSI, software de reconocimiento y síntesis MRCP (por ejemplo, Nuance y SpeechWorks) y CCXML 1.0. También está disponible un escritorio VoiceXML SDK libre para el desarrollo de aplicaciones y el entorno OpenCall Speech Web para una gestión eficiente e implementación de portales de voz a gran escala.

- **IBM** [IBM] tiene una variedad de productos y herramientas para el desarrollo, depuración e implementación de aplicaciones VoiceXML. Los productos de aplicaciones VoiceXML de IBM incorporan la mayoría de las especificaciones VBWG, incluyendo VoiceXML 2.0 y SRGS 1.0.

- **Omvia Media Server** [OMVIAMEDIA] desarrollado por **Intervoice** incluye un navegador de VoiceXML 2.0. Intervoice también ofrece InVision Studio, una herramienta para el diseño, desarrollo y depuración de aplicaciones de VoiceXML. El Omvia Media Server soporta múltiples reconocedores y sintetizadores de voz.

- **JVoice XML** [JVOICEXML] es un intérprete de VoiceXML *de código libre* desarrollado en JAVA que soporta APIs como JSAPI y JTAPI. Su objetivo principal es tener una plataforma independiente de la implementación que se pueda utilizar de forma gratuita.

- **Loquendo** [LOQUENDO] ha desarrollado un intérprete de VoiceXML que administra documentos tanto VoiceXML 2.0 como 2.1. El intérprete está actualmente integrado en la plataforma **Loquendo VoxNauta** y se utiliza en una amplia gama de servicios de voz, incluso a gran escala. Por otra parte, **LoquendoCafé** proporciona a los desarrolladores recursos y herramientas para aprender a crear aplicaciones de voz y les permite ejecutar su aplicación de VoiceXML en una plataforma y escuchar el servicio que han creado por teléfono en diversos idiomas.

- **Lucent Technologies** [LUCENT] ofrece la pasarela **MiLife VoiceXML**, que proporciona acceso telefónico a servicios de voz en entornos web.

- **Motorola** licencia el sistema **VoxGategay** [VOXGATEGAY], que ha sido integrado en gran número de plataformas de voz.

- **Nuance** [NUANCE] ofrece una plataforma VoiceXML, herramientas gráficas para el desarrollo de aplicaciones, una plataforma para el desarrollo y prueba de prototipos y un navegador de voz basado en VoiceXML sin coste para los desarrolladores.

- **OpenVXI** [OPENVXI] es un intérprete de VoiceXML *de código libre* y portable desarrollado por la Universidad Carnegie Mellon y SpeechWorks. Puede ser usado de forma gratuita en aplicaciones comerciales y permite añadir módulos propietarios. Cumple la mayoría de las especificaciones definidas para VoiceXML 2.0.

- **OptimSys** ofrece **OptimTalk** [OPTIMTALK] una plataforma VoiceXML modular, flexible y escalable. Incluye intérpretes compatibles con VoiceXML y CCXML, y un SDK para agregar soporte a las soluciones VoiceXML y CCXML existentes. OptimTalk permite la elección de las tecnologías de reconocimiento y síntesis de voz y del hardware de telefonía.

- **PublicVoiceXML** [PUBLICVOICEXML] es una implementación *open source* de un explorador de VoiceXML 2.0 completo. Está diseñado para trabajar con hardware de telefonía de bajo coste mediante navegación DTMF y módulos de reconocimiento de texto a voz. Dispone de soporte y ejemplos de aplicaciones para telefonía móvil.

- **SpeechWorks** ofrece **OpenSpeech Browser** [OPENSPEECH], un kit modular de herramientas para los desarrolladores de aplicaciones VoiceXML 2.0. Su diseño flexible es la base de muchas soluciones comerciales de VoiceXML. OpenSpeech está disponible para sistemas operativos Windows y Linux, e incluye entre otras funcionalidades un intérprete de VoiceXML.

- **Tellme** [TELLME] es una plataforma que responde a más de un millón de llamadas de teléfono VoiceXML 2.0 cada día. Los desarrolladores VoiceXML pueden acceder a Tellme Studio para construir gratis sus propias aplicaciones.

- **Vocalocity** [VOCALOCITY] ha desarrollado una plataforma que cumple las últimas especificaciones de VoiceXML 2.0. Está diseñada específicamente para OEM (original equipment manufacturer) y Channel Partners que proporcionan soluciones de código libre a sus clientes. La plataforma Vocalocity soporta múltiples teléfonos, sintetizadores, reconocedores de voz y diversos sistemas operativos.

- **VoiceGenie** [VOICEGENIE] proporciona una plataforma que soporta los protocolos PSTN y SIP simultáneamente, 4 tipos de reconocedores de voz (Speech Works, BBN, AT&T Watson y Nuance) y 7 sintetizadores (incluyendo entre ellos SpeechWorks, AT&T Natural Voices, Rhetorical y SVOX). Además, dispone de un portal web donde las aplicaciones pueden desarrollarse y probarse de forma gratuita. VoiceGenie también ofrece la plataforma VoiceGenieCCXML

que implementa las especificaciones CCXML del W3C y es capaz de interactuar con la plataforma VoiceXML.

- **Voxpilot** [VOXPILOT] ofrece un entorno para el desarrollo e implementación online de aplicaciones multilingües VoiceXML. Incluye números de acceso locales en la mayoría de países europeos, proporcionando una robusta infraestructura VoIP.

Capítulo 3

Descripción general del sistema desarrollado

En este capítulo se describen las características fundamentales del conjunto de módulos que conforman el sistema desarrollado: funcionalidad, arquitectura y esquema general de bloques. A continuación se analizan las distintas tecnologías utilizadas para su desarrollo. Finalmente, se incluyen las operaciones más generales y codificaciones especiales que se han tenido en cuenta a lo largo de la implementación.

3.1 Presentación del sistema

El portal de voz municipal desarrollado para este Proyecto Final de Carrera ofrece al ciudadano el acceso telefónico a los siguientes servicios:

- ☎ Consultar información municipal.
- ☎ Realizar gestiones y trámites.
- ☎ Realizar encuestas.

☎ Acceder al buzón del ciudadano.

☎ Ser transferido a un tele-operador del Ayuntamiento.

Cada uno de estos servicios se explica en detalle en su módulo correspondiente dentro del Capítulo 4. La Figura 3.1 representa gráficamente las funcionalidades generales ofrecidas.

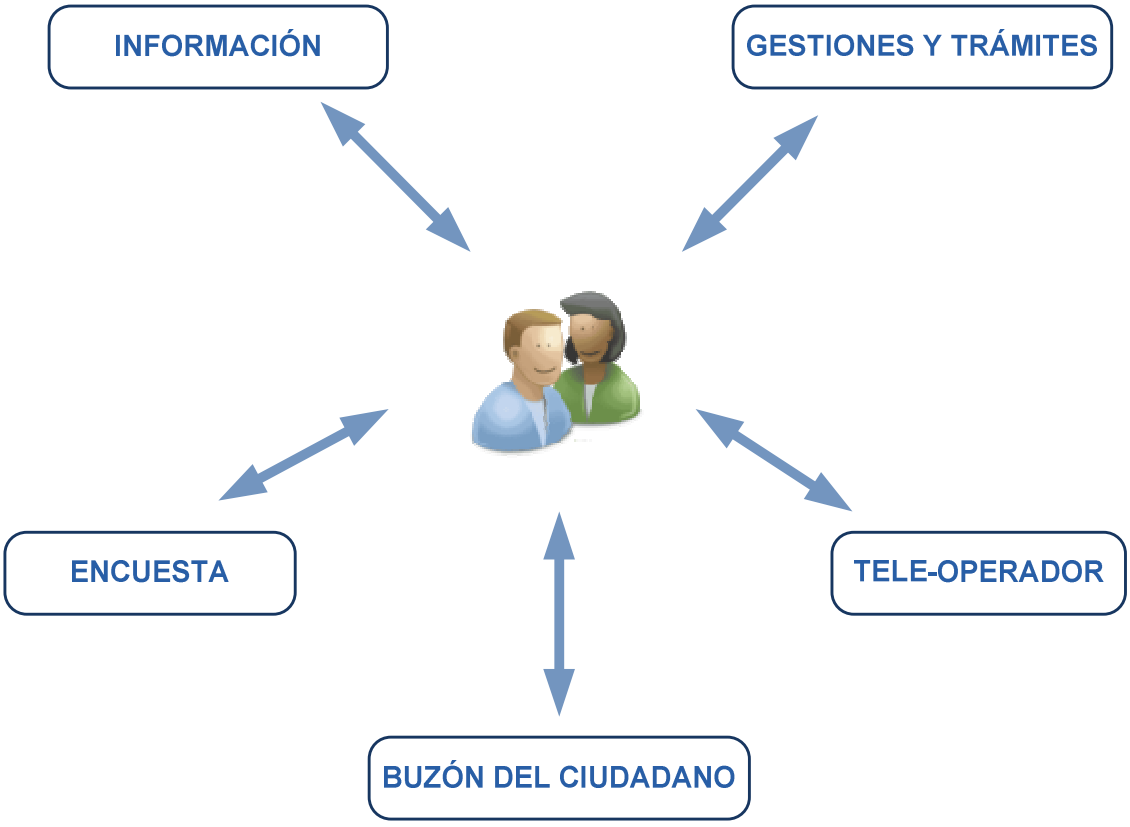


Figura 3.1. Servicios ofrecidos por el portal de voz municipal desarrollado para el Proyecto Final de Carrera

El desarrollo de la aplicación se ha hecho utilizando una arquitectura cliente-servidor cuyo esquema se puede observar en la Figura 3.2.

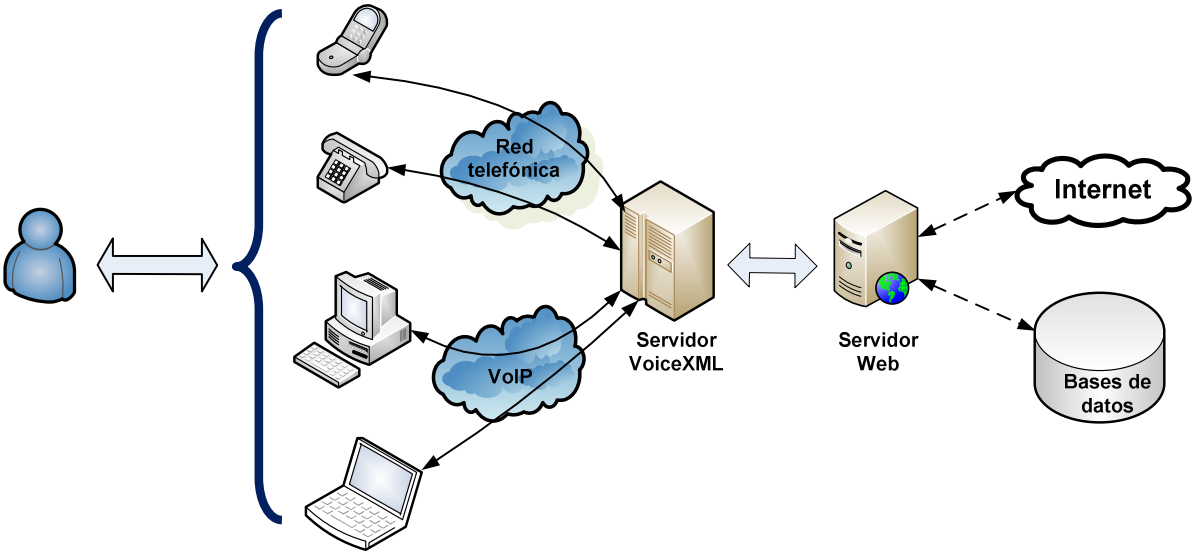


Figura 3.2. Arquitectura de la aplicación

La interacción con el sistema comienza cuando el usuario inicia una llamada, bien mediante la línea telefónica o bien mediante cualquier cliente de VoIP (por ejemplo, Skype).

En el servidor VoiceXML (para nuestro proyecto, la plataforma Voxeo Evolution), el intérprete de VoiceXML se encarga por un lado de responder las llamadas de los usuarios y, por otro, de interpretar los documentos VoiceXML para ofrecer el servicio al usuario. También es el encargado de solicitar los recursos necesarios para la ejecución de la aplicación, de seguir la lógica del servicio y de mantener el estado de sesión de los usuarios actuando en consecuencia. Para llevar a cabo estas acciones el intérprete VoiceXML debe disponer de una serie de elementos:

- Un sistema que se encargue de atender las llamadas de los usuarios. Para ello debe tener conexión al canal de acceso, ya sea por red telefónica o por voz sobre IP.

- Un sistema que gestione la comunicación con los servidores y vaya solicitando en cada momento los recursos necesarios identificados por sus URIs mediante, por ejemplo, los protocolos HTTP y FTP.

- Un sistema que reproduzca los ficheros de audio.

- Un sistema que genere audio a partir de texto: conversión de texto a voz, TTS (Text-to-Speech).

- Un sistema que se encargue de la recogida de datos del usuario, ya sea mediante mecanismos de reconocimiento de voz, como de reconocimiento de tonos DTMF.

- Un sistema que posibilite la grabación de voz.

- Un sistema que se encargue de la ejecución de la lógica del servicio, del mantenimiento de la sesiones y de la gestión de eventos.

En el servidor web se almacenan los ficheros *vxml* y *php* que implementan cada uno de los servicios prestados por el portal de voz, además de las bases de datos que contienen la información utilizada. Todos los ficheros se interrelacionan de tal forma que el usuario en su llamada puede navegar por cualquiera de los servicios ofrecidos, pudiendo realizar más de una acción en cada llamada.

El esquema general de la estructura de la aplicación se presenta en la Figura 3.3, donde se observa cómo el usuario interactúa con la aplicación, formada ésta por la plataforma Voxeo y el servidor web que aloja los distintos módulos y bases de datos.

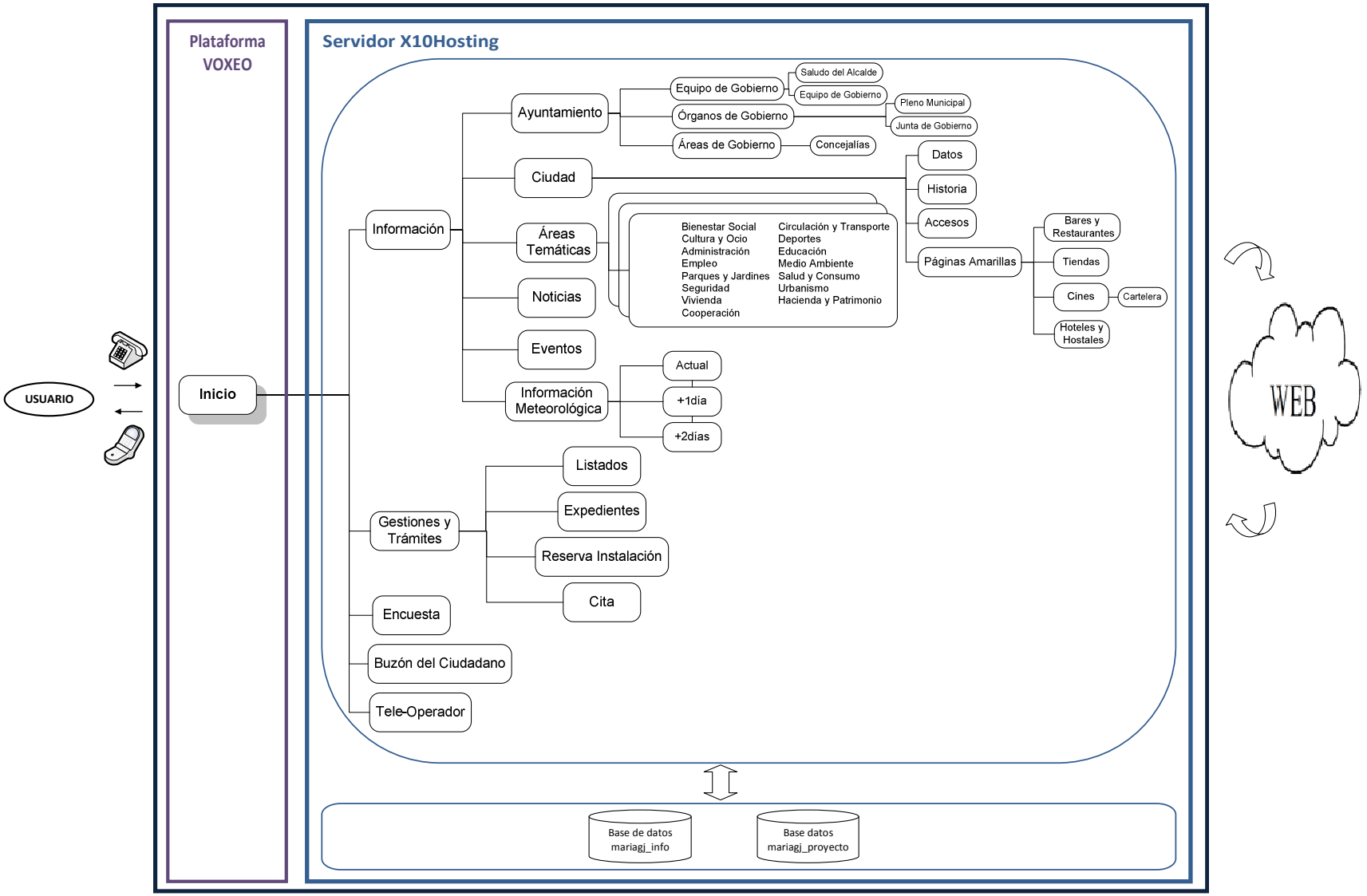


Figura 3.3. Visión completa de la aplicación desarrollada para el Proyecto Final de Carrera

3.2 Tecnologías

3.2.1 Intérprete VoiceXML: Voxeo

Actualmente existen multitud de intérpretes del lenguaje VoiceXML, tal y como se ha explicado en el apartado 2.3. Uno de ellos es ***Voxeo***, que proporciona la infraestructura y componentes necesarios para poder crear y usar aplicaciones de voz. Dado que en el apartado 2.3 se analizó en detalle esta herramienta, en las siguientes líneas se razonarán los motivos por los que ha sido elegida para la implementación y alojamiento de la aplicación desarrollada en este Proyecto Final de Carrera.

En primer lugar, Voxeo permite crear una aplicación VoiceXML y acceder a ella a través de distintos medios, ya que proporciona un número de teléfono local (según país y provincia en el caso de España) y un número Skype para llamadas desde la aplicación. Voxeo permite además hacer un seguimiento en tiempo real de las llamadas realizadas al servicio, así como visualizar y almacenar los ficheros de logs de las mismas, muy útil a la hora de depurar y optimizar la aplicación, estudiar el uso que se hace de ella e incluso obtener estadísticas. De esta forma, facilita el acceso a la aplicación tanto para poder utilizarla una vez finalizada como para realizar las pruebas necesarias a la hora de la implementación y depuración de sus funcionalidades. Cabe añadir que la plataforma posee un sistema de soporte rápido y eficaz, que incluye foros, tickets de soporte y una documentación muy completa. Además, todo ello de forma gratuita.

Por último, Voxeo proporciona el intérprete VoiceXML y los componentes ASR y TTS necesarios para la ejecución de la aplicación. En el presente Proyecto se ha utilizado la implementación *Prophecy 9 - Multi-Language VXML* proporcionada por Voxeo, que ha permitido desarrollar la aplicación para su uso en idioma castellano.

3.2.2 Base de datos: MySQL

MySQL es un sistema de gestión de bases de datos relacionales que se sustenta en un modelo de datos basado en la lógica de predicados y en la teoría de conjuntos. Se trata también de un sistema multihilo, ya que permite el acceso concurrente a la información de forma ordenada y segura. Finalmente es multiusuario, porque permite proveer servicio a múltiples usuarios simultáneamente.

MySQL AB (desde enero de 2008 una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril de 2009) desarrolla MySQL como software libre en un esquema de licencia dual.

Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos comerciales

deben comprar a la empresa una licencia específica que les permita este uso. Además de la venta de estas licencias privativas, la compañía ofrece soporte y servicios.

Se trata además del sistema de base de datos con uso más generalizado en aplicaciones web, en diferentes plataformas y sistemas operativos (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada al lenguaje PHP.

En el presente proyecto se optó por MySQL debido a su facilidad de manejo y a que aporta las siguientes ventajas:

- Escalabilidad: es posible manipular bases de datos enormes, del orden de seis mil tablas y alrededor de cincuenta millones de registros, y hasta 32 índices por tabla.

- MySQL está escrito en C y C++ y probado con multitud de compiladores y dispone de APIs para muchas plataformas diferentes.

- Conectividad: es decir, permite conexiones entre diferentes máquinas con distintos sistemas operativos. Es corriente que servidores Linux o Unix, usando MySQL, sirvan datos para ordenadores con Windows, Linux, Solaris, etc. Para ello se usa TCP/IP, tuberías, o sockets Unix.

- Tal y como se ha comentado, es multihilo, con lo que puede beneficiarse de sistemas multiprocesador.

- Permite manejar multitud de tipos para columnas.

- Permite manejar registros de longitud fija o variable.

- Uso gratuito tanto del sistema como de diferentes administradores de bases de datos MySQL, como el que se presenta en la siguiente sección.

3.2.3 Administrador de base de datos: phpMyAdmin

PhpMyAdmin es un completo y potente administrador de bases de datos MySQL que se utiliza vía web. Con phpMyAdmin se pueden realizar casi todas las tareas de administración de una base de datos MySQL: crear o eliminar bases de datos; crear, eliminar o alterar tablas; eliminar, editar o agregar campos; ejecutar consultas SQL, etc. Las principales características de phpMyAdmin son:

- Multiplataforma.

- Multilenguaje (más de 50).

- Licencia GPL.

- Escrito en PHP.

Por todas las anteriores propiedades se ha decidido hacer uso de *phpMyAdmin versión 3.3.9.2* para administrar y gestionar las bases de datos utilizadas en la aplicación del portal de voz. Además, posee una interfaz gráfica muy intuitiva que facilita su uso, la cual se puede observar en la captura de pantalla de la Figura 3.4.

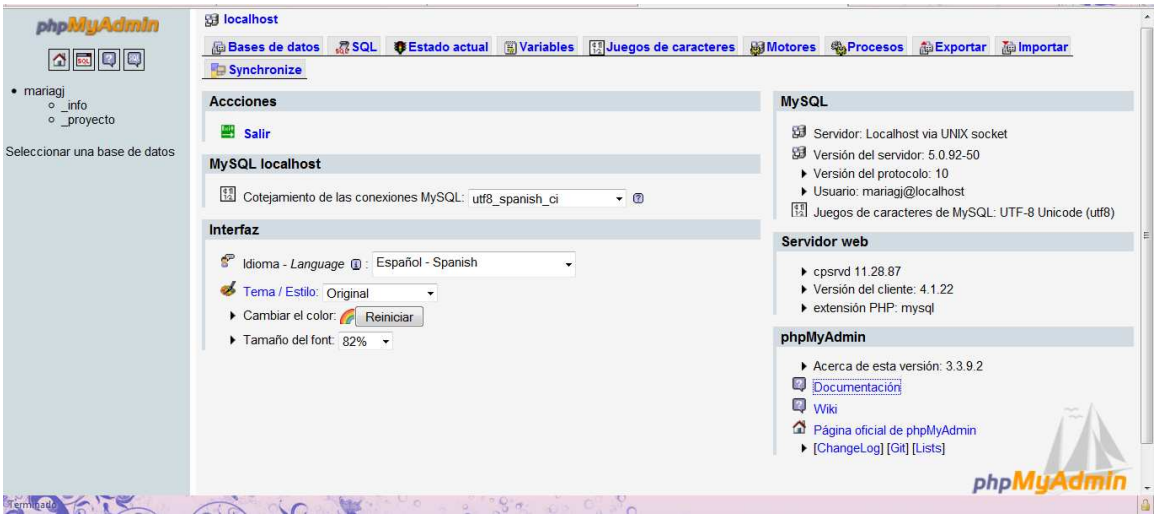


Figura 3.4. Pantalla de inicio del administrador phpMyAdmin

3.2.4 Servidor web: x10hosting

Como servidor web para el almacenamiento de los ficheros PHP y de las bases de datos de la aplicación se ha utilizado ***x10hosting***. Este servidor tiene las siguientes características para su plan gratuito (sin publicidad):

- cPanel 11+.
- 300 MB de almacenamiento.
- 10 Gb de banda ancha.
- Cuentas FTP.
- Posibilidad de agregar dominios o subdominios.
- Servidor Apache 2.2.
- PHP 5 y 3 bases de datos MySQL.
- CGI, PHP mail, POP3, IMAP, Sendmail y cuentas de e-mail.
- Auto-instalador de scripts.
- 99.9 % online y soporte técnico 24/7.

Otras ventajas de *x10hosting* es que permite subir archivos comprimidos, descomprimir en el servidor y subir archivos al directorio no web (directorio de acceso no público).

Una captura de la pantalla de inicio de x10hosting es la mostrada en la Figura 3.5.

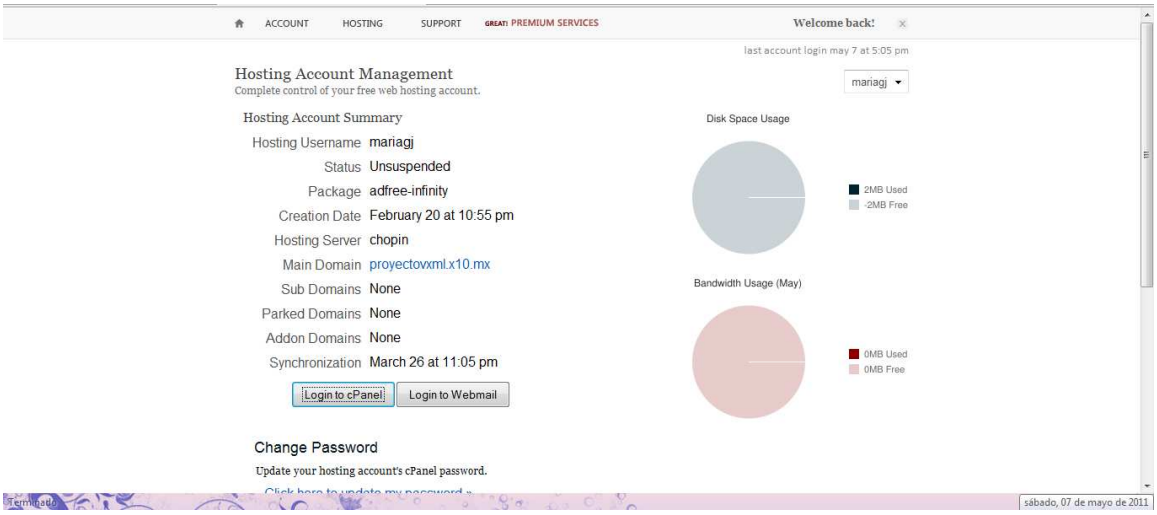


Figura 3.5. Pantalla de inicio del servidor x10hosting

3.3 Implementación de las operaciones generales

En este apartado se explican en detalle las operaciones comunes al conjunto de módulos que ha sido necesario tratar e implementar a lo largo del desarrollo del portal de voz.

3.3.1 Tratamiento de la información

En la aplicación desarrollada se trabaja con dos tipos de información: información estática e información dinámica. Cada una de ellas se trata de diferente forma a lo largo de la aplicación, tal y como se explica a continuación.

3.3.1.1 Información estática

Se ha considerado información estática a aquella que no cambia con el paso del tiempo, o al menos no lo hace en un tiempo considerable. Este tipo de información se recopila de páginas web, principalmente de la página web del Ayuntamiento de Alcorcón, y se almacena, perfectamente clasificada, en la base de datos de la aplicación. Cuando el usuario solicita este tipo de información, el sistema accede a ella en la base de datos y la devuelve encapsulada en un fichero VoiceXML.

Ejemplos de este tipo de información son la historia de Alcorcón, los accesos a la ciudad, los datos de contacto de un hotel de la ciudad o de la oficina de empleo del municipio.

3.3.1.2 Información dinámica

La información dinámica es aquella que cambia a lo largo del tiempo. Esta información se tiene que recargar en la base de datos cada cierto tiempo, ya que si no se actualizara podría llegar un momento en que la base de datos contuviera datos obsoletos. Ejemplos de este tipo de información son las noticias y los eventos municipales, la información meteorológica, la encuesta municipal y la cartelera del cine de Alcorcón.

Para no tener que cargar estos datos manualmente, se ha implementado un método que realiza este procedimiento automáticamente, *cargar_datos.php*. En este método se accede a las páginas web correspondientes, se hace un *procesado sintáctico* de la información y se almacenan los datos necesarios y actualizados en la base de datos. Cada vez que el usuario requiera este tipo de información, el sistema sólo tiene que acceder a la base de datos y devolverla. Este método lo ejecuta el administrador de la base de datos cada cierto tiempo predeterminado.

Hay un caso concreto de este tipo de información, la situación meteorológica actual, en la que la información va variando cada hora, y por tanto, es mejor devolverla directamente desde la página web correspondiente que ofrece este servicio sin necesidad de almacenarla en la base de datos.

3.3.2 Gramáticas

En todos los diálogos de la aplicación se utilizan gramáticas de voz y de DTMF, lo que implica que se puede navegar por los diferentes menús tanto mediante la voz como usando las teclas del teléfono, facilitando que la aplicación sea más accesible.

Las gramáticas codificadas son de tipo XML ya que es el formato estándar definido por el W3C y, por tanto, es soportado en cualquier plataforma VoiceXML. Además, este formato permite una mayor flexibilidad en cuanto a la estructura de la gramática y es más fácil de depurar.

A lo largo de la aplicación se diferencia entre gramáticas estáticas y gramáticas dinámicas, cuyas diferencias se explican a continuación.

3.3.2.1 Gramáticas estáticas

Las gramáticas estáticas tratan información que no varía a lo largo del tiempo, además de tener un número pequeño de opciones a elegir. Se codifican de forma fija en el mismo fichero donde se utilizan.

En la aplicación desarrollada se emplean dos tipos de estructuras para estas gramáticas, que se van alternando para hacer el diálogo más fluido y natural.

Una de estas estructuras es una gramática que permite únicamente mencionar la palabra o conjunto reducido de palabras exacto, además del DTMF correspondiente, para poder ser reconocida la opción que el usuario quiere elegir.

Este tipo de gramática se utiliza en los diálogos iniciales, ya que de esta forma se va indicando al usuario cómo interaccionar con el sistema, pues mediante un *prompt* se le informa qué tiene que decir exactamente para acceder a cada una de las opciones.

Un ejemplo de este tipo de gramática se encuentra en la bienvenida del fichero *inicio.vxml*, que se muestra en la Figura 3.6.

```
<field name="opciones">
    <prompt>
            <break/>
            <value expr="saludo"/>
              Bienvenido al portal de voz del municipio de Alcorcón.
              Para consultar información diga información o marque 1.
              Para realizar gestiones y trámites diga gestiones o marque 2.
              Si desea realizar la encuesta semanal, diga encuesta o marque 3.
              Para acceder al buzón del ciudadano diga buzón o marque 4.
              Si en cambio desea ser atendido por un operador diga operador o marque 5.
    </prompt>

    <grammar type="application/grammar-xml" mode="voice" root="VOICE">
            <rule id="VOICE">
            <one-of>
                    <item> información </item>
                    <item> gestiones </item>
                    <item> encuesta </item>
                    <item> buzón </item>
                    <item> operador </item>
            </one-of>
            </rule>
    </grammar>

    <grammar type="application/grammar-xml" mode="dtmf" root="DTMF">
              <rule id="DTMF">
              <one-of>
                    <item> 1 </item>
                    <item> 2 </item>
                    <item> 3 </item>
                    <item> 4 </item>
                    <item> 5 </item>
              </one-of>
              </rule>
     </grammar>
            ….
</field>
```

Figura 3.6. Código con estructura de gramática XML

El otro tipo de estructura de gramática que se emplea funciona de tal forma que el conjunto de palabras que tiene que mencionarse para coincidir con una de las opciones

ofrecidas no sea único. Es decir, se han definido combinaciones de las posibles palabras y conjunto de palabras que el usuario podría decir para referirse a una misma opción a elegir.

Un ejemplo del funcionamiento de dicha estructura es el siguiente: si se quiere acceder al área de 'Circulación y Transporte', se podrá decir tanto 'circulación y transporte' como 'circulación' o como 'transporte', y se accederá a esa área. Esto aporta una mayor naturalidad al diálogo y facilita al usuario su navegación por los menús orales, además de evitar un mayor número de errores de reconocimiento.

Este tipo de gramáticas se utilizan especialmente en los menús finales, ya que el usuario ya sabe cómo interactuar con el sistema, y no haría falta darle tantas indicaciones de lo que tiene que decir. La Figura 3.7 muestra la estructura básica usada en este tipo de gramática.

```
<item>
    <one-of>
            <item>VOICE UTTERANCE</item>
            <item>VOICE UTTERANCE VARIATION 1</item>
            <item>VOICE UTTERANCE VARIATION 2</item>
            <item>DTMF KEY</item>
    </one-of>
    <tag>out.area="RETURN VALUE";</tag>
</item>
```

Figura 3.7. Código con estructura de gramática con diferentes alternativas definida para la aplicación

3.3.2.2 Gramáticas dinámicas

Las gramáticas dinámicas utilizan información que varía con el tiempo y suelen tratar gran cantidad de datos. Si se quisieran incluir en los ficheros *vxml* de forma manual habría que modificarlas cada poco tiempo, tarea que sería muy laboriosa debido al gran número de ítems de los que constan.

Los casos de este tipo de gramáticas en la aplicación son las gramáticas correspondientes a los nombres de los establecimientos de las páginas amarillas de Alcorcón (*gramatica_bares.*xml y *gramatica_tiendas.xml*), o las preguntas y respuestas de la encuesta municipal (*gramatica_encuesta.xml*).

Se ha creído conveniente que lo más eficaz es crear y eliminar automáticamente estas gramáticas. El procedimiento es el siguiente:

- Para crear estas gramáticas se utiliza la función *crear_gramatica* en los ficheros *crear_gramaticas.php* y *crear_encuesta.php* a la que se le pasa el nombre de la gramática que se quiere crear y los nombres para cada uno de los ítems. Estos nombres se obtienen de la base de datos a través de la sentencia SQL SELECT.

- Para abrir el archivo con el nombre de la gramática y colocar el puntero al final del archivo se utiliza la función *fopen* de PHP. Si este archivo no existe, se crea.

- Para ir escribiendo el contenido deseado en el fichero se usa la función *fwrite* de PHP.

- Para eliminar la gramática se usa la función *borrar_gramatica*, a la que se le pasa como parámetro el nombre de la gramática que se quiere borrar. En primer lugar se comprueba que la gramática indicada existe, utilizando para ello la función *file_exists* de PHP, y si es así, mediante la función *unlink* de PHP, se elimina dicho archivo.

El procedimiento anteriormente explicado y que se utiliza para el caso de las gramáticas correspondientes a los bares, restaurantes y tiendas es el mostrado en el código de la Figura 3.8.

```
<?php
header('Content-Type: text/html; charset=utf8');
include "/home/mariagj/public_html/informacion/conectarbasedatos2.php";
set_time_limit(500);

function crear_gramatica($nombre_gramatica, $array_nombres){
$gramatica=$nombre_gramatica;
$fp =fopen($gramatica,"a");

$contenido='<?xml version= "1.0"?>';
$write= fwrite($fp, $contenido);
fwrite($fp, chr(13).chr(10));

$contenido='<grammar
      xmlns="http://www.w3.org/2001/06/grammar"
      xml:lang="es-es"
      version="1.0" root="MYRULE">';
$write= fwrite($fp, $contenido);
fwrite($fp, chr(13).chr(10));

$contenido='<rule id="MYRULE">';
$write= fwrite($fp, $contenido);
fwrite($fp, chr(13).chr(10));

$contenido='<one-of>';
$write= fwrite($fp, $contenido);
fwrite($fp, chr(13).chr(10));

foreach ($array_nombres as $opciones){
$contenido='<item>';
$write= fwrite($fp, $contenido);
fwrite($fp, $opciones);

$contenido='<tag>out.MySlot="';
$write= fwrite($fp, $contenido);
fwrite($fp, $opciones);
$contenido='";</tag> </item>';
$write= fwrite($fp, $contenido);
fwrite($fp, chr(13).chr(10));
}
$contenido='</one-of>';
$write= fwrite($fp, $contenido);
fwrite($fp, chr(13).chr(10));
```

```
$contenido='</rule>';
$write= fwrite($fp, $contenido);
fwrite($fp, chr(13).chr(10));
$contenido='</grammar>';
$write= fputs($fp, $contenido);
fclose($fp);
}

function borrar_gramatica($nombre_gramatica){
    $gramatica=$nombre_gramatica;
    if (file_exists($gramatica)){
    unlink($gramatica);
    }
}

    $cnx=conectar2();
    $query="SELECT nombre FROM tiendas";
    $result=mysql_query($query) or die (mysql_error());
    $num=mysql_numrows($result);
    $i=0;
    if ($num == 0){
    echo "No existen tiendas en la base de datos ";
    }else{
    while ($i<$num){
    $nombres[$i]=mysql_result($result,$i,"nombre");
    $i++;
    }
    $nombres_tiendas=$nombres;
    }

    $query="SELECT nombre FROM bares";
    $result=mysql_query($query) or die (mysql_error());
    $num=mysql_numrows($result);
    $i=0;
    if ($num == 0){
    echo "No existen bares en la base de datos ";
    }else{
    while ($i<$num){
    $nombres[$i]=mysql_result($result,$i,"nombre");
    $i++;
    }
    $nombres_bares=$nombres;
    }
    mysql_close($cnx);

    $gramatica= 'gramatica_tiendas.xml';
    borrar_gramatica($gramatica);
    echo "gramática tiendas borrada";
    crear_gramatica($gramatica, $nombres_tiendas);
    echo "gramática tiendas creada";

    $gramatica= 'gramatica_bares.xml';
    borrar_gramatica($gramatica);
    echo "gramática bares borrada";
    crear_gramatica($gramatica, $nombres_bares);
    echo "gramática bares creada";
?>
```

Figura 3.8. Código para gestionar las gramáticas dinámicas (ejemplo del módulo de información de bares y restaurantes)

3.3.3 Gestión de bases de datos

3.3.3.1 Formato de almacenamiento

MySQL soporta distintas tecnologías de almacenamiento de datos, entre ellas destacan MyISAM e InnoDB:

- InnoDB es una tecnología de almacenamiento de datos de fuente abierta para MySQL, incluido como formato de tabla estándar en todas las distribuciones de MySQL AB a partir de las versiones 4.0. Su característica principal es que soporta transacciones de tipo ACID (Atomicity, Consistency, Isolation and Durability), bloqueo de registros e integridad referencial.

- MyISAM es la tecnología de almacenamiento de datos usada por defecto por el sistema administrador de bases de datos relacionales MySQL. Este tipo de tablas están basadas en el formato ISAM, incluyendo además nuevas extensiones.

Como el mejor rendimiento de uno u otro formato depende de la aplicación específica, en el desarrollo del proyecto se ha elegido en cada caso concreto de tabla el formato que proporciona la mejor relación de calidad.

De este modo, cuando la aplicación hace un uso elevado de INSERT y UPDATE se ha escogido InnoDB, ya que proporciona un aumento de rendimiento en cuanto a fiabilidad y consistencia con respecto a MyISAM. En aquellos casos en los que predominan las consultas SELECT a la base de datos, se utiliza MyISAM, ya que se obtiene mayor velocidad en general a la hora de recuperar datos.

3.3.3.2 Codificación

El **Estándar Unicode** es un estándar de codificación de caracteres diseñado para facilitar el tratamiento informático, transmisión y visualización de textos de múltiples lenguajes y disciplinas técnicas. El término Unicode proviene de los tres objetivos perseguidos: universalidad, uniformidad y unicidad.

Uno de los formatos Unicode es **UTF-8** (8-bit *Unicode Transformation Format*). Se trata de un formato de codificación de caracteres que utiliza símbolos de longitud variable. Está definido como estándar por la RFC 3629 de la *Internet Engineering Task Force* (IETF). Actualmente es una de las tres posibilidades de codificación reconocidas por Unicode y lenguajes web, y de las cuatro definidas en ISO 10646.

MySQL tiene dos conjuntos de caracteres Unicode: `ucs2` (UCS-2 Unicode) y `utf8` (UTF-8 Unicode). Usando estos conjuntos de caracteres se puede almacenar texto en unos 650 idiomas.

Las colaciones *`utf8_spanish_ci`* y *`utf8_spanish2_ci`* se corresponden con español moderno y español tradicional respectivamente. En ambas colaciones, '`ñ`' es una letra independiente, entre '`n`' y '`o`'. Además, para español tradicional '`ch`' es una letra, ordenada entre '`c`' y '`d`', y '`ll`' es una letra que se coloca entre '`l`' y '`m`'.

Para poder utilizar el idioma español moderno en la codificación de todos los datos que se usan en la aplicación del portal de voz, se almacena la información en tablas con cotejamiento `utf8_spanish_ci`.

3.3.3.3 Bases de datos utilizadas

Para el almacenamiento de toda la información necesaria en el portal de voz se utilizan dos bases de datos: 'mariagj_proyecto' y 'mariagj_info'.

- **Base de datos 'mariagj_proyecto'**

En esta base de datos se almacena la información relacionada con el apartado de gestiones y trámites. Para acceder a cualquiera de las opciones de ese apartado, se requiere la identificación del usuario, en nuestro caso, su DNI. Por tanto, almacenando la información personal en una base de datos expresamente dedicada para ello se proporciona confidencialidad y un acceso más restringido a esta información.

Esta base de datos consta de 4 tablas, tal y como se muestra en la captura de pantalla de la consola phpMyAdmin de la Figura 3.9.

Figura 3.9. Tablas de la base de datos 'mariagj_proyecto'

El contenido de las tablas de esta base de datos es el siguiente:

Tabla 'listado': contiene los números de identificación personal de ciudadanos que pertenecen a un listado de una determinada gestión o trámite.

Tabla 'trámites': incluye los expedientes de cada ciudadano clasificados por el número de identificación personal (DNI).

Tabla 'reserva': almacena la fecha y las horas en que se puede reservar una cierta instalación municipal. Para cada intervalo de tiempo se indica el estado de dicha instalación ('Libre' u 'Ocupado') y, en caso que sea 'Ocupado', el número de identificación del ciudadano que hizo dicha reserva.

Tabla 'cita': almacena la fecha y las horas en que se puede asistir a una cita en una cierta institución municipal. Los intervalos de tiempo repartidos tendrán el estado de

'Libre' u 'Ocupado'. En caso que el estado sea 'Ocupado' se almacena el número de identificación personal del ciudadano que tiene reservada cita para ese horario.

- **Base de datos mariagj_info**

En esta base de datos se almacena toda la información pública, tanto estática como dinámica de la ciudad. Ya que los datos no son confidenciales, esta base de datos no tiene acceso restringido.

Las 15 tablas que integran esta base de datos son las que se pueden ver en la captura de pantalla de la consola phpMyAdmin de la Figura 3.10.

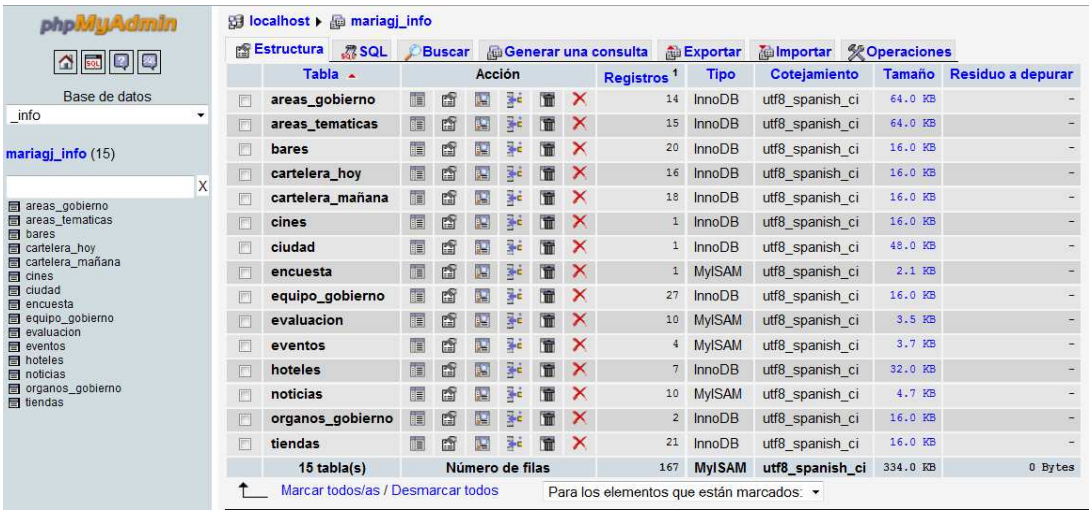


Figura 3.10. Tablas de la base de datos 'mariagj_info'

A continuación se detalla el contenido de cada una de dichas tablas:

Tabla 'areas_gobierno': incluye la información de cargos, competencias, actividades, y datos de contacto de las 14 concejalías del municipio de Alcorcón.

Tabla 'areas_tematicas': contiene la información, competencias y datos de contacto de las 15 áreas temáticas en que se divide la información del municipio de Alcorcón.

Tabla 'bares': incluye nombre, tipo, dirección, código postal, teléfono y distrito de cada uno de los bares, cafés y restaurantes pertenecientes a las páginas amarillas de la ciudad.

Tabla 'cartelera_hoy': almacena los títulos y pases de las películas de la cartelera del cine de Alcorcón para el día actual.

Tabla 'cartelera_mañana': almacena los títulos y pases de las películas de la cartelera del cine de Alcorcón para el día siguiente.

Tabla 'cines': incluye nombre, dirección, teléfonos, precios y transportes del cine de Alcorcón.

Tabla ‘ciudad’: contiene datos, historia y accesos de la ciudad.

Tabla ‘encuesta’: contiene el identificador de la encuesta, la pregunta, las 4 posibles respuestas, los resultados actuales para cada una de las respuestas y el número total de votos.

Tabla ‘equipo_gobierno’: engloba los grupos políticos del municipio de Alcorcón con los nombres y los cargos de cada uno de sus componentes.

Tabla ‘eventos’: almacena el título, área, descripción, fecha y lugar para cada evento de Alcorcón.

Tabla ‘hoteles’: contiene el nombre, la dirección y el teléfono de cada uno de los hoteles y hostales de la ciudad.

Tabla ‘noticias’: almacena fecha, título y subtítulo de cada una de las noticias de Alcorcón

Tabla ‘organos_gobierno’: contiene los órganos de gobierno de Alcorcón y sus descripciones.

Tabla ‘tiendas’: incluye nombre, tipo, dirección, código postal, teléfono y distrito de cada una de las tiendas de las páginas amarillas de la ciudad.

3.3.3.4 Acceso a las bases de datos

Para acceder a una base de datos, en primer lugar se debe crear una conexión con esa base de datos mediante la función *mysql_connect* de PHP, pasándole como parámetros las variables correspondientes al servidor web, nombre de usuario y contraseña para esa base de datos en concreto. La implementación de este código se muestra en la Figura 3.11.

```
function conectar2(){
$username="mariagj_maria";
$password="100025080";
$database="mariagj_info";
$cnx=mysql_connect('localhost',$username,$password);
mysql_select_db($database) or die (mysql_error());
mysql_query ("SET NAMES 'utf8'");
return $cnx;
}
```

Figura 3.11. Código para realizar la conexión con una base de datos MySQL desde PHP

Una vez se ha creado la conexión, se utilizan sentencias MySQL para acceder a las distintas tablas y obtener (SELECT), insertar (INSERT), modificar (UPDATE) o borrar (DELETE) sus datos.

Al terminar de utilizar la base de datos, la conexión anteriormente creada debe cerrarse mediante la función *mysql_close* de PHP.

A modo de ejemplo, se presenta en la Figura 3.12 el código de acceso a la tabla 'areas_tematicas' y en concreto a la columna 'informacion', donde cada área temática viene especificada por una variable.

```
<?php
$area=$_REQUEST["area"];
$cnx=conectar2();
$query="SELECT informacion FROM areas_tematicas WHERE area='".$area."'";
$result=mysql_query($query) or die (mysql_error());
$num=mysql_numrows($result);
$informacion=mysql_result($result,$i,"informacion");
echo $informacion;
mysql_close($cnx);
?>
```

Figura 3.12. Código para acceder a la tabla 'areas_tematicas'

3.3.4 Acceso a las páginas web

Para poder tener almacenada la información dinámica actualizada (anteriormente explicada en el apartado 3.3.1.2), se necesita cargarla desde la web correspondiente hasta la base de datos cada cierto periodo de tiempo.

Este procedimiento se realiza mediante el fichero *cargar_datos.php* (ver código de la Figura 3.13) en el que, mediante las funciones *cargar_noticias*, *cargar_eventos* y *cargar_cartelera*, se analizan sintácticamente las páginas web en busca de la información necesaria para almacenarla en la base de datos y posteriormente proporcionársela al usuario cuando pregunte por ella. Cada una de estas funciones se explica en detalle en el apartado correspondiente a su módulo en la aplicación (Capítulo 4).

```
<?php
header('Content-Type: text/html; charset=utf8');
include "/home/mariagj/public_html/informacion/cargar_informacion.php";
include "/home/mariagj/public_html/informacion/ciudad/cargar_cartelera.php";

set_time_limit(300);

//Se cargan las noticias desde la página web del Ayuntamiento a la base de datos
$url_noticias='http://www.ayto-alcorcon.es/portal/rss?detalle=noticias';
cargar_noticias($url_noticias);

//Se cargan los eventos desde la página web del Ayuntamiento a la base de datos
$url_eventos= "http://www.ayto-alcorcon.es/portal/agendaeventos/index";
cargar_eventos($url_eventos);

/* Se carga la cartelera del cine de Alcorcón para el día actual y para el día siguiente
 * desde la página web www.ecartelera.com a la base de datos
```

```
 */
$url1 = "http://www.ecartelera.com/cines/63,0,1.html";
$url2= "http://www.ecartelera.com/cines/63,20110406,1.html";
$dia1="cartelera_hoy";
$dia2="cartelera_mañana";
cargar_cartelera($url1,$dia1);
cargar_cartelera($url2,$dia2);

?>
```

Figura 3.13. Código para cargar datos de noticias, eventos y cartelera

La Figura 3.14 muestra un ejemplo de cómo se realiza esta recopilación de información y su almacenamiento para el caso de los eventos municipales. En primer lugar, se realiza una conexión con la base de datos 'marya.gj_info'. A continuación, se borra la tabla con los eventos antiguos y se crea una nueva tabla 'eventos' para almacenar los eventos actuales. Mediante la función *file_get_contents* de PHP se obtiene el contenido de la web de eventos de Alcorcón. Finalmente, se analiza ese contenido y se extrae la información necesaria (id, título, área, descripción y fecha de cada evento) para almacenarla en las celdas correspondientes de la base de datos.

```
function cargar_eventos($url){

$cnx=conectar2();
$id=1;
$query="DROP TABLE eventos";
mysql_query($query);

$query="CREATE TABLE eventos ( id int(3) NOT NULL, titulo TEXT, area VARCHAR (200), descripcion
MEDIUMTEXT, fecha TEXT, PRIMARY KEY (id) )
DEFAULT CHARSET=utf8 COLLATE=utf8_spanish_ci";

mysql_query($query);

$xml2 = file_get_contents($url);

//transformamos las entidades
$xml2=str_replace("&aacute;", "á", $xml2);
$xml2=str_replace("&eacute;", "é", $xml2);
$xml2 = str_replace("&aacute;", "á", $xml2) ;
$xml2 = str_replace("&ntilde;", "ñ", $xml2) ;
$xml2 = str_replace("&eacute;", "é", $xml2) ;
$xml2 = str_replace("&iacute;", "í", $xml2) ;
$xml2 = str_replace("&oacute;", "ó", $xml2) ;
$xml2 = str_replace("&uacute;", "ú", $xml2) ;
$xml2 = str_replace("&Aacute;", "Á", $xml2) ;
$xml2 = str_replace("&Ntilde;", "Ñ", $xml2) ;
$xml2 = str_replace("&Eacute;", "É", $xml2) ;
$xml2 = str_replace("&Iacute;", "Í", $xml2) ;
$xml2= str_replace("&Oacute;", "Ó", $xml2) ;
$xml2 = str_replace("&Uacute;", "Ú",$xml2) ;
$xml2 = str_replace(" ", "",$xml2);
$xml2 = str_replace("&", "" ,$xml2);
```

```
$patron_titulo='h2 class="tit"';
$patron_titulofin='h2';
$patron_area='h3 class="tit"';
$patron_areafin='h3';
$patron_descripcion='div class="modContent"';
$patron_descripcionfin='div';
$patron_fechas='ul class="listType01"';
$patron_fechasfin='ul';

$titulos_encontrado=
preg_match_all('#<'.$patron_titulo.'(?:\s+[^>]+)?>'.'(.*?)</'.$patron_titulofin.'>#s', $xml2, $titulos);
$area_encontrada =
preg_match_all('#<'.$patron_area.'(?:\s+[^>]+)?>'.'(.*?)</'.$patron_areafin.'>#s', $xml2, $areas);
$descripcion_encontrada=
preg_match_all('#<'.$patron_descripcion.'(?:\s+[^>]+)?>'.'(.*?)</'.$patron_descripcionfin.'>#s',
$xml2, $descripciones);
$titulos_encontrado=
preg_match_all('#<'.$patron_fechas.'(?:\s+[^>]+)?>'.'(.*?)</'.$patron_fechasfin.'>#s', $xml2,
$fechas);

for($i=0;$i<count($titulos[0]);$i++) {

$titulo=strip_tags($titulos[0][$i]);
$titulo=limpiar($titulo);

$area=strip_tags($areas[0][$i]);
$area=limpiar($area);

$descripcion=strip_tags($descripciones[0][$i]);
$descripcion=limpiar($descripcion);

$fecha=strip_tags($fechas[0][$i]);
$fecha=limpiar($fecha);

$query="INSERT INTO eventos (id, titulo, area, descripcion, fecha) VALUES ('$id',
'$titulo','$area','$descripcion','$fecha')";
$id++;
mysql_query($query);

}
mysql_close($cnx);
}
```

Figura 3.14. Código para obtener la información de los eventos

3.3.5 Eventos VoiceXML

En VoiceXML para capturar y tratar los eventos, independientemente de su origen, se dispone del elemento '*catch*'. Dentro de este elemento se especifican las acciones a realizar en caso que el evento sea capturado.

El intérprete VoiceXML proporciona manejadores de *catch* implícitos por defecto para los eventos *noinput*, *nomatch* y *help*. En la aplicación desarrollada interesa tratar estos tres eventos de forma específica y mostrar un mensaje para cada uno de ellos.

- Evento *noinput*:
  Este evento ocurre cuando el usuario no responde dentro del intervalo del tiempo de espera.

- Evento *nomatch*:
  Este evento se lanza cuando el usuario dice algo pero no es reconocido por ninguna de las gramáticas activas.

- Evento *help*:
  Este evento ocurre en el caso en que el usuario pide ayuda.

El código que se utiliza para tratar estos eventos se muestra en la Figura 3.15.

```
<property name="universals" value="help"/>
…..
<!—El usuario no ha dicho nada -->
    <noinput>
      Lo siento, no he oído nada. Por favor, inténtelo de nuevo.
     <reprompt/>
    </noinput>

    <!—El usuario ha dicho algo no reconocido por las gramáticas-->
    <nomatch>
     Lo siento, no he entendido qué ha dicho. Por favor, inténtelo de nuevo.
    <reprompt/>
    </nomatch>

    <help>
    <prompt>
     Para consultar información diga información o marque 1.
     Para realizar gestiones y trámites diga gestiones o marque 2.
     Si desea realizar la encuesta semanal, diga encuesta o marque 3.
     Para acceder al buzón del ciudadano diga buzón o marque 4.
     Si en cambio desea ser atendido por un operador diga operador o marque 5.
    </prompt>
    <reprompt/>
    </help>
      …..
```

Figura 3.15. Código para tratar los eventos *nomatch*, *noinput* y *help*

3.3.6 Propiedades VoiceXML

En VoiceXML el elemento '*property*' establece un valor de propiedad. Las propiedades se utilizan para definir los valores que afectan al comportamiento de la plataforma.

Las propiedades se pueden definir para toda la aplicación, para todo el documento en el nivel <vxml>, para un determinado diálogo en un formulario o menú, o para un elemento de forma particular.

Para la implementación de la aplicación se han utilizado varias de estas propiedades, de las cuales se analizan a continuación las siguientes:

- **'Confidencelevel'**

Esta propiedad permite ajustar la exactitud que la expresión de un usuario debe tener para ser reconocida por la gramática activa. Su valor por defecto es de '.45'. Si se disminuye este valor, por ejemplo a '0.1', se reconocería cualquier entrada de usuario, mientras que si se ajusta a '1.0', entonces toda expresión de usuario sería considerada 'nomatch'. El valor que se ha fijado para esta propiedad en los ficheros VoiceXML de la aplicación es de '0.3'.

- **'Sensitivity'**

Esta propiedad permite ajustar la sensibilidad del intérprete frente a la entrada nula. Su valor por defecto es de '0.5'. Si se aumenta este valor, el intérprete será más sensible, mientras que si se disminuye el intérprete será menos sensible. Esta propiedad puede ser útil si el usuario final accede a la aplicación en un entorno muy ruidoso. Se recomienda utilizar esta propiedad con cuidado, ya que puede tener un efecto perjudicial sobre la precisión del reconocimiento. En el portal de voz se ajusta este valor a '0.3'.

- **'Documentfetchhint'**

La propiedad 'documentfetchint' define el comportamiento para la búsqueda (*fetching*) del documento dentro de la aplicación. Cuando su valor se establece a '*safe*', el documento será buscado sólo cuando se necesite. En cambio, cuando se ajusta a '*prefetch*' (por defecto), entonces el contenido será buscado con antelación para optimizar los tiempos de respuesta. En la aplicación desarrollada se hace uso del valor por defecto de esta propiedad.

- **'Grammarfetchhint'**

Esta propiedad especifica cuándo una gramática externa debe ser buscada. Su valor por defecto, '*prefetch*', significa que la gramática será buscada siempre que la página VoiceXML se cargue. En cambio, el valor '*safe*' establece que sólo se buscará la gramática cuando ésta sea utilizada. En la aplicación se utiliza su valor por defecto.

Capítulo 4

Descripción detallada de los módulos del sistema

En este capítulo se analizan en detalle cada uno de los módulos de la aplicación desarrollada: se describen las funcionalidades que proporcionan, se comentan sus arquitecturas, se muestran gráficamente sus esquemas de flujo de información y, finalmente, se exponen sus escenarios de uso.

4.1 Módulo Inicio

4.1.1 Funcionalidad

El ***módulo Inicio*** implementa el primer diálogo que se le proporciona al usuario al llamar al número de teléfono del portal de voz. Las opciones que el usuario puede elegir se reparten en 5 módulos bien diferenciados según el tipo de interacción y de datos que se van a proporcionar: información, gestiones y trámites, encuesta, buzón del ciudadano, y operador. Por tanto, es en este *módulo inicio* donde se bifurca el diálogo entre el resto de módulos. Esta división de la aplicación se puede observar en la Figura 4.1.

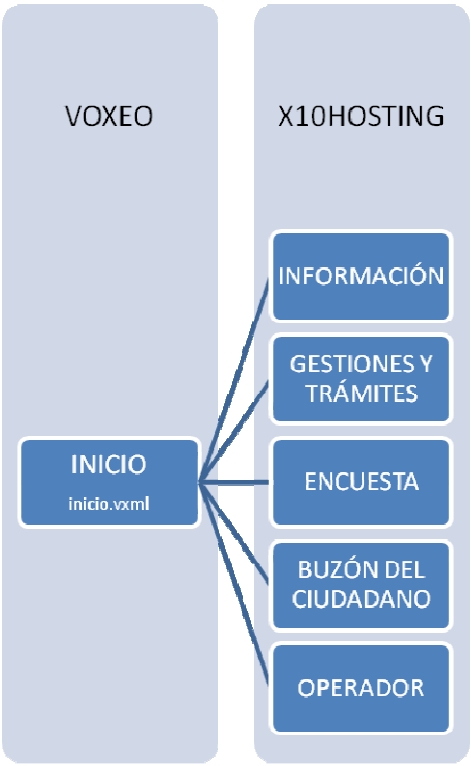


Figura 4.1. Módulos de la aplicación desarrollada

4.1.2 Arquitectura

El primer fichero de la rutina, *inicio.vxml*, está alojado en el servidor Voxeo. Las principales acciones que se llevan a cabo en él son reproducir un mensaje de bienvenida al usuario y dar a elegir al usuario la acción que desea realizar.

- **Saludo**

El saludo se implementa mediante un script (Figura 4.2) en el que, dependiendo de la hora que sea, el saludo será personalizado a “Buenos días”, “Buenas tardes” o “Buenas noches” seguido de “Bienvenido al portal de voz del municipio de Alcorcón”.

```
<var name="horas"/>
<var name="saludo"/>

<block>
<script>
        var d = new Date();
        horas=d.getHours();
        saludo="Buenos días.";
        if (horas>14) saludo="Buenas tardes.";
        if (horas>20) saludo="Buenas noches.";
</script>
</block>
```

Figura 4.2. Script que implementa el saludo de bienvenida

- **Elección de la acción**

A continuación, al usuario se le reproducen las opciones que tiene para elegir. Una vez que el usuario elija una de estas opciones, se continúa la rutina en el fichero correspondiente a esta opción elegida, ya almacenado en el servidor web externo (x10hosting en nuestro caso).

- **Fichero principal**

El fichero *inicio.vxml* además de contener la gramática desde donde parten los diálogos hacia los demás módulos, contiene una gramática que permite volver a este diálogo inicial desde cualquier punto de este fichero. Para que esto ocurra, el usuario debe decir “inicio” o “principio”, de lo que es informado en caso que pida ayuda. El código que lo implementa es el que se presenta en la Figura 4.3.

```
<link next="#inicio">
    <grammar type="text/gsl">
            [inicio principio]
    </grammar>
</link>
```

Figura 4.3. Gramática para volver al formulario inicio

El flujo de datos y el esquema de ficheros utilizados en este módulo se pueden observar en la Figura 4.4. El usuario accede a la aplicación, y desde el fichero inicio.vxml se le da la bienvenida y se le proporciona un menú. Dependiendo de la opción seleccionada, la ejecucción continúa en un fichero u en otro.

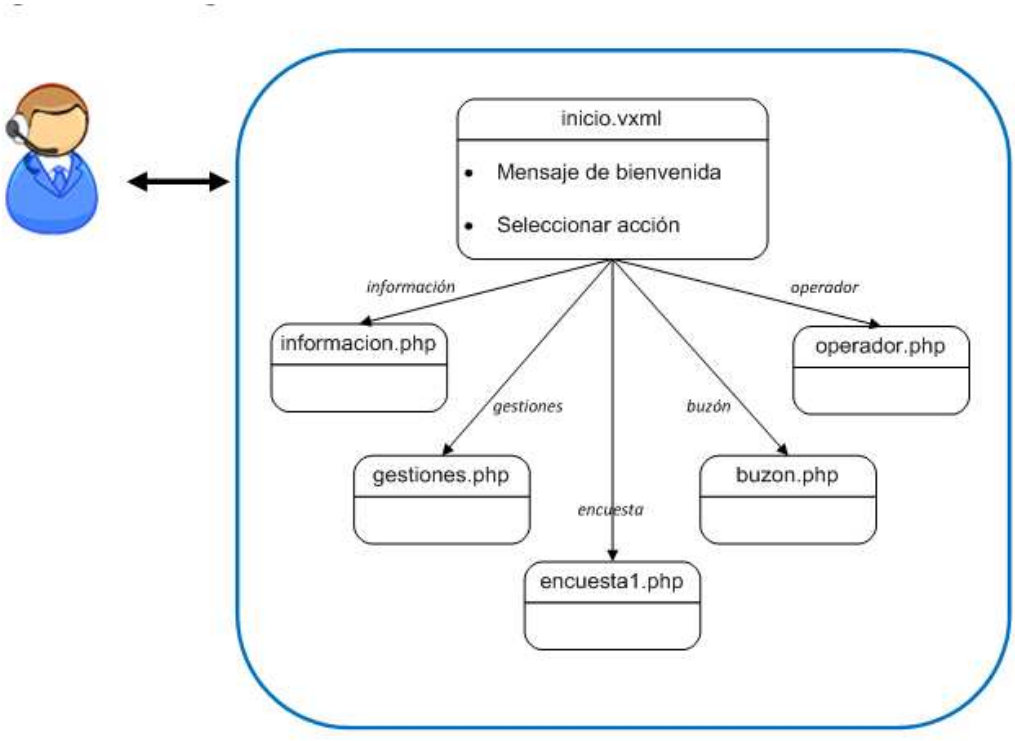


Figura 4.4. Flujo de datos del *módulo Inicio*

4.1.3 Escenarios de uso

En la Figura 4.5 se muestra cómo se desarrolla el primer diálogo de la aplicación. 'S' indica el diálogo correspondiente al sistema y 'U' el diálogo correspondiente al usuario.

S:	Buenos días. Bienvenido al portal de voz del municipio de Alcorcón. Para consultar información diga información o marque 1. Para realizar gestiones y trámites diga gestiones o marque 2. Si desea realizar la encuesta semanal, diga encuesta o marque 3. Para acceder al buzón del ciudadano diga buzón o marque 4. Si en cambio desea ser atendido por un operador diga operador o marque 5.
U:	Encuesta.
S:	A continuación, usted podrá realizar la encuesta semanal y obtener los resultados provisionales. …

Figura 4.5. Escenario de uso del *módulo Inicio*

4.2 Módulo Información

En el ***módulo Información*** el usuario puede acceder a toda la información del municipio de Alcorcón. La información, según su tipo, se divide en 6 grandes submódulos bien diferenciados y clasificados de tal forma que facilitan el acceso del usuario a la información que esté buscando:

Ayuntamiento: Proporciona toda información relacionada con el Equipo de Gobierno, los Órganos de Gobierno y las Áreas de Gobierno municipales.

Ciudad: En este submódulo se accede a la información referente a Alcorcón como ciudad. Se pueden consultar los datos de la ciudad, la historia, los accesos, y unas páginas amarillas compuestas por los bares, cafés, restaurantes, tiendas, hostales, hoteles y cines (incluida la información sobre la cartelera) del municipio.

Áreas temáticas: Cualquier otro tipo de información que busque el usuario se ha repartido y clasificado en una de las 15 áreas que forman esta sección. Para cada una de estas áreas se proporciona información general, competencias y datos de contacto. Simplemente con la incorporación de más información estática en la base de datos se podrían añadir más funcionalidades y áreas en este módulo de la aplicación.

Noticias: Proporciona las noticias del municipio. Se facilita la fecha, título y subtítulo de cada noticia.

Eventos: Reproduce el listado de eventos del municipio. Se proporciona el área temática, título, fecha, lugar y descripción de cada evento.

Información meteorológica: El usuario puede obtener la información meteorológica actual del municipio y la previsión para los siguientes dos días.

Esta división por tipo de información se puede observar gráficamente en la Figura 4.6.

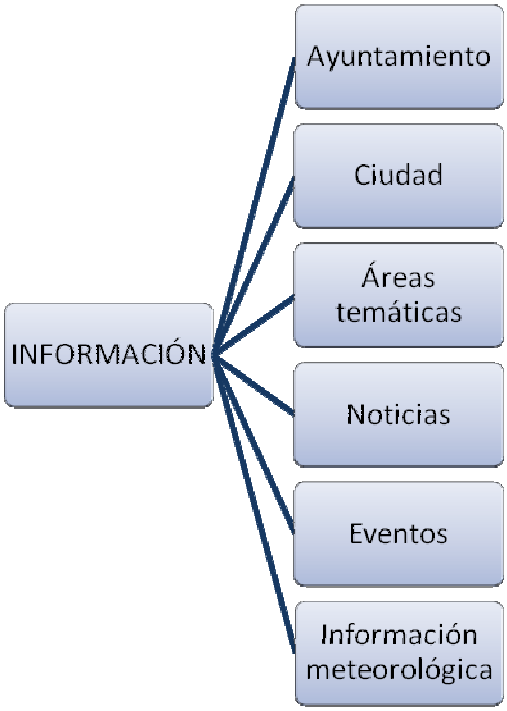


Figura 4.6. *Módulo Información* y su división en submódulos

4.2.1 Submódulo Ayuntamiento

4.2.1.1 Funcionalidad

En el ***submódulo Ayuntamiento*** se gestiona el acceso a toda la información relacionada con el Ayuntamiento. El usuario puede elegir entre consultar información acerca del Equipo de Gobierno, sobre los Órganos de Gobierno o sobre las distintas Áreas de Gobierno. Cada una de estas secciones se divide en subsecciones con información más concreta para poder, de esta forma, proporcionar de una manera más exacta la información requerida por el usuario.

La división de la información del *submódulo Ayuntamiento* se puede observar en la Figura 4.7.

Figura 4.7. Esquema *del submódulo Ayuntamiento*

4.2.1.2 Arquitectura

Desde el fichero *información_ayuntamiento.php* el diálogo se bifurca en las tres posibles opciones anteriormente comentadas. Estas 3 opciones descritas con detalle son:

- Equipo de Gobierno (*ayuntamiento_equipo.php*).

En este fichero se tiene la opción de oír, mediante la reproducción de un fichero *.wav*, el saludo del Alcalde de Alcorcón, o acceder al apartado del Equipo de Gobierno, en donde se ofrece la información de los miembros que forman el Gobierno de Alcorcón, su partido político y sus cargos.

- Órganos de Gobierno (*ayuntamiento_organos.php*).

En este fichero se elige entre acceder a información del Pleno municipal de Alcorcón o a información sobre la Junta de Gobierno de Alcorcón para a continuación oír reproducida la información que ha sido seleccionada.

- Áreas de Gobierno (*ayuntamimento_areas.php*).

En esta sección se puede acceder a las distintas áreas de gobierno, que están repartidas según su función:

Servicios a la ciudad: lo conforman aquellas concejalías encargadas de la atención vecinal más directa, como son Cultura, Educación, Deportes, Bienestar social, Infancia, Adolescencia y Juventud, Mujer, Mayores y Salud y mercados. En el fichero *concejalías_servicios.php* se elige una de estas concejalías pertenecientes al área servicios a la ciudad.

Desarrollo Territorial: está compuesta por aquellas concejalías encargadas del desarrollo urbano y la sostenibilidad ambiental, y lo forman las concejalías de Urbanismo y Medio Ambiente, Vivienda (EMGIASA), Servicios Generales (ESMASA), y Parques, Jardines y Mantenimiento. En el fichero *concejalías_desarrollo.php* se debe seleccionar una de las concejalías del área desarrollo territorial para obtener más información sobre ella.

Hacienda y Administración general: abarca aquellas concejalías cuya misión es gestionar los recursos humanos, económicos y patrimoniales de la ciudad. Comprende las concejalías de Hacienda y Patrimonio, RRHH, Circulación y Seguridad, y Nuevas tecnologías, Promoción Empresarial, Formación y Empleo. En el fichero *concejalías_administracion.php* se selecciona una de las concejalías del área Hacienda y Administración General.

A continuación, una vez seleccionada la concejalía en la que se está interesado, en el fichero *concejalias_general.php* se pregunta al usuario por el tipo de información que desea de dicha concejalía:

- Competencias de la concejalía

- Actividades de la concejalía.

- Cargos de la concejalía.

- Datos de contacto de la concejalía.

Se accede a la información solicitada en la tabla '*organos_gobierno*' de la base de datos *mariagj_info* y se proporciona al usuario. Finalmente, se accede al fichero *ayuntamiento_enlazar.php* en el que se pregunta sobre la siguiente acción:

1- Consultar otros datos de esta misma concejalía. Si el usuario está interesado en una concejalía es muy probable que quiera consultar más datos sobre esa concejalía. Para ello se ha guardado en una variable el nombre de la concejalía que se acaba de consultar, para no tener que volver a preguntar al usuario por la concejalía, y retornar así al fichero de *concejalias_general*.php con el valor de dicha concejalía.

2- Volver al menú de información del Ayuntamiento.

3- Volver al menú de información general.

4- Volver al inicio del portal de voz.

5- Salir de la aplicación.

El flujo de información de este submódulo se puede visualizar en la Figura 4.8.

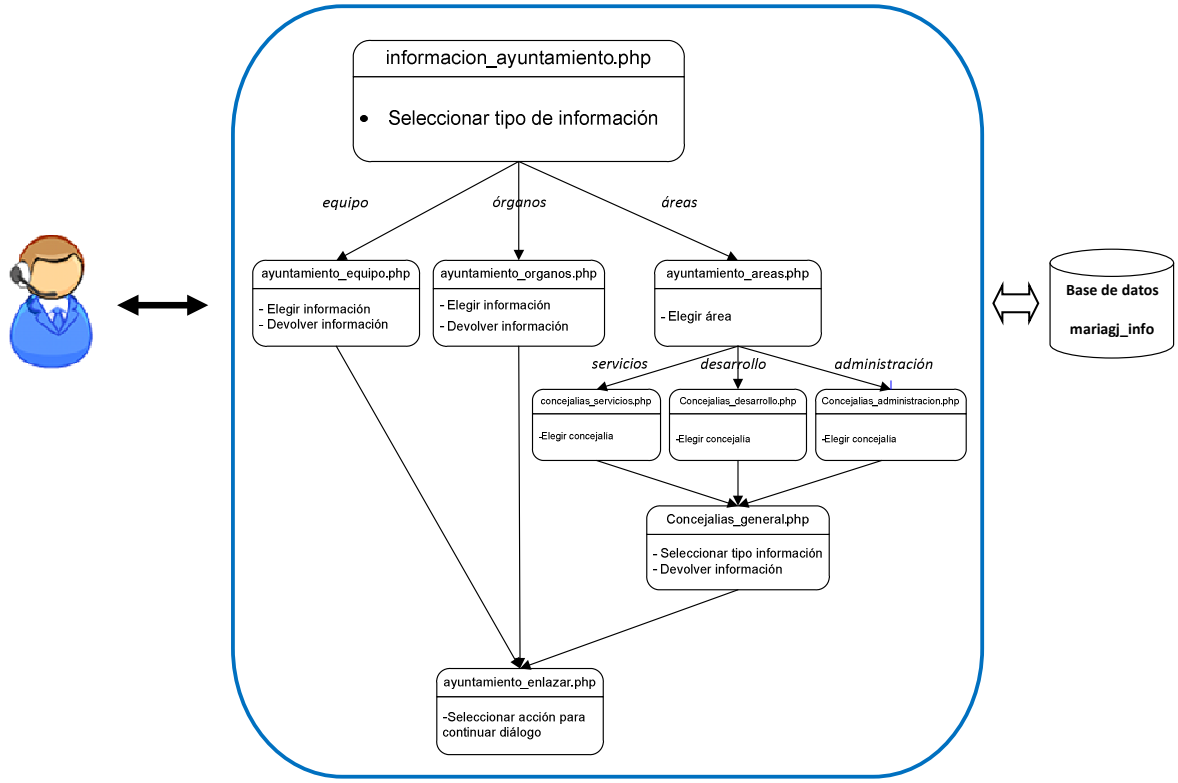


Figura 4.8. Flujo de datos del *submódulo Ayuntamiento*

4.2.1.3 Escenarios de uso

En la Figura 4.9 se muestra cómo se desarrollaría el diálogo entre un usuario (‘U’) y el sistema (‘S’) en este submódulo *Ayuntamiento*.

S:	Usted ha accedido al apartado de información sobre el Ayuntamiento de Alcorcón. Para consultar información sobre el Equipo de Gobierno diga equipo o marque 1. Para consultar información sobre los Órganos de Gobierno diga órganos o marque 2. O si desea consultar información sobre alguna de las distintas Áreas de Gobierno diga áreas o marque 3.
U:	Áreas.
S:	Las áreas de gobierno se dividen entre: Servicios a la ciudad, que lo conforman aquellas concejalías encargadas de la atención vecinal más directa. Desarrollo Territorial, que está compuesto por aquellas concejalías encargadas del desarrollo urbano y la sostenibilidad ambiental. Y Hacienda y Administración general, que abarca aquellas concejalías cuya misión es gestionar los recursos humanos, económicos y patrimoniales de la ciudad.
S:	¿Qué área de gobierno quiere consultar?

1- Servicios a la ciudad,
2- desarrollo territorial, o
3- administración general.

U: Administración.

S: Elija una concejalía para más información:
Diga Hacienda y Patrimonio o marque 1.
Diga Recursos humanos o marque 2.
Diga Nuevas tecnologías o marque 3.

U: Hacienda.

S: ¿Qué tipo de información quiere de la concejalía?
1-Competencias.
2-Actividades.
3-Cargos.
4-Datos de contacto.

U: Competencias.

S: Accedemos a la información de competencias de esta concejalía.
Concejalía de Hacienda.
Intervención general municipal.
Unidad central de contabilidad y tesorería.
Oficina presupuestaria.
Agencia municipal tributaria.
Consejos Sectoriales, y mecanismos de participación ciudadana y vecinal en los asuntos relacionados con esta Dirección General.

Concejalía de Patrimonio.
Inventario Municipal de Bienes. Su comprobación, rectificación y presentación para la aprobación por el órgano competente. La reivindicación de bienes municipales. El aprovechamiento de los bienes comunales. La adquisición por cualquier título jurídico-privado o administrativo de bienes inmuebles. La enajenación o venta, disposición por cualquier título, arrendamiento, cesión, constitución de cualquier derecho real, gravamen, etc. de cualquier bien perteneciente al patrimonio municipal. La afectación y desafectación de bienes. La creación de cargas o gravámenes sobre cualquier bien municipal. Adquisición del material fungible para el funcionamiento de oficinas y del mobiliario de éstas. Consejos Sectoriales, y mecanismos de participación ciudadana y vecinal en los asuntos relacionados con esta Dirección General.

S: ¿A qué menú desea ir ahora?
1- Esta concejalía
2- Ayuntamiento.
3- Información.
4- Inicio del portal de voz.
5- Salir de la aplicación.

U: Esta concejalía.

S: Volvemos al apartado de información sobre esta concejalía.

S: ¿Qué tipo de información quiere de la concejalía?
1-competencias.
2-actividades.
3-cargos.
4-datos de contacto.

U: Datos de contacto.

S: Accedemos a los datos de contacto de esta concejalía.
Concejalía de Hacienda.
Tesorería.
C/ La Iglesia, 7 1ª Planta.
Teléfono: 91 664 81 91.
Fax: 91 664 85 73.
tstesoreria@ayto-alcorcon.es.
Unidad de recaudación.
C/ Mayor, 19 1ª Planta.
Teléfono Atención General: 91 664 81 00.
Fax: 91 664 82 83.
recaudación@ayto-alcorcon.es.
Inspección municipal de tributos.
C/ Mayor, 19 1ª Planta.
Teléfonos: 91 664 83 12 / 8562 / 63. Fax. 91 664 85 73.
inspecciontributos3@ayto-alcorcon.es.
Departamento de Gestión tributaria.
Pza. del Tejar, 8.
Horario de atención al público: de lunes a viernes de 8,30 a 14 horas.
Teléfonos: Atención general: 91 664 81 00.

Concejalía de Patrimonio.
Lunes a Viernes: 8:00 a 15:00.
Teléfono: 91 664 81 60.
Fax: 91 664 81 89.
Dirección: C/ La Iglesia, 7.
sechacienda@aytio-alcorcon.es.
Bus: 513; L 2.

S: ¿A qué menú desea ir ahora?
1- Esta concejalía
2- Ayuntamiento.
3- Información.
4- Inicio del portal de voz.
5- Salir de la aplicación.

U: Salir.

S: Usted ha elegido salir de la aplicación. Gracias por utilizar nuestro portal de voz. Hasta pronto.

Figura 4.9. Escenario de uso del *submódulo Ayuntamiento*

4.2.2 Submódulo Ciudad

4.2.2.1 Funcionalidad

En el ***sumódulo Ciudad*** el ciudadano tiene acceso a la información relacionada con Alcorcón como ciudad. Se pueden consultar sus datos geográficos y demográficos, su historia, cómo llegar a la ciudad y unas páginas amarillas donde se encuentran los datos de contacto de los bares, cafés, restaurantes, tiendas, hoteles y hostales de la ciudad, además del cine y su cartelera para el día actual y para el día siguiente. La información se distribuye en los apartados que se observan en la Figura 4.10.

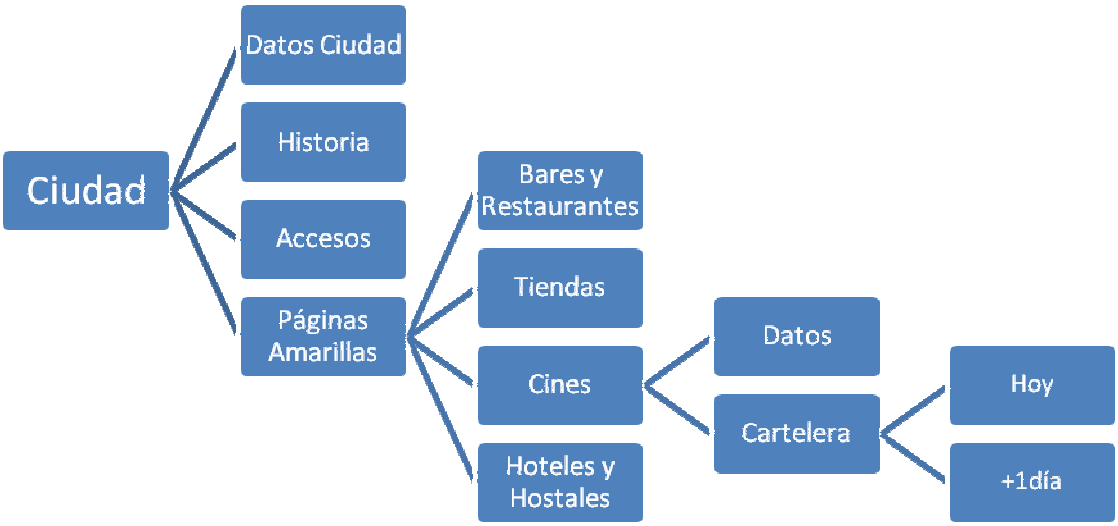


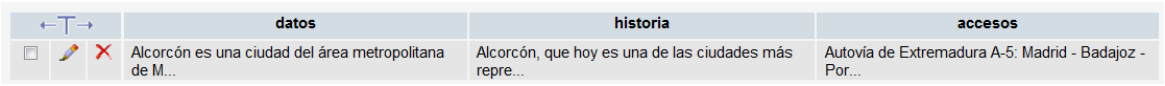
Figura 4.10. Esquema del *submódulo Ciudad*

4.2.2.2 Arquitectura

En un primer fichero, *informacion_ciudad.php*, se solicita al usuario el tipo de información que quiere consultar entre ‘datos de la ciudad’, ‘historia’, ‘accesos’ o ‘páginas amarillas’.

Para la información ‘datos de la ciudad’, ‘historia’ y ‘accesos’ se sigue el mismo procedimiento: se accede a la información correspondiente dentro de tabla ‘ciudad’ de la base de datos ‘*mariagj_info’*, y se proporcionan estos datos al usuario.

La tabla ‘ciudad’ con las columnas correspondientes a estos tres tipos de información se observa en la Figura 4.11.



	datos	historia	accesos
	Alcorcón es una ciudad del área metropolitana de M...	Alcorcón, que hoy es una de las ciudades más repre...	Autovía de Extremadura A-5: Madrid - Badajoz - Por...

Figura 4.11. Campos de la tabla ‘ciudad’

Un ejemplo del acceso y reproducción de esta información al usuario se muestra en el código de la Figura 4.12, en el que se proporcionan los datos de la historia de Alcorcón.

```
<?php
echo "Historia de Alcorcón:";
$cnx=conectar2();
$query="SELECT historia FROM ciudad";
$result=mysql_query($query);
$num=mysql_numrows($result);
$i=0;
$historia[$i]=mysql_result($result,$i,"historia");
print_r($historia[$i]);
mysql_close($cnx);
?>
```

Figura 4.12. Código para acceder a la historia de la ciudad

Para gestionar la información de las 'páginas amarillas' se accede al fichero *paginas_amarillas.php*, en el que se elige la información a consultar entre 'bares y restaurantes', 'tiendas', 'cines' y 'hoteles'. La rutina seguida para cada uno de ellos es diferente y se explica a continuación:

- **Bares, cafés y restaurantes**

En el fichero *bares.php* se pregunta al usuario si conoce el nombre del establecimiento en que está interesado. Si es así, se le pide el nombre de dicho establecimiento y se utiliza la gramática *gramatica_bares.xml* creada dinámicamente con el fichero *crear_gramatica.php* con los nombres de los bares y restaurantes almacenados en la base de datos, tal y como se explicó en el apartado 3.3.2.2. Una vez reconocido el nombre del establecimiento, se accede al fichero *nombre_bar.php*, en el que se proporciona al usuario los datos de contacto de dicho establecimiento (tipo de establecimiento, dirección y teléfono).

Si en cambio, el usuario lo que busca son ciertos establecimientos en una zona, se le da a elegir el tipo de establecimiento (bares y cafés o restaurantes) y la zona de Alcorcón en la que esté interesado (distrito norte, distrito sur o distrito centro). Las elecciones del usuario son almacenadas en sendas variables que se pasan al fichero *listado_bares.php*. En este fichero se accede a la base de datos y se comprueba que existen establecimientos del tipo y en la zona que el usuario ha seleccionado.

Si no existen establecimientos de estas características, se informa al usuario, y se le pide que lo intente de nuevo con otros criterios de búsqueda, volviendo entonces al formulario 'listado' en el fichero *bares.php*.

Si existen establecimientos con las características elegidas en la zona seleccionada, se informa del número de establecimientos existentes y se comunica al usuario que se reproducirán uno a uno los nombres de los establecimientos, y en el momento en que desee más información detallada de alguno de ellos (dirección y teléfono), diga sí o pulse 1.

Una vez llegado al final de la lista de establecimientos con las características elegidas, se informa de ello al usuario y se le pregunta si desea escuchar el listado de nuevo. En caso contrario se accede el fichero *ciudad_enlazar.php*.

- **Tiendas**

Para la información sobre tiendas se procede de manera similar a los bares y restaurantes. Si el usuario conoce el nombre concreto de la tienda, se utiliza el fichero *tiendas.php* y la gramática *gramatica_tiendas.xml*, creada de forma dinámica mediante el fichero *crear_gramaticas.php* a partir de todas las tiendas almacenadas en la tabla 'tiendas' de la base de datos '*mariagj_info'*. Mediante el siguiente fichero, *nombre_tienda.php*, se accede a los datos de contacto de esta tienda.

Si el usuario no pregunta por una tienda en concreto, se le solicita que indique el tipo de tienda en que está interesado (por ejemplo, tiendas de ropa o calzado y complementos) y un distrito de Alcorcón. Seguidamente, se accede al fichero *listado_tiendas.php*, donde se reproduce el listado de las tiendas del tipo y la zona indicados por el usuario. Si de cualquiera de ellas se requiere más información, se proporcionará la dirección y el teléfono. Finalmente, se ofrece volver a escuchar el listado o se accede al fichero *ciudad_enlazar.php*.

- **Cines**

En el fichero *cines.php* se pregunta al usuario qué tipo de información desea: datos de contacto del cine, la cartelera actual o la cartelera para el día siguiente.

Si el usuario solicita los datos de contacto del cine, se accede a la tabla 'cines' de base de datos '*mariagj_info*' y se le proporciona la dirección, los teléfonos, los precios y los medios de transportes públicos para llegar.

Si, en cambio, el usuario quiere consultar la cartelera, se almacena en una variable el día (actual o día siguiente) en el que está interesado y se accede al fichero *listado_cartelera.php*, en el que se comprueba que haya películas para ese día. A continuación, se le informa del número de películas de la cartelera y se reproducen uno a uno los títulos. Cuando el usuario oiga el título de cualquiera de la películas en que esté interesado, sólo tendrá que decir sí o marcar 1 y entonces oirá la sala y los pases de esa película. Al final del listado de los títulos de las películas se pregunta si se quiere volver a oírlos y se repite el mismo procedimiento. En caso contrario, se accede al fichero *ciudad_enlazar.php*.

Los datos de las películas se cargan en la base de datos de forma dinámica, accediendo a la página web 'http://www.ecartelera.com/cines/63,0,1.html' y obteniendo de ella los títulos, salas y pases de las películas. Para almacenar esta información se utiliza la función *cargar_cartelera* del fichero *cargar_cartelera.php*, que es llamada desde el fichero *cargar_datos.php* (ejecutado por el administrador de la base de datos diariamente).

- **Hoteles y hostales**

Para el apartado de hoteles y hostales se procede de manera similar al apartado de tiendas. Debido a que el número de hoteles y hostales del municipio es muy limitado, el sistema accede a la base de datos y reproduce los nombres de todos los hoteles. En caso que el usuario desee más información de alguno de ellos, debe decir 'sí' o marcar '1' y entonces se le ofrecerá la dirección y teléfono de este hotel. Toda esta rutina se realiza en el fichero *hoteles.php*, siendo necesario el acceso a la tabla 'hoteles' de la base de datos '*mariagj_info*'.

Al final de cualquiera de estas 4 rutinas se accede al fichero *ciudad_enlazar.php*, en el que se ofrecen las siguientes opciones:

1- Otros datos de Alcorcón ciudad.

2- Volver al menú de información.

3- Volver al inicio del portal de voz.

4- Salir de la aplicación

En la Figura 4.13 se muestra el flujo de datos, la relación entre las rutinas y todos los ficheros que implementan este *submódulo Ciudad*.

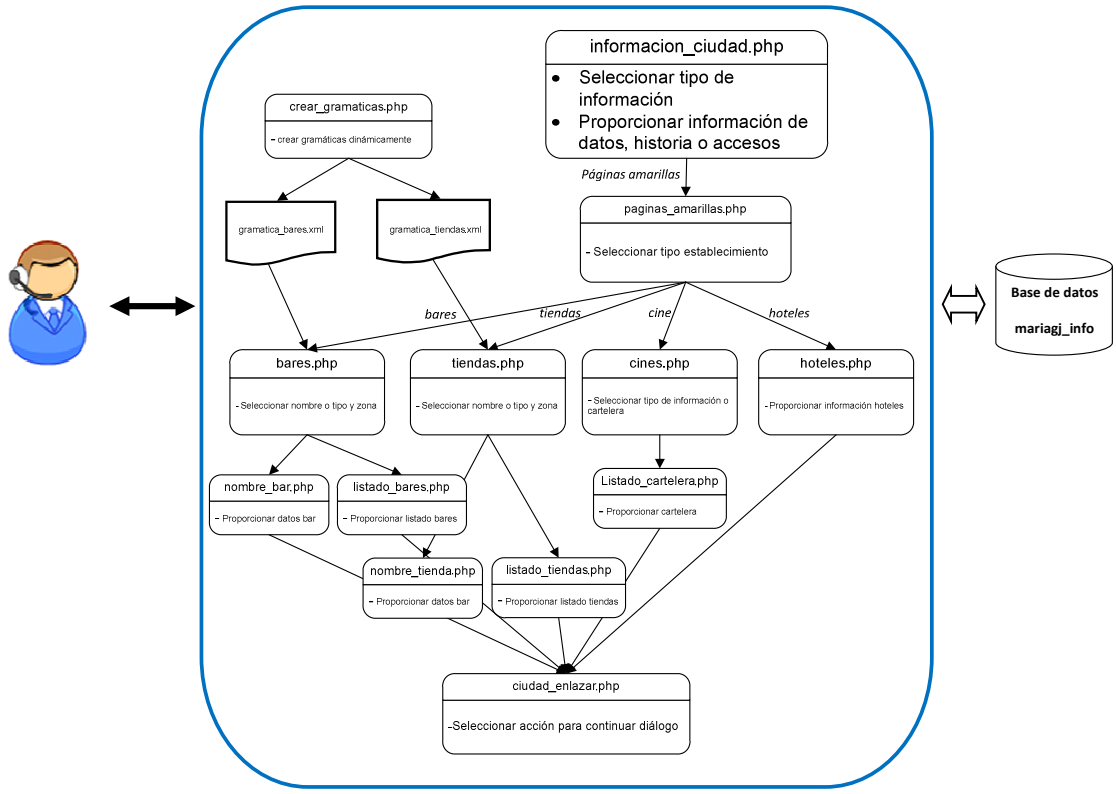


Figura 4.13. Flujo de datos del *submódulo Ciudad*

4.2.2.3 Escenarios de uso

Se muestran a continuación varios ejemplos de diálogo para el *submódulo Ciudad.*

☎ Consulta de los accesos a la ciudad (Figura 4.14).

S: Usted ha accedido al apartado de información sobre Alcorcón Ciudad.
Para consultar datos sobre la ciudad diga datos o marque 1.
Para consultar información sobre la historia de Alcorcón diga historia o marque 2.
Para consultar información sobre los accesos a la ciudad diga accesos o marque 3.
O, si desea consultar las páginas amarillas de Alcorcón diga páginas amarillas o marque 4.

U: Accesos.

S: Accesos a la ciudad:
Autovía de Extremadura A-5: Madrid - Badajoz - Portugal.
Autopistas de circunvalación M-40 y M-50.
Carretera M-501: Madrid - San Martín de Valdeiglesias - Plasencia.
Carretera M-406: Getafe - Leganés - Alcorcón.
Autopista de peaje R-5: Madrid - Navalcarnero.
Carretera M-506: Alcorcón - Villaviciosa de Odón.
Avenida de Móstoles: comunica Móstoles con Alcorcón, siendo el antiguo trazado de la N-V a su paso por estas dos localidades.

S: ¿A qué apartado desea ir ahora?
1- Alcorcón ciudad.
2- Menú información.
3- Inicio del portal de voz.
4- Salir de la aplicación.

U: Salir de la aplicación.

S: Usted ha seleccionado salir de la aplicación.
Gracias por utilizar nuestro portal de voz. Hasta pronto.

Figura 4.14. Escenario de uso para consultar los accesos a la ciudad

☎ Consulta de los datos de contacto de un bar del municipio: Gambrinus (Figura 4.15).

S: Usted ha accedido al apartado de información sobre Alcorcón Ciudad.
Para consultar datos sobre la ciudad diga datos o marque 1.
Para consultar información sobre la historia de Alcorcón diga historia o marque 2.
Para consultar información sobre los accesos a la ciudad diga accesos o marque 3.
O, si desea consultar las páginas amarillas de Alcorcón diga páginas amarillas o marque 4.

U: Páginas amarillas.

S: Para disfrutar de nuestra ciudad le proporcionamos un listado páginas amarillas con información sobre bares, cafés y restaurantes, tiendas., cines y hoteles y hostales en la ciudad.

S: Elija una de las secciones de las páginas amarillas para más información:
Para consultar información sobre bares y restaurantes diga bares o marque 1.
Para consultar información sobre tiendas diga tiendas o marque 2.
Para consultar información sobre el cine diga cine o marque 3.
O, para consultar información sobre hoteles diga hoteles o marque 4.

U: Bares.

S: ¿Conoce el nombre del establecimiento del que desea información?
Diga si o marque 1, o diga no o marque 2.

U: Sí

S: Por favor, diga el nombre del bar.

U: Gambrinus.

S: Usted ha dicho Gambrinus.

S: Los datos de contacto del establecimiento Gambrinus de tipo bares y cafés son: C/ Oslo, 53 CC Opción - Alcorcón. 916091857.

S: ¿Qué desea ahora?
1- Otros datos de Alcorcón ciudad.
2- Volver al menú de información.
3- Volver al inicio del portal de voz.
4- Salir de la aplicación.

U: Salir.

S: Usted ha seleccionado salir de la aplicación.
Gracias por utilizar nuestro portal de voz. Hasta pronto.

Figura 4.15. Escenario de uso de consulta de datos de contacto de un establecimiento

☎ Consulta de la cartelera del cine de Alcorcón para el día actual (Figura 4.16).

S: Usted ha accedido al apartado de información sobre Alcorcón Ciudad.
Para consultar datos sobre la ciudad diga datos o marque 1.
Para consultar información sobre la historia de Alcorcón diga historia o marque 2.
Para consultar información sobre los accesos a la ciudad diga accesos o marque 3.
O, si desea consultar las páginas amarillas de Alcorcón diga páginas amarillas o marque 4.

U: Páginas amarillas.

S: Para disfrutar de nuestra ciudad le proporcionamos un listado páginas amarillas con información sobre bares, cafés y restaurantes, tiendas., cines y hoteles y hostales en la ciudad.

S: Elija una de las secciones de las páginas amarillas para más información:
Para consultar información sobre bares y restaurantes diga bares o marque 1.
Para consultar información sobre tiendas diga tiendas o marque 2.
Para consultar información sobre el cine diga cine o marque 3.
O, para consultar información sobre hoteles diga hoteles o marque 4.

U: Cine.

S: ¿Qué tipo de información quiere?
Diga datos cine o marque 1,
diga cartelera hoy o marque 2, o ,
diga cartelera mañana o marque 3.

U: Cartelera hoy.

S: Hay 16 películas.
Se reproducirán los títulos de las películas.
Si quiere más información de alguna de ellas, diga sí o pulse 1.
Cisne negro
El discurso del Rey

U: Sí.

S: Sala 5: 17:45.

S: Siguiente película.
Encontrarás dragones
Esta abuela es mi padre
Furia Ciega
Gnomeo y Julieta
Invasión a la Tierra

U: Sí.

S: Sala 9: 17:50 | 20:10 | 22:30

S: Siguiente película
La legión del águila
Nada que declarar
…..
Sucker Punch
Torrente 4: Lethal crisis

S: Ya ha oído todos los títulos.
¿Desea volver a escuchar la lista de películas?
Diga sí o marque 1, o, diga no o marque 2.

U:	No.	
S:	S:	¿Qué desea ahora? 1- Otros datos de Alcorcón ciudad. 2- Volver al menú de información. 3- Volver al inicio del portal de voz. 4- Salir de la aplicación.
U:	Salir.	
S:	Usted ha seleccionado salir de la aplicación. Gracias por utilizar nuestro portal de voz. Hasta pronto.	

Figura 4.16. Escenario de uso de la consulta de la cartelera

4.2.3 Submódulo Áreas Temáticas

4.2.3.1 Funcionalidad

En el ***submódulo Áreas temáticas*** el usuario puede acceder a cualquier tipo de información sobre su municipio. Para la clasificación de toda esta información municipal se utilizan 15 áreas temáticas, que son las mostradas en la Figura 4.17.

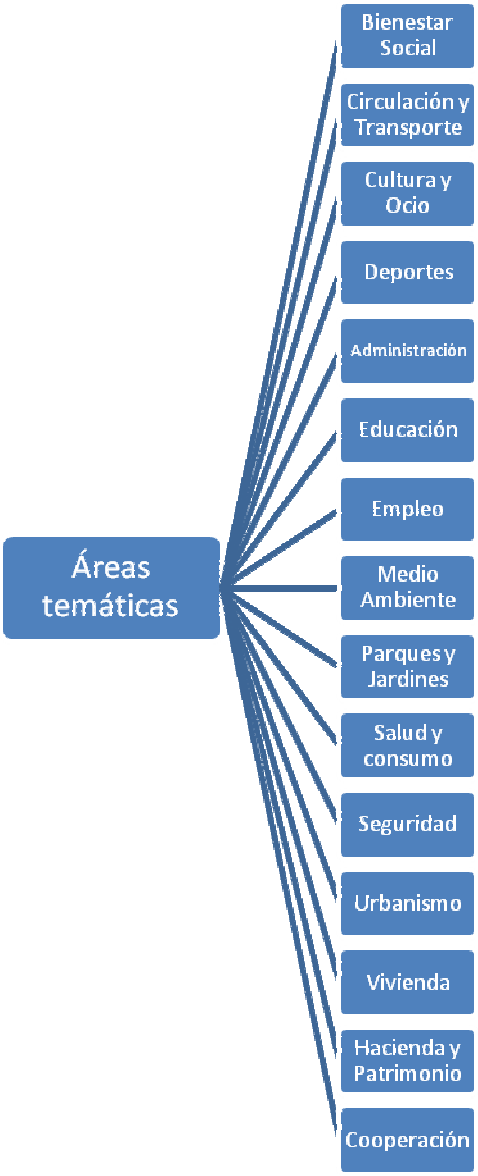


Figura 4.17. Esquema del *submódulo Áreas Temáticas*

4.2.3.2 Arquitectura

Al ser el número de áreas bastante elevado (15 áreas), se ha creído conveniente utilizar las características del parámetro '*bargein*', anteriormente explicado en el apartado 2.2. De esta forma, las áreas existentes y posibles a elegir son reproducidas al usuario, con la característica de que esta reproducción puede ser interrumpida en cualquier momento. Así, para un usuario que acceda por primera vez, le será de gran utilidad oír la clasificación y entonces elegir la opción donde se encuentre la información a la que quiera acceder. En cambio, para un usuario experimentado, nada más acceder a este apartado, no haría falta que esperase a oír las áreas, y podría decir su área elegida en cualquier momento, siendo inmediatamente redirigido al siguiente paso.

Además, para una mayor naturalidad del diálogo, se ha construido la gramática de tal forma que el conjunto de palabras que tiene que mencionarse para coincidir con una de las opciones ofrecidas no sea único. Es decir, se han incorporado combinaciones de las posibles palabras que el usuario podría decir para referirse a esa misma área.

Un ejemplo de esta gramática es el siguiente: si se quiere acceder al área de ‘Cultura y ocio’, se podrá decir tanto ‘cultura y ocio’ como ‘cultura’ o ‘ocio’, y accederemos a esta área. Esta funcionalidad facilita al usuario la navegación por los menús orales y los hace más naturales, además de evitar un mayor número de errores de reconocimiento.

Esta gramática devuelve un valor, que además de ser guardado como variable, es comprobado en un bloque *if/else* para dirigir el diálogo al siguiente fichero, *general.php*, en el que se le pregunta al usuario qué tipo de información quiere del área elegida. La información disponible para elegir es:

- Información general del área.

- Competencias del área.

- Datos de contacto del área.

La gramática usada en este fichero también está compuesta por varios ítems para el mismo valor devuelto, por tanto facilita al usuario la interacción a la hora de pedir el tipo de información que desea.

Una vez elegido el tipo de información, se accede a la tabla *areas_tematicas* de la base de datos *mariagj_info* para recopilar el tipo de información del área requerida. Un ejemplo para acceder a información de competencias del área es el mostrado en el código de la Figura 4.18.

```
<if cond="info == 'competencias'">
<prompt>
    Las competencias del área <value expr="area"/> son:
</prompt>

    <?php
    $cnx=conectar2();
    $query="SELECT competencias FROM areas_tematicas WHERE area='".$area."'";
    $result=mysql_query($query) or die (mysql_error());
    $i=0;
    $num=mysql_numrows($result);
    $competencias=mysql_result($result,$i,"competencias");
    echo $competencias;
    mysql_close($cnx);
    ?>
```

Figura 4.18. Código para acceder a información de un área temática

Una vez proporcionada la información al usuario, se accede al fichero *areas_tematicas_enlazar.php*, en el que se le pregunta qué desea hacer en ese punto. Las opciones proporcionadas son:

1- Volver a consultar otros datos de esta área. Ya que si el usuario ha consultado un tipo de información, seguramente este interesado en más información de la misma área.

2- Volver al menú de áreas temáticas. Si el usuario no ha encontrado en el área anteriormente consultada el tipo de información que buscaba, querrá intentarlo con otra área.

3- Volver al menú de información general.

4- Volver al inicio del portal de voz.

5- Salir de la aplicación. Se agradece al usuario haber utilizado el portal de voz municipal y se reproduce un mensaje de despedida.

El esquema del flujo de datos y la relación entre los archivos de este *submódulo Áreas Temáticas* los podemos observar en la Figura 4.19.

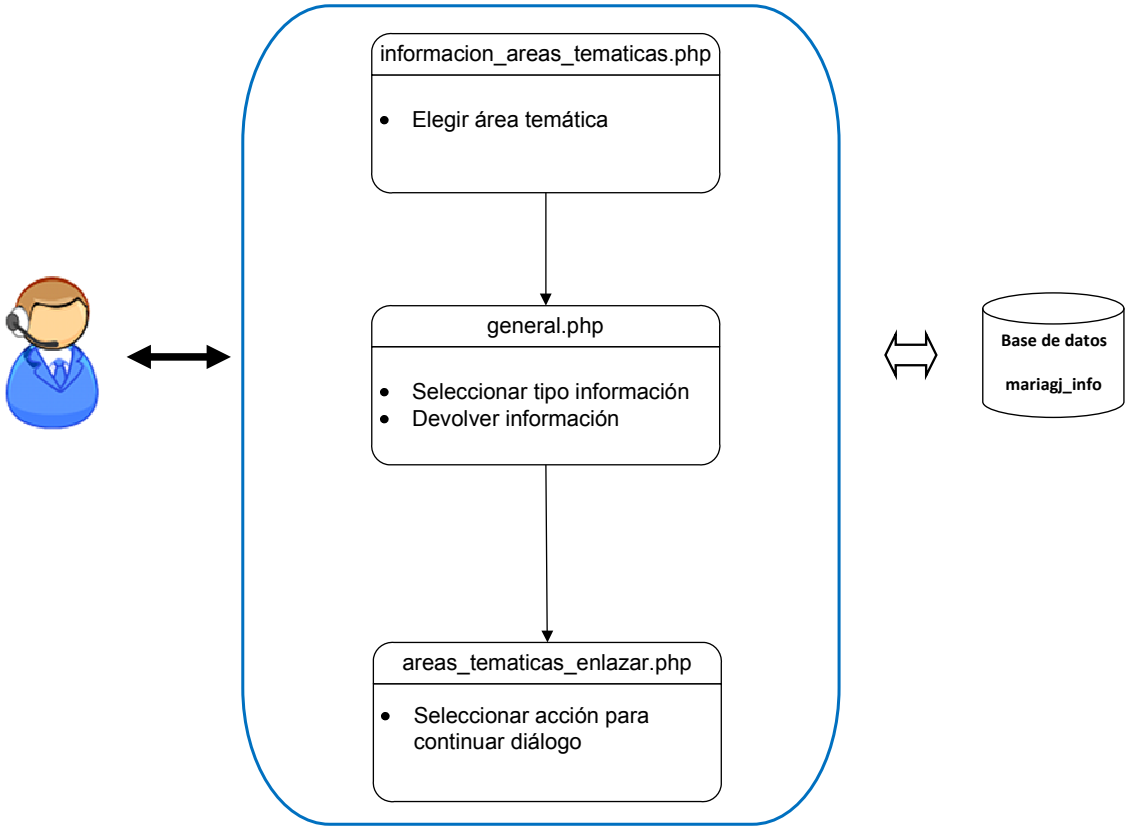


Figura 4.19. Flujo de datos del *submódulo Áreas Temáticas*

4.2.3.3 Escenarios de uso

En la Figura 4.20 se muestra un ejemplo de posible diálogo en este *submódulo Áreas Temáticas*.

S: ¿Sobre qué área temática quiere obtener información?
1-Bienestar Social,
2-Circulacion y Transporte,
3-Cultura y Ocio,
4-Deportes,
5-Administracion,
6-Educacion,
7-Empleo,
8-Medio Ambiente,
9-Parques y Jardines,
10-Salud y Consumo,
11-Seguridad,
12-Urbanismo,
13-Vivienda,
14-Hacienda y Patrimonio, y
15-Cooperación.

U: Bienestar Social.

S: Accedemos a consultar información sobre el área Bienestar Social.

S: ¿Qué tipo de información quiere del área Bienestar Social?
1-competencias del área.
2-información general.
3-datos de contacto.

U: Datos de contacto.

S: Los datos de contacto del área Bienestar Social son:

S: Centro de servicios sociales "VALDERAS".
Parque del Teide, 1. 28924 Alcorcón.
MetroSur: Puerta del Sur. Autobus: 514, 511.
Teléfonos: 916 106 611 - 916 106 797 - 916 426 024.
Fax: 916 425 311.
Centro de servicios sociales "EL TEJAR".
Plaza del Tejar, 4. 28921 Alcorcón.
MetroSur y Cercanías: Alcorcón Central.
Teléfonos: 916 648 503 - 916 648 504.
Fax: 916 425 311.

S: ¿Qué desea ahora?
1- Otros datos de esta área.
2- Menú de áreas temáticas.
2- Menú de información.
3- Inicio del portal de voz.
4- Salir de la aplicación.

U: Salir.

S: Usted ha seleccionado salir de la aplicación. Gracias por utilizar nuestro portal de voz. Hasta pronto.

Figura 4.20. Escenario de uso del *submódulo Áreas Temáticas*

4.2.4 Submódulo Noticias

4.2.4.1 Funcionalidad

En el ***submódulo Noticias***, se reproducen las noticias existentes en la página web del Ayuntamiento de Alcorcón. Puede obtenerse la fecha, el título y el subtítulo de cada noticia.

4.2.4.2 Arquitectura

Esta funcionalidad se implementa mediante dos acciones principales: insertar las noticias en la base de datos y reproducir estas noticias al usuario.

Para optimizar la velocidad de esta funcionalidad se ha decidido implementar estas dos acciones en rutinas separadas. Las noticias se deben cargar una vez al día, y así estarán disponibles para poder acceder a ellas en cualquier momento en el que un usuario solicite leerlas, sin necesidad de cargarlas cada vez.

- **Almacenar noticias**

Las noticias se insertan en la base datos mediante el fichero *cargar_infromacion.php*, en el que la función *cargar_noticias* recibe como parámetro el enlace a la página web de noticias del Ayuntamiento de Alcorcón (http://www.ayto-alcorcon.es/portal/rss?detalle=noticias'). El formato de las noticias es RSS y, por tanto, son fácilmente accesibles y están estructuradas en etiquetas. En este caso, para obtener el titular de cada noticia se debe buscar entre las etiquetas "*title*", y para obtener la fecha de publicación entre las etiquetas "*publish_date*".

Además, se necesita algo más de información de cada noticia. Interesa un subtítulo en el que se describa cada noticia, pero esto no lo proporciona directamente la información RSS. Por tanto, es necesario buscar las etiquetas "*link*" para obtener el enlace a otra página web en el que se describa cada noticia con más detalle. Entonces, se analiza esta otra página buscando la cadena existente entre "*subTit*" y "*span*" mediante la función *preg_match* de PHP, que busca este patrón en el código fuente de la página web. De este modo, se obtiene el subtítulo de cada noticia, tal y como se muestra en el código de la Figura 4.21.

```
$subtitulo='span class="subTit"';
$subtitulofin='span';
$m=preg_match('#<'.$subtitulo.'(?:\s+[^>]+)?>(.*?)'.'</'.$subtitulofin.'>#s',$xml2, $coincidencias);
$subtitulo=$coincidencias[1];
```

Figura 4.21. Código para obtener el subtítulo de cada noticia

Se deben tratar las cadenas de texto obtenidas para que estén en formato utf-8. Por tanto, debido a que la página web contiene entidades *html* correspondientes a caracteres especiales como signos de puntuación, letras con tilde o diéresis, o símbolos del lenguaje, se reemplaza cualquiera de estos caracteres por su

correspondiente carácter ASCII mediante la función *str_replace* de PHP. Un ejemplo para la expresión "&aacute" se presenta en la Figura 4.22.

```
$xml2=str_replace("&aacute;", "á", $xml2);
```

Figura 4.22. Código para reemplazar la expresión “&aacute”

El título, la fecha y el subtítulo de cada noticia son almacenados en la base de datos ‘*mariagj_info’* en la tabla ‘*noticias*’, tal y como se observa en la imagen de la Figura 4.23.

			id	fecha	titulo	subtitulo
☐	✎	✕	1	Martes, 29 de marzo de 2011	Puesta en marcha del portal ciudadano, una adminis...	A través de la página web del Ayuntamiento (www.ay...
☐	✎	✕	2	Martes, 29 de marzo de 2011	El alcalde denuncia el desprecio de Van Halen y Pé...	Recuerda que el director general de Vivienda estuv...
☐	✎	✕	3	Martes, 29 de marzo de 2011.	“La prioridad política debe ser la inversión en ed...	Alcorcón, ensalzado como referente en la implantac...
☐	✎	✕	4	Lunes, 28 de marzo de 2011	El Pleno Municipal aprueba una rebaja de las tasas...	El alcalde anuncia esta medida ante hosteleros de ...
☐	✎	✕	5	Martes, 29 de marzo de 2011	El Ayuntamiento y COGAM pone cara a las personas s...	Hasta el 14 de abril en el centro cívico Los Pinos
☐	✎	✕	6	Martes, 29 de marzo de 2011	El Teatro Buero Vallejo acoge la XXVIII Muestra Cu...	Los asistentes a este acto, celebrado en el Teatro...
☐	✎	✕	7	Viernes, 25 de marzo de 2011	El centro cívico Los Pinos acoge mañana la represe...	En el marco de las actividades programadas para co...
☐	✎	✕	8	Viernes, 25 de marzo de 2011	Alcorcón, referente europeo en Educación Infantil ...	Es la tasa más alta, solamente superada por Dinama...
☐	✎	✕	9	Jueves, 24 de marzo de 2011	Por tercer año consecutivo el Ayuntamiento de Alco...	Se apagarán las luces del Ayuntamiento, las ilumin...
☐	✎	✕	10	Jueves, 24 de marzo de 2011	La Comunidad de Madrid inaugura la escuela infanti...	De las diez escuelas infantiles que hay en la ciud...

Figura 4.23. Contenido de la tabla ‘noticias’

- **Leer noticias**

Una vez se tienen las noticias almacenadas en la base de datos, cuando el usuario pide escuchar las noticias, sólo hay que acceder a la base de datos y gestionar cómo reproducirlas.

Normalmente, la página de Alcorcón tiene alrededor de unas 10 noticias, pudiendo ser actuales del mismo día o de fechas pasadas. Por este motivo, el método que se ha pensado para facilitar estas noticias es el siguiente:

- En primer lugar, se comprueba que haya noticias, y se comunica al usuario el número total de noticias publicadas.

- A continuación, se reproduce una a una la fecha y el título de cada noticia, desde la más actual a la más antigua. En el caso en el que el usuario solicite más información de una de estas noticias, debe decir “sí” o pulsar 1, y entonces oirá el subtítulo de la noticia que ha elegido.

- Una vez terminado de reproducirse el subtítulo, se comunica la fecha y títulos de las noticias restantes, con la opción de volver a escuchar el subtítulo de

cualquiera en la que se esté interesado, y así sucesivamente hasta el final de la lista de noticias.

Terminadas de leer todas las noticias, se accede al fichero *enlazar_noticas.php* en el que se le pregunta al usuario qué desea hacer. Las opciones ofrecidas son:

1- Volver a escuchar las noticias.

2- Volver al apartado de información.

3- Volver al inicio del portal de voz.

4- Salir de la aplicación.

En la Figura 4.24 se representa la relación entre los ficheros y el flujo de datos de este *submódulo Noticias*.

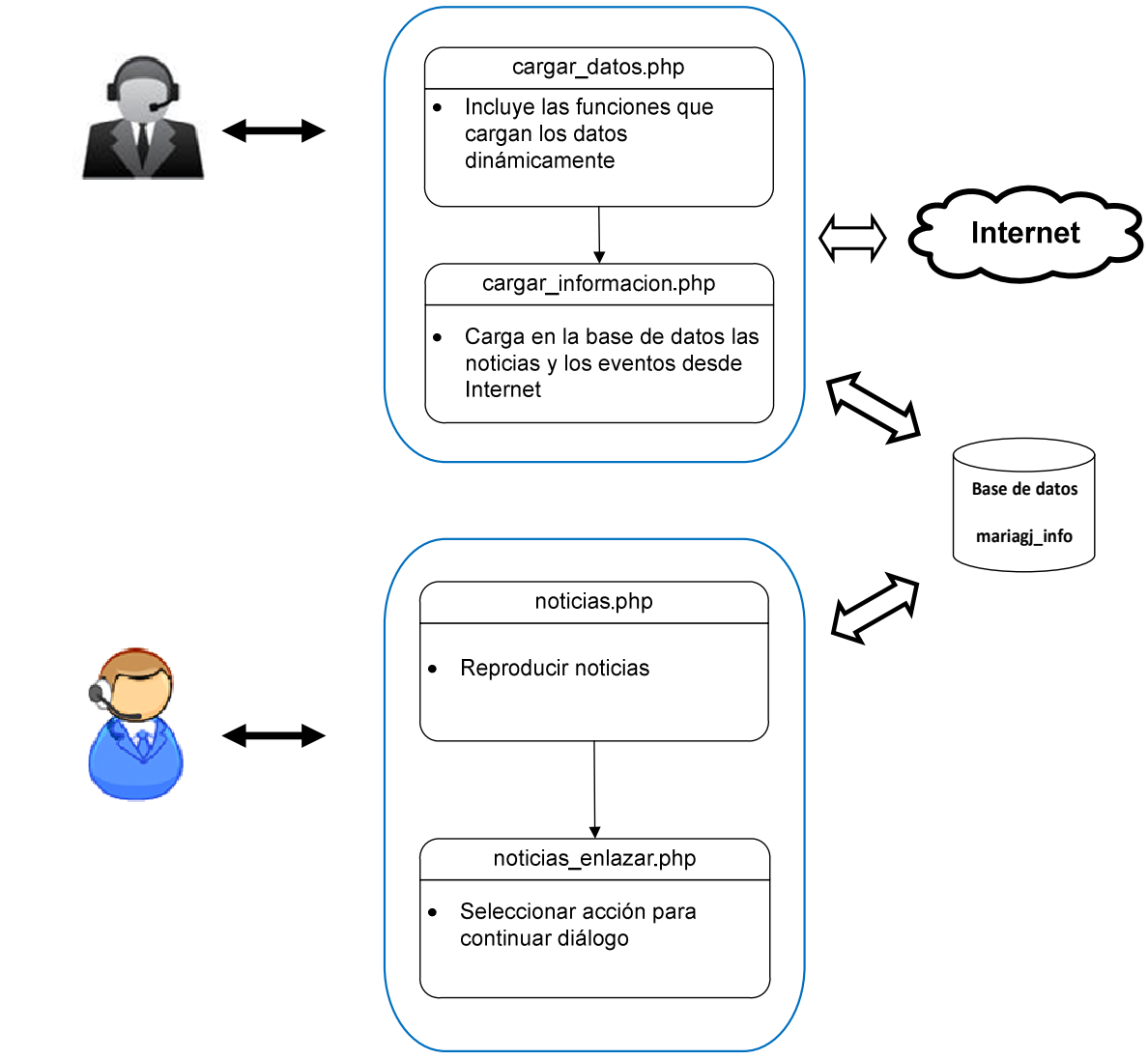


Figura 4.24. Flujo de datos del *submódulo Noticias*

4.2.4.3 Escenarios de uso

Un ejemplo de diálogo de este *submódulo Noticias* es el que se presenta en la Figura 4.25.

S: Hay 10 noticias.
Se reproducirán las fechas y los títulos.
Si quiere más información de alguna de ellas, diga sí o pulse 1.

S: Lunes, 18 de abril de 2011. La Junta de Gobierno aprueba nuevas licencias de primera ocupación y de edificación que generarán nuevos puestos de trabajo.
Lunes, 18 de abril de 2011. El Ayuntamiento recibe una subvención del Ministerio de Trabajo e Inmigración para el desarrollo de programas innovadores a favor de la integración de inmigrantes.

U: Sí.

S: Subtítulo de la noticia:
Las actividades se enmarcan en el proyecto "Tú eres 1 +", para fomentar la participación y el asociacionismo.

S: Siguiente noticia.
Viernes, 15 de abril de 2011. Desde 2006 se han construido más de 150.000 metros cuadrados de edificabilidad en el polígono Ventorro el Cano, que han hecho posible la creación de 4.000 puestos de trabajo.

…….

Miércoles, 13 de abril de 2011. El alcalde firma un acuerdo con las centrales sindicales de ESMASA que recoge las reivindicaciones de los trabajadores a tiempo parcial.
Ya ha oído todas las noticias.

S: ¿A qué menú desea ir ahora?
1- Noticias.
2- Información.
3- Inicio del portal de voz.
4- Salir de la aplicación.

Figura 4.25. Escenario de uso del *submódulo Noticias*

4.2.5 Submódulo Eventos

4.2.5.1 Funcionalidad

En el ***submódulo Eventos*** se facilitan los eventos publicados en la página web del Ayuntamiento de Alcorcón. Se proporciona el área temática, título, fecha, lugar y descripción de cada uno de ellos.

4.2.5.2 Arquitectura

El proceso mediante el que se implementa esta funcionalidad es similar al proceso utilizado para el *submódulo Noticias*, llevándose a cabo mediante dos acciones principales: cargar los eventos en la base de datos y reproducir estos eventos al usuario.

Al igual que en el caso de las noticias, para optimizar el funcionamiento se han implementado estas dos acciones en rutinas separadas. Los eventos se almacenan en la base de datos diariamente, y así éstos están disponibles en cualquier momento en el que el usuario los solicite, sin necesidad de cargarlos en cada solicitud.

- **Almacenar eventos**

Los eventos se almacenan en la base datos mediante el fichero *cargar_informacion.php*, en el que a la función *cargar_eventos* se le facilita como parámetro el enlace a la página web de eventos del Ayuntamiento de Alcorcón (http://www.ayto-alcorcon.es/portal/agendaeventos/index). En este caso, el formato no es RSS, con lo que se obtienen los datos que se necesitan de cada evento accediendo a la página web de eventos, realizando un análisis del código fuente y almacenando el contenido de las etiquetas que interesan.

Para obtener el título de cada evento, se busca entre las etiquetas *'h2 class="tit"'* y '*h2*', para obtener el área temática de cada evento se exploran las etiquetas que empiecen por *'h3 class="tit"'* y terminen en '*h3*'. Para obtener la descripción de la noticia se examina el patrón que empiece por *'div class="modContent"'* y termine en '*div*', y para las fechas, el patrón de etiquetas buscado es *'ul class="listType01"'* terminado en '*ul*'. Mediante la función *preg_match_all* de PHP se buscan estos patrones en el código fuente de la página web.

Un ejemplo de cómo se obtiene esta información se muestra en la Figura 4.26, en la que se puede observar el código para obtener los títulos de los eventos.

```
$patron_titulo='h2 class="tit"';
$patron_titulofin='h2';
$titulos_encontrado=preg_match_all('#<'.$patron_titulo.'(?:\s+[^>]+)?>'.'(.*?)</'.$patron_titulofin.'>
#s', $xml2, $titulos);
```

Figura 4.26. Código para obtener los títulos de los eventos

Una vez obtenido cada uno de estos datos buscados, se eliminan las etiquetas HTML con la función *strip_tags* de PHP y se adecúa la cadena resultante a formato ASCII, de tal forma que se eliminen los retornos de carro, las nuevas líneas, las tabulaciones, y los caracteres no permitidos, tal y como se observa en el código de la Figura 4.27.

```
$caracteres_no_permitidos = array("\"","'");
//se eliminan todos los retorno de carro
$s = str_replace("\r", '', $s);

//se elimina la inserción de nuevas líneas
$s = str_replace("\n", '', $s);

//se eliminan tabulaciones y caracteres no permitidos
$s = str_replace("\t", '', $s);
$s = str_replace($caracteres_no_permitidos,"",$s);
```

Figura 4.27. Código para adecuar la cadena de texto con la información obtenida de la web

Toda la información utilizada en la aplicación se codifica en utf-8. Por tanto, en las cadenas de texto obtenidas se reemplaza cualquiera de las expresiones HTML mencionadas anteriormente por su correspondiente carácter ASCII.

Finalmente, los datos de cada evento (título, área temática, descripción y fecha) son almacenados en la tabla '*eventos*' de la base de datos '*mariagj_info'* como se aprecia en la Figura 4.28.

	id	titulo	area	descripcion	fecha
	1	NUEVA CONVOCATORIA PLAN PERMUTA 2011 - Abierto Pla...	Vivienda	El conocido Plan Permuta abre nueva convocatoria. ...	Desde 24/03/2011 - Hasta 06/05/2011Lugar: Alcorcón
	2	Seminario y Viaje Cultural: Rumania, punto de encu...	Cultura y Ocio	El seminario se realizará durante el mes de mayo ...	Desde 15/04/2011 - Hasta 01/06/2011Lugar: Centro C...
	3	EXPOSICIÓN: Urbscapes, espacios de hibridación	Cultura y Ocio	Producida por la Universidad Politécnica de Valenc...	Desde 01/03/2011 - Hasta 31/03/2011Lugar: Centro M...
	4	EXPOSICIÓN ANA GALÁN: Subjeeeetivos: Ahora retrato...	Cultura y Ocio	Subjeeeetivos: Ahora retratos, es una exposición c...	Desde 01/03/2011 - Hasta 31/03/2011Lugar: Centro M...
	5	Vidrioh! - Indice 010 - MAVA	MAVA	Índice 010 es la exposición multimedia de contenid...	Desde 25/02/2011 - Hasta 20/04/2011Lugar: MAVA - M...
	6	PROCESO DE ESCOLARIZACIÓN PARA EL CURSO 2011/2012	Educación	Plazo de presentación de solicitudes del 21 de mar...	Desde 21/03/2011 - Hasta 11/04/2011Lugar: Alcorcón

Figura 4.28. Tabla 'eventos'.

- **Leer eventos**

Una vez se tienen los eventos almacenadas en la base de datos, sólo hay que acceder a la base de datos y gestionar su reproducción por voz.

La página web de Alcorcón suele tener entre 5 y 10 eventos. El método que se ha creído más adecuado para facilitar la información de los eventos es el siguiente:

- En primer lugar, se comprueba que hay eventos almacenados en la base de datos, esto indica que hay programados eventos en el municipio. Si no hay eventos, se le informa al usuario, y se pasa al siguiente menú. En caso que sí haya eventos, se informa al usuario del número total de eventos programados.

- A continuación, se reproduce el área y el título de cada evento. En el caso en el que el usuario quiera más información de uno de los eventos, deberá decir "sí" o pulsar 1, y entonces escuchará la fecha, el lugar y la descripción del evento elegido. Una vez terminada de reproducirse esta información extra, se seguirán leyendo las áreas y los títulos de los eventos restantes, con la posibilidad de

volver a escuchar más información de cualquiera de ellos, y así hasta el final de la lista de eventos.

Al finalizar la reproducción de todos los eventos, se le comunica este hecho al usuario y se accede al fichero *enlazar_eventos.php*, en el que se le pregunta qué desea hacer en ese momento. Las opciones ofrecidas son:

1- Volver a escuchar los eventos.

2- Volver al apartado de información.

3- Volver al inicio del portal de voz.

4- Salir de la aplicación.

En la Figura 4.29 se presenta el flujo de datos del *submódulo Eventos* junto con la relación entre los distintos ficheros que implementan esta funcionalidad.

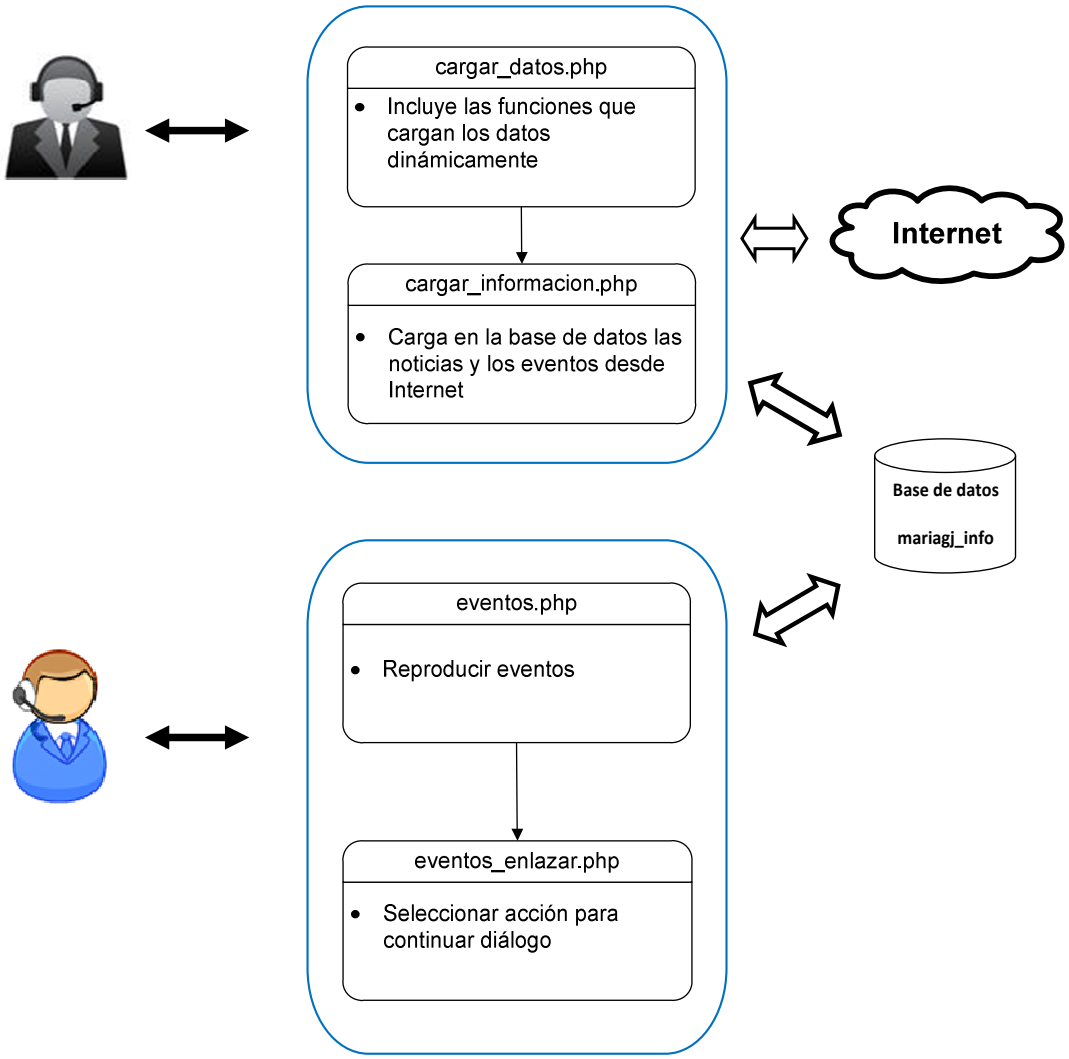


Figura 4.29. Flujo de datos del *submódulo Eventos*

4.2.5.3 Escenarios de uso

En la Figura 4.30 se ejemplifica un posible diálogo para el *submódulo Eventos*.

S: Hay 4 eventos.
Se reproducirán las áreas y los títulos de cada evento.
Si quiere fecha, lugar e información más detallada de alguno de ellos, diga sí o pulse 1.

S: Cultura y Ocio. Seminario UPA: Historia de las Religiones - 7 de Abril a 26 de Mayo.
Vivienda. NUEVA CONVOCATORIA PLAN PERMUTA 2011 - Abierto Plazo de Presentación de Solicitudes.
Cultura y Ocio. Seminario y Viaje Cultural: Rumania, punto de encuentro de Tres Culturas.

U: Sí.

S: Descripción del evento.
Desde 15/04/2011 - Hasta 01/06/2011. Lugar: Centro Cívico Margarita Burón.
El seminario se realizará durante el mes de mayo de 2011, desarrollando la historia, el arte, la literatura, los monumentos a visitar y la parte social de país, para ello contaremos con distintos especialistas, la Federación de Asociaciones de Rumanos en España (FEDROM) y la Embajada de Rumanía.

Siguiente evento.
MAVA. Vidrioh! - Índice 010 – MAVA

U: Sí.

S: Descripción del evento:
Desde 25/02/2011 - Hasta 20/04/2011. Lugar: MAVA - Museo de Arte en Vidrio de Alcorcón.
Índice 010 es la exposición multimedia de contenidos y piezas a los que vidrioh! Ha dado cobertura online durante 2010.La muestra, organizada en colaboración con el MAVA, incluye una proyección de vídeos, una exposición de piezas y una galería visual de contenidos difundidos en Vidrioh.

S: Ya ha oído todos los eventos.

S: ¿A qué menú desea ir ahora?
1- Noticias.
2- Información.
3- Inicio del portal de voz.
4- Salir de la aplicación.

Figura 4.30. Escenario de uso del *submódulo Eventos*

4.2.6 Submódulo Información Meteorológica

4.2.6.1 Funcionalidad

El ***submódulo Información Meteorológica*** permite al usuario consultar la situación meteorológica actual y la previsión para los próximos dos días del municipio de Alcorcón. Las condiciones meteorológicas que se proporcionan son las que se muestran en la Tabla 4.1.

Instante actual	+1día	+2días
Estado: * Mayormente soleado * Despejado * Nublado * Posibilidad de lluvia	Estado: * Mayormente soleado * Despejado * Nublado * Posibilidad de lluvia	Estado: * Mayormente soleado * Despejado * Nublado * Posibilidad de lluvia
Temperatura actual (ºC)	Temperatura mínima (ºC)	Temperatura mínima (ºC)
Humedad (%)	Temperatura máxima (ºC)	Temperatura máxima (ºC)

Tabla 4.1. Información que proporciona el *submódulo Información Meteorológica*

4.2.6.2 Arquitectura

Se necesitan dos acciones para implementar este submódulo: obtener la información de la web y facilitar esta información al usuario.

- **Obtener los datos de Google**

La información meteorológica que se necesita se obtiene del servicio correspondiente proporcionado por Google (http://www.google.com/ig/api?weather=city). Para ello, se accede a la URL correspondiente en iGoogle, que facilita un código XML con los datos del tiempo actual (condición, temperatura, humedad, vientos) para la ciudad seleccionada y el pronóstico para los siguientes tres días.

Estos datos se proporcionan por defecto en idioma inglés, pero agregando un parámetro adicional '*hl*' para indicar el idioma de la respuesta, se pueden obtener en español: (http://www.google.com/ig/api?weather=city&hl=es). La Figura 4.31 muestra un ejemplo de código XML obtenido de la consulta a esta página.

```
<xml_api_reply version="1">
<weather module_id="0" tab_id="0" mobile_row="0" mobile_zipped="1" row="0"
section="0">

<forecast_information>
<city data="Alcorcón, Madrid"/>
<postal_code data="alcorcon"/>
<latitude_e6 data=""/>
<longitude_e6 data=""/>
<forecast_date data="2011-04-16"/>
<current_date_time data="2011-04-16 17:00:00 +0000"/>
<unit_system data="SI"/>
</forecast_information>

<current_conditions>
<condition data="Parcialmente nublado"/>
<temp_f data="66"/>
<temp_c data="19"/>
<humidity data="Humedad: 46%"/>
<icon data="/ig/images/weather/partly_cloudy.gif"/>
<wind_condition data="Viento: E a 11 km/h"/>
</current_conditions>

<forecast_conditions>
<day_of_week data="sáb"/>
<low data="6"/>
<high data="22"/>
<icon data="/ig/images/weather/mostly_sunny.gif"/>
<condition data="Mayormente soleado"/>
</forecast_conditions>

<forecast_conditions>
<day_of_week data="dom"/>
<low data="7"/>
<high data="23"/>
<icon data="/ig/images/weather/sunny.gif"/>
<condition data="Despejado"/>
</forecast_conditions>

<forecast_conditions>
<day_of_week data="lun"/>
<low data="6"/>
<high data="22"/>
<icon data="/ig/images/weather/mostly_sunny.gif"/>
<condition data="Mayormente soleado"/>
</forecast_conditions>

<forecast_conditions>
<day_of_week data="mar"/>
<low data="7"/>
<high data="23"/>
<icon data="/ig/images/weather/sunny.gif"/>
<condition data="Despejado"/>
</forecast_conditions>
</weather>
</xml_api_reply>
```

Figura 4.31. XML con la información meteorológica

Mediante la función *file_get_contents* de PHP se transmite este archivo XML entero en forma de cadena de texto, que posteriormente se convierte en un array utilizando la función *xml2array*.

Seguidamente, se almacenan los datos de los nodos *forecast_information*, *current_conditions* y *forecast_conditions* en tres variables. Estos nodos contienen los datos de la ciudad consultada, los datos actuales del tiempo y la predicción meteorológica para los siguientes tres días. Este procedimiento se muestra en el código de la Figura 4.32.

```
// hl=es para idioma español
$url = "http://www.google.com/ig/api?weather=alcorcon&hl=es";

// codificación utf-8
$contents = utf8_encode(file_get_contents($url));
$data = xml2array($contents);

$tiempo_info = $data['xml_api_reply']['weather']['forecast_information'];
$tiempo_actual = $data['xml_api_reply']['weather']['current_conditions'];
$tiempo_prevision = $data['xml_api_reply']['weather']['forecast_conditions'];

$estado= $tiempo_actual['condition_attr']['data'];
$temperatura= $tiempo_actual['temp_c_attr']['data'];
$humedad= $tiempo_actual['humidity_attr']['data'];

$dia_x1 = $tiempo_prevision[0];
$dia_sem_x1=$dia_x1['day_of_week_attr']['data'];
$dia_sem_x1=dia($dia_sem_x1);
$temp_min_x1= $dia_x1['low_attr']['data'];
$temp_max_x1= $dia_x1['high_attr']['data'];
$estado_prevision_x1= $dia_x1['condition_attr']['data'];

$dia_x2 = $tiempo_prevision[1];
$dia_sem_x2=$dia_x2['day_of_week_attr']['data'];
$dia_sem_x2=dia($dia_sem_x2);
$temp_min_x2= $dia_x2['low_attr']['data'];
$temp_max_x2= $dia_x2['high_attr']['data'];
$estado_prevision_x2= $dia_x2['condition_attr']['data'];
```

Figura 4.32. Código utilizado para obtener la información meteorológica

- **Proporcionar la información meteorológica**

El siguiente paso es proporcionar los datos obtenidos y que están almacenados en las respectivas variables. Se reproducirán los datos del tiempo actual y las predicciones para los siguientes dos días.

La variable correspondiente al día de la semana está publicada en la web sólo con tres caracteres. Interesa adaptarla antes de proporcionársela al usuario y por tanto, se ha implementado la función *dia*, que hace una correspondencia entre los días de la semana facilitados en la web con tres caracteres y el nombre completo del día de la semana. Esta transformación se contempla en el código de la Figura 4.33.

```
function dia($dia_semana){
if($dia_semana=="lun") $dia_semana="lunes";
if($dia_semana=="mar") $dia_semana="martes";
if($dia_semana=="mié") $dia_semana="miércoles";
if($dia_semana=="jue") $dia_semana="jueves";
if($dia_semana=="vie") $dia_semana="viernes";
if($dia_semana=="sáb") $dia_semana="sábado";
if($dia_semana=="dom") $dia_semana="domingo";
return $dia_semana;
}
```

Figura 4.33. Código para adaptar el día de la semana

Finalmente, se reproduce la información al usuario de la forma más natural posible, intercalando para ello texto fijo con las variables PHP donde se almacenan los datos, tal y como se muestra en el código de la Figura 4.34.

```
<form>
  <block>
  <prompt>
       Usted ha elegido consultar la previsión meteorológica.
       En estos momentos en Alcorcón el estado es <?=$estado?>, con una temperatura de
       <?=$temperatura?> grados, y <?=$humedad?>.
       Mañana <?=$dia_sem_x1?>, se espera que el estado sea <?=$estado_prevision_x1?>, con una
       temperatura mínima de <?=$temp_min_x1?> grados, y una temperatura máxima de
       <?=$temp_max_x1?> grados.
       Pasado mañana, <?=$dia_sem_x2?>, se espera que el estado sea <?=$estado_prevision_x2?>,
       con una temperatura mínima de <?=$temp_min_x2?> grados, y una temperatura máxima de
       <?=$temp_max_x2?> grados.
  </prompt>
  <goto
next="http://proyectovxml.x10.mx/informacion/informacion_meteorologica/informacion_meteorol
ogica_enlazar.php"/>
  </block>
</form>
```

Figura 4.34. Código que reproduce al usuario la información meteorológica.

Una vez proporcionada la información al usuario, se accede al fichero *tiempo_enlazar.php* donde se le pregunta qué acción desea realizar a continuación. Las opciones disponibles en este caso son:

1- Volver al menú información.

2- Volver al menú inicio.

3- Salir de la aplicación.

El diagrama de flujo de la información y la relación entre los ficheros que componen este submódulo se observan en la Figura 4.35.

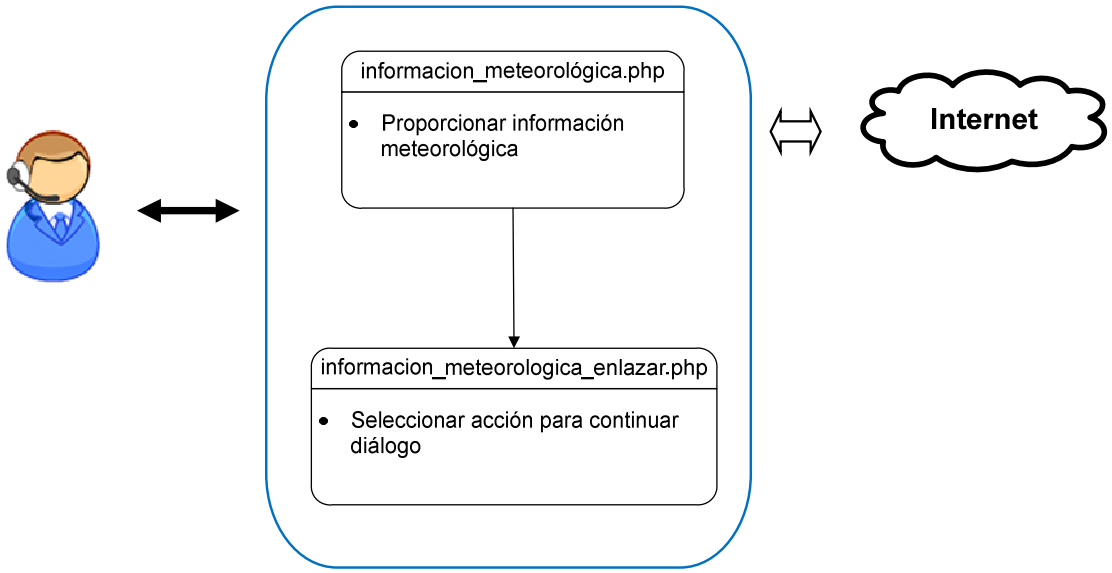


Figura 4.35. *Submódulo Información Meteorológica*

4.2.6.3 Escenarios de uso

En la Figura 4.36 se presenta un ejemplo de diálogo del *submódulo Información Meteorológica*.

S: ¿Qué tipo de información quiere consultar?
Diga Ayuntamiento o marque 1,
diga ciudad o marque 2,
diga áreas temáticas o marque 3,
diga noticias o marque 4,
diga eventos o marque 5,
o, diga información meteorológica o marque 6.

U: Información meteorológica.

S: Usted ha seleccionado consultar la información meteorológica del municipio de Alcorcón.

S: En estos momentos en Alcorcón el estado es despejado, con una temperatura de 19 grados, y humedad 40 %.
Mañana domingo, se espera que el estado sea parcialmente nublado, con una temperatura mínima de 9 grados, y una temperatura máxima de 21 grados.
Pasado mañana, lunes, se espera que el estado sea mayormente soleado con una temperatura mínima de 6 grados, y una temperatura máxima de 22 grados.

Figura 4.36. Escenario de uso del *submódulo Información Meteorológica*

4.3 Módulo Gestiones y Trámites

4.3.1 Funcionalidad

Mediante el ***módulo Gestiones y Trámites*** el usuario puede verificar si está incluido en un listado, comprobar el estado de sus expedientes, reservar una instalación municipal o pedir cita para ser atendido en un servicio municipal. La distribución de estos servicios se observa en la Figura 4.37.

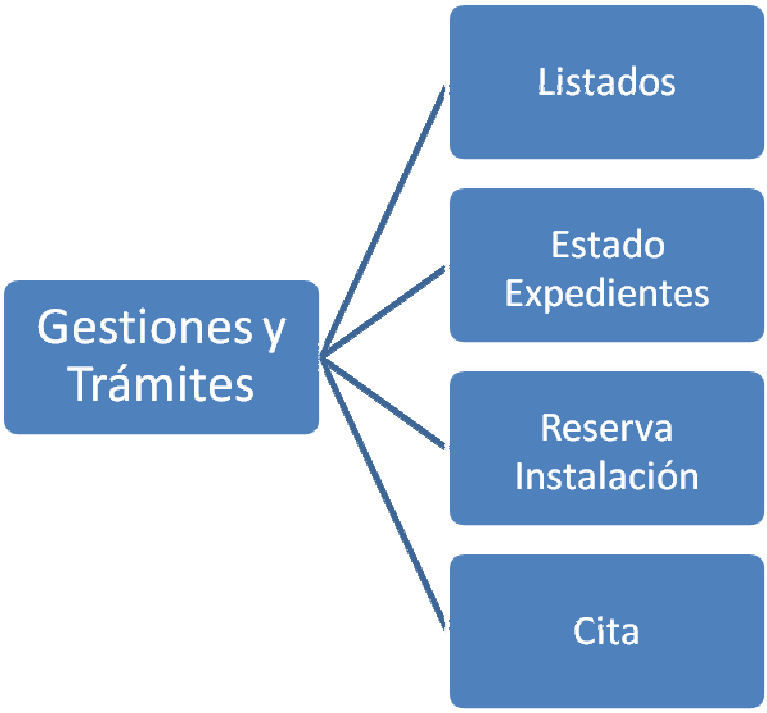


Figura 4.37. Servicios del *módulo Gestiones y Trámites*

4.3.2 Arquitectura

En el fichero *gestiones.php* se pregunta al usuario qué gestión o trámite desea realizar de entre los enumerados anteriormente: comprobar si está en un listado, comprobar el estado de sus expedientes, reservar una instalación o pedir cita.

Se guarda en una variable la respuesta, y se accede al fichero *gestiones_identificador.php*, donde se le solicita que se identifique mediante los 8 números de su DNI. Para ello debe pronunciar los 8 números de su DNI, uno a uno, o facilitarlos mediante DTMF pulsando las teclas del teléfono.

Una vez que se ha almacenado el número identificador del usuario, se accede a un bloque de condiciones (*if/else*) en el que, en función de la gestión a realizar que el usuario indicó en el fichero anterior, se accede a uno de los submódulos de la aplicación u otro.

A partir de este punto, las cuatro opciones del *módulo Gestiones y Trámites* se desarrollan en rutinas diferentes que se explican a continuación:

- **Listados**

En el fichero *gestiones_listado.php* se recoge el número identificador. Se accede entonces a la base de datos '*mariagj_proyecto*', y se comprueba si el número identificador está en la tabla '*DNI*', la cual se muestra en la Figura 4.38.

DNI
47451000
47452060
47452061
50678472
53467198

Figura 4.38. Tabla '*DNI*'

Si está este número de DNI en la base de datos, se le comunica al usuario que aparece en el listado. Si no se encuentra el número identificador en la tabla 'DNI', se le comunica al usuario que no está en el listado.

El esquema del flujo de datos y la relación entre los ficheros de la rutina para comprobar listados se muestran en la Figura 4.39.

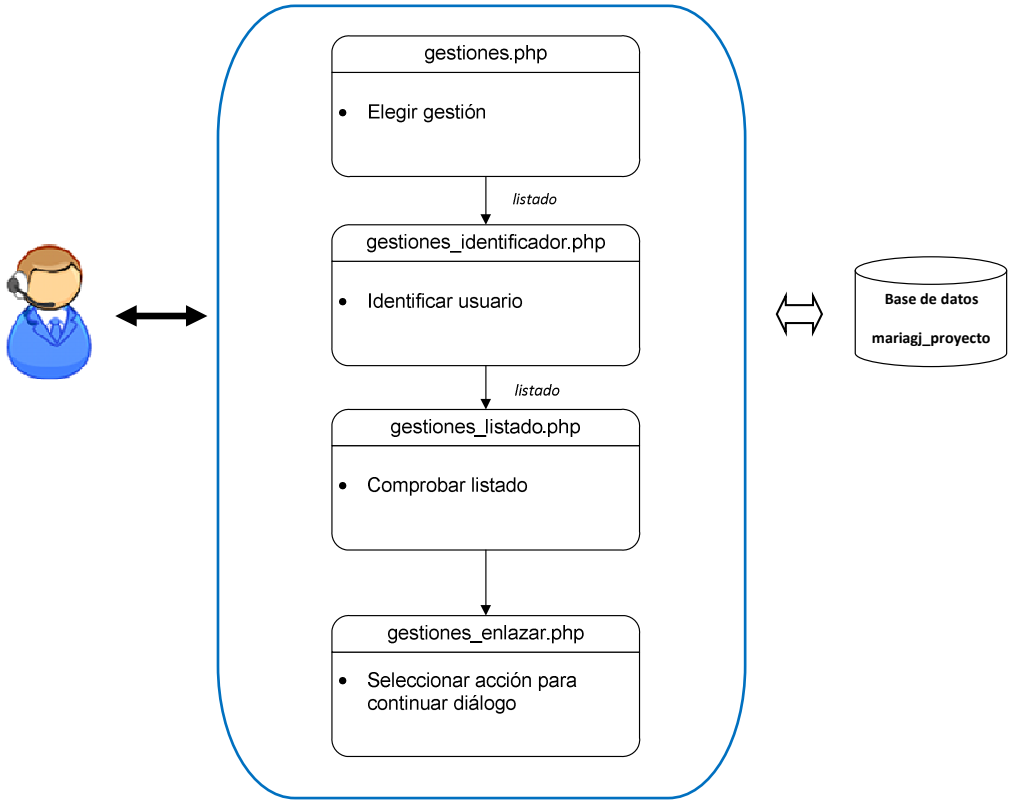


Figura 4.39. Flujo de datos de la rutina comprobación de listados

- **Estado de expedientes**

En el fichero *gestiones_tramites.php* se recoge la variable que contiene el identificador del usuario. Se accede a la base de datos '*mariagj_proyecto*' y se busca en la tabla '*tramites*' (Figura 4.40) el trámite para ese DNI.

	DNI	tramites
	47452060	Su cita para la declaración de la renta es el dia ...
	47452088	Su certificado de empadronamiento está solicitado.
	70788117	Multa de tráfico pendiente de pago

Figura 4.40. Tabla '*tramites*'.

Si no se encuentra el número DNI en la tabla, el usuario no tiene trámites pendientes. En cambio, si existe el número DNI, se le proporciona al usuario el estado de sus expedientes (celda correspondiente a su DNI en la columna trámites).

La relación entre los ficheros y el flujo de datos para esta rutina se presentan en la Figura 4.41.

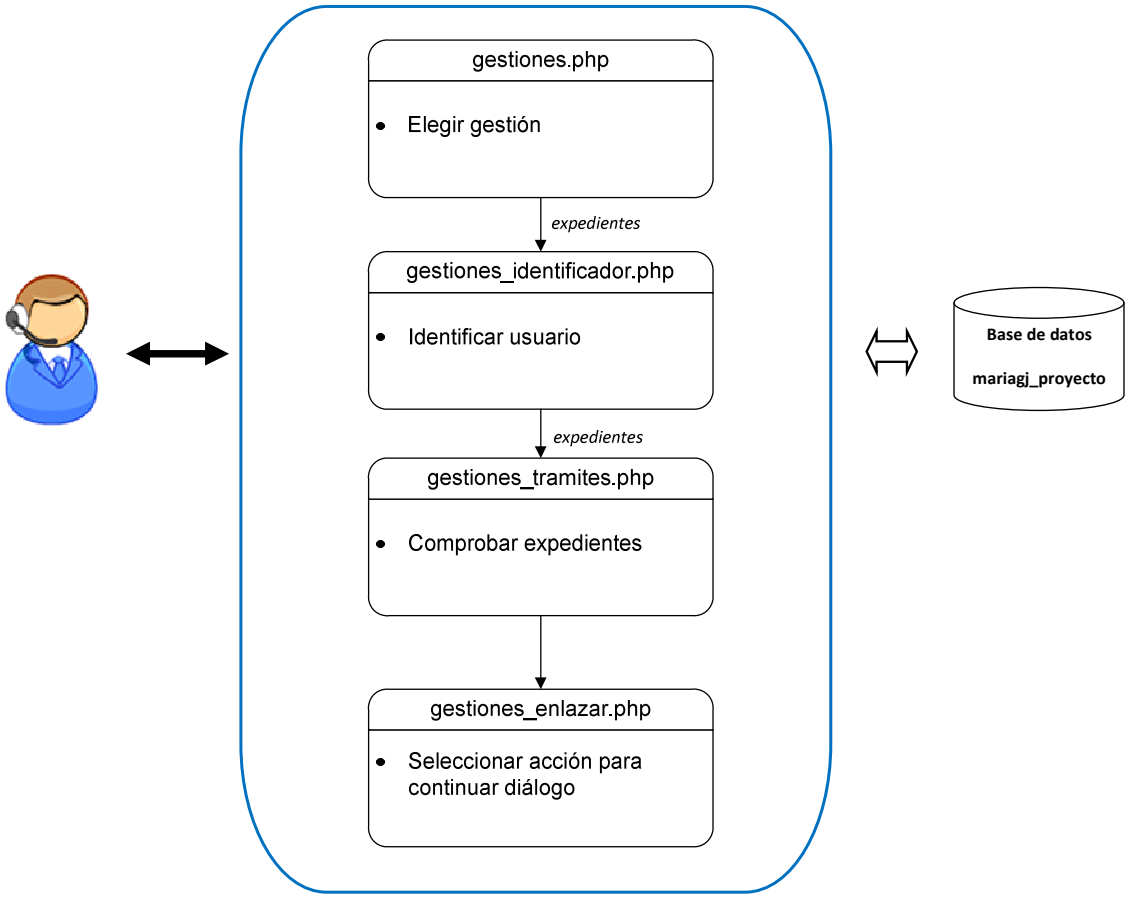


Figura 4.41. Flujo de datos de la rutina comprobación del estado de expedientes

- **Reserva de instalación**

En el fichero *gestiones_reserva_dia.php* se le requiere al usuario la fecha en la que quiere realizar la reserva. Puede introducir la fecha mediante voz (año, mes y día) o mediante DTMF, de la forma: aaaammdd.

Seguidamente, se almacena la fecha en la variable '*fecha*' y se continúa la ejecución en el fichero *comprobación.php*, en el que se obtiene mediante *$_REQUEST* dicha fecha. En este fichero se comprueba que la fecha es válida.

El formato de la fecha recibido es "aaaammdd". De este modo, lo primero que es necesario realizar es separar el año, mes y día mediante la función *substr* de PHP, para utilizar a continuación la función *php ckeckdate* según se muestra en el código de la Figura 4.42.

```
$anio=substr($fecha, 0, 4);
$mes=substr($fecha, 4, 2);
$dia=substr($fecha, 6, 2);

$fecha_valida=checkdate($mes, $dia, $anio);
```

Figura 4.42. Código para comprobar que una fecha es válida

Si la fecha no es válida, se le indica al usuario y se le pide que ingrese otra fecha, volviendo la ejecución al fichero en el que se pregunta al usuario por una fecha. Si la fecha es válida, se accede al fichero *gestiones_reserva_horas.php*. En este fichero se le da formato a la fecha proporcionada por el usuario y ya comprobada como válida. El formato "aaaammdd" se transforma en "aaaa-mm-dd" para poder incluirlo y procesarlo en la base de datos mediante sentencias SQL, según se observa en el código de la Figura 4.43.

```
$anio=substr($fecha, 0, 4);
$mes=substr($fecha, 4, 2);
$dia=substr($fecha, 6, 2);

$fecha_mysql= $anio."-".$mes."-".$dia;
```

Figura 4.43. Código para transformar el formato de fecha

Seguidamente, se accede a la tabla '*reserva*' (Figura 4.44) de la base de datos y se comprueba para esta fecha qué horas hay libres.

←T→	id	dia	hora	estado	DNI
	1	2011-05-11	09:00:00	Ocupado	47452060
	2	2011-05-11	10:00:00	Ocupado	47452060
	3	2011-05-11	11:00:00	Libre	0
	4	2011-05-11	12:00:00	Libre	0
	5	2011-05-11	13:00:00	Libre	0
	6	2011-05-12	09:00:00	Libre	0

Figura 4.44. Tabla ‘reserva’

Si no hay horas libres, se le comunica al usuario, y se le pide que introduzca otra fecha. Si hay horas libres, se construye un array con las mismas y se reproduce por voz al usuario, informándole previamente que en el momento en que le interese una de las horas disponibles, diga “sí” o pulse la tecla 1.

Una vez elegida la hora, se accede al fichero *gestiones_reserva_fin.php* con los parámetros ‘horareservar’, ‘fecha_mysql’ e ‘identificador’.

En este fichero se realizan varias acciones: se modifica la hora en el formato 00:00:00 para poder usarla en las sentencias SQL de acceso a la base de datos, se accede a la tabla “reserva” de la base de datos ‘*mariagj_proyecto’* para la fecha y horas indicadas y se actualiza el ‘estado’ a ‘Ocupado’, introduciendo en la columna ‘DNI’ el número de identificación del usuario.

Finalmente, se comunica al usuario que su reserva se ha realizado con éxito. En la Figura 4.45 se puede comprobar el flujo entre los distintos ficheros que conforman esta rutina.

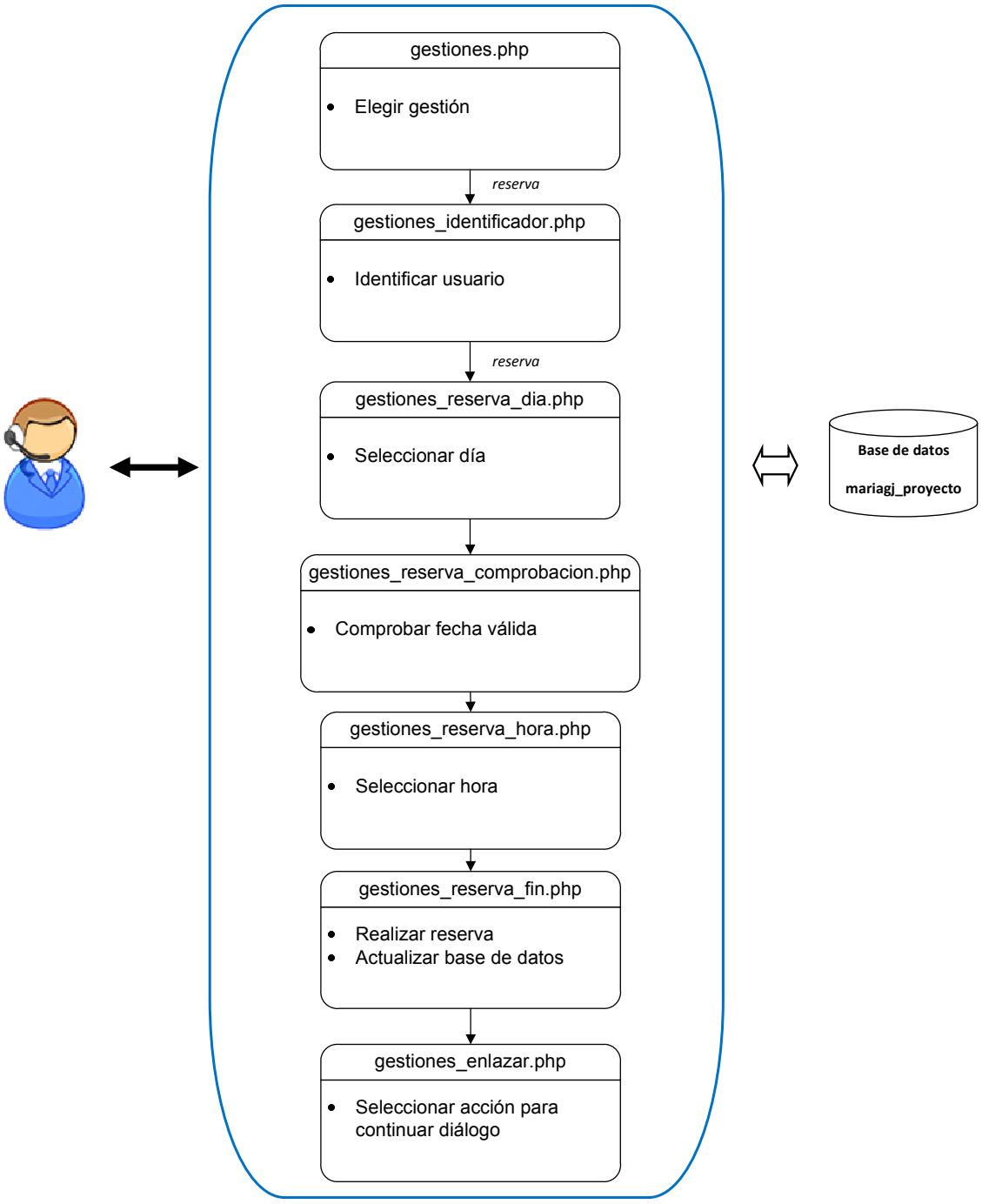


Figura 4.45. Flujo de datos de la rutina reserva de instalación

- **Pedir cita**

Para la opción de pedir cita el procedimiento es idéntico al desarrollado para reservar una instalación, accediendo en este caso a la tabla correspondiente “cita” (donde las citas están espaciadas en intervalos de 30 minutos). Esta modificación se aprecia en la tabla ‘cita’ que se muestra en la Figura 4.46.

	id	dia	hora	estado	DNI
	1	2011-05-11	09:00:00	Libre	0
	2	2011-05-11	09:30:00	Libre	0
	3	2011-05-11	10:00:00	Ocupado	47452060
	4	2011-05-11	10:30:00	Libre	0
	5	2011-05-12	09:00:00	Libre	0
	6	2011-05-12	09:30:00	Libre	0

Figura 4.46. Tabla ‘cita’

En la Figura 4.47 se expone la relación existente entre los ficheros desarrollados para este módulo.

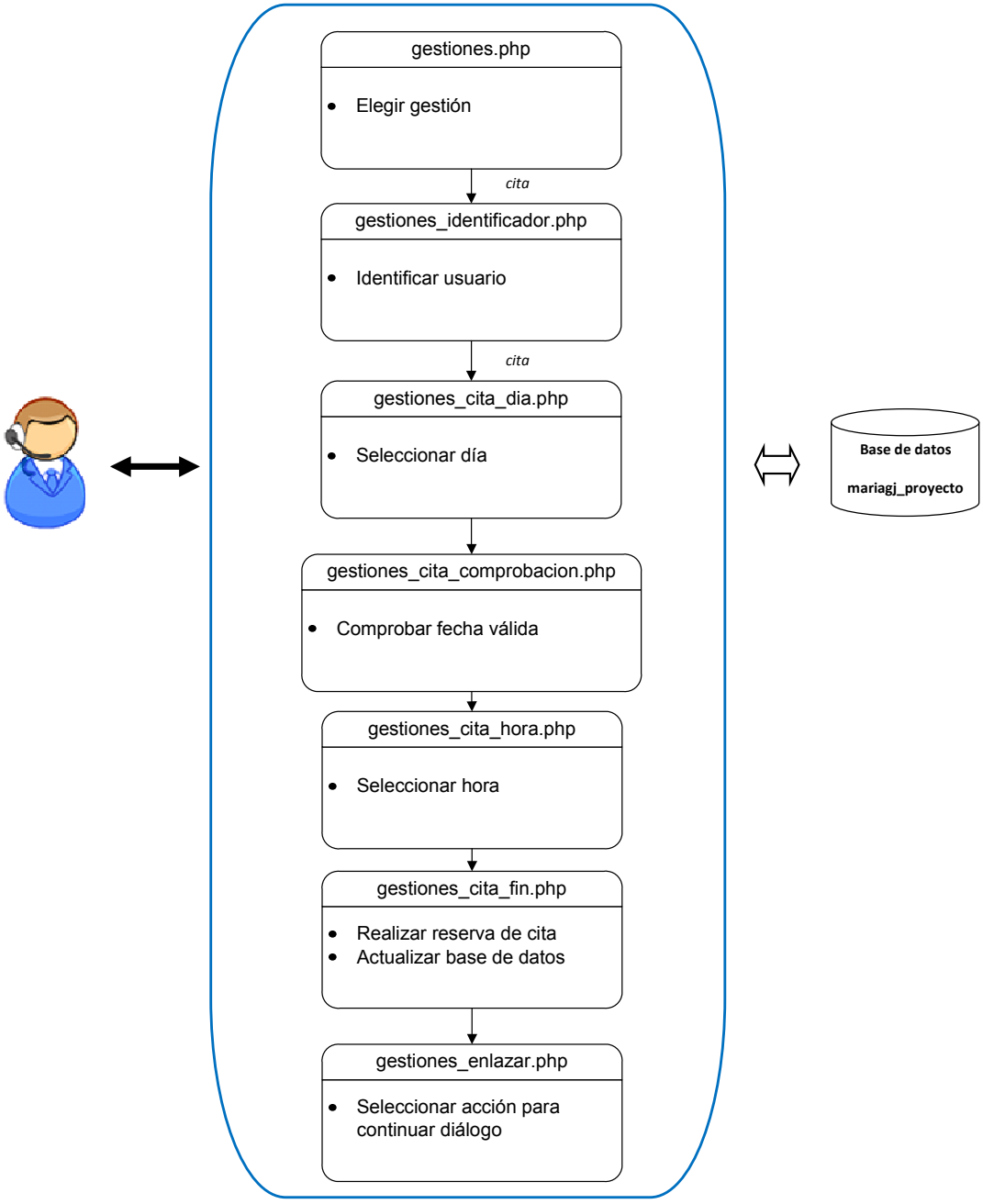


Figura 4.47. Flujo de datos de la rutina pedir cita

Después de cualquiera de los ficheros en los que se termina de realizar una de las anteriores cuatro gestiones, se sigue la ejecución en el fichero *gestiones_enlazar.php*, en el que se pregunta al usuario qué quiere hacer llegado a ese punto:

1- Volver al menú inicial del portal de voz.

2- Realizar otra gestión.

3- Salir de la aplicación.

4.3.3 Escenarios de uso

Se muestran a continuación diferentes diálogos dentro del *módulo Gestiones y Trámites*:

☎ Ejemplo de diálogo en el que el usuario comprueba el estado de sus expedientes (Figura 4.48).

S: A continuación usted podrá realizar gestiones y trámites personales.
Para comprobar si está en un listado diga listado o marque 1.
Para comprobar el estado de sus expedientes diga expedientes o marque 2.
Para realizar una reserva de instalación diga reserva o marque 3.
O si quiere pedir cita para un servicio diga cita o marque 4.

U: Expedientes.

S: Por favor, en primer lugar, indíquenos su DNI diciendo o marcando los 8 números uno a uno.

U: 47452060.

S: Usted ha dicho 47452060.
Accedemos a comprobar el estado de sus expedientes.
Si su DNI es 47452060 su estado de trámites es:
Su certificado de empadronamiento está solicitado.
Acaba de escuchar el estado de sus expedientes.

S: ¿Qué desea hacer ahora?
Para volver al inicio del portal de voz de Alcorcón diga inicio o marque 1.
Para realizar otra gestión diga gestión o marque 2.
O si quiere salir de la aplicación diga salir.

U: Salir.

S: Usted ha seleccionado salir de la aplicación.
Gracias por utilizar nuestro portal de voz. Hasta pronto.

Figura 4.48. Escenario de uso donde se comprueba el estado de expedientes

☎ Ejemplo de diálogo en el que el usuario reserva una instalación (Figura 4.49).

S: Estamos en el apartado de gestiones y trámites personales.
Para comprobar si está en un listado diga listado o marque 1.
Para comprobar el estado de sus expedientes diga expedientes o marque 2.
Para realizar una reserva de instalación diga reserva o marque 3.
O si quiere pedir cita para un servicio diga cita o marque 4.

U: Reserva.

S: Por favor, en primer lugar, indíquenos su DNI diciendo o marcando los 8 números uno a uno.

U: 47452060.

S: Usted ha dicho 47452060.
Accedemos a reservar una instalación.

S: Por favor, diga el año, mes y día en el que le gustaría hacer su reserva.

U: 2011 mayo 5.

S: Usted ha seleccionado la fecha 2011 mayo 5.

S: La fecha es válida. Pasamos a buscar las horas libres para esa fecha.

S: Por favor, diga sí o pulse 1 cuando oiga la hora que desea para su reserva.

S: '11:00','12:00'.

U: Sí.

S: Usted ha seleccionado las ’12:00’.

S: "Su reserva se ha realizado con éxito".

S: ¿Qué desea hacer ahora?
Para volver al inicio del portal de voz de Alcorcón diga inicio o marque 1.
Para realizar otra gestión diga gestión o marque 2.
O si quiere salir de la aplicación diga salir.

U: Salir.

S: Usted ha seleccionado salir de la aplicación.
Gracias por utilizar nuestro portal de voz. Hasta pronto

Figura 4.49. Escenario de uso donde se reserva una instalación

4.4 Módulo Encuesta

4.4.1 Funcionalidad

Otra de las opciones del portal de voz es la realización de encuestas. De esta forma, se puede recoger de forma fácil y rápida la opinión de los ciudadanos de algún tema en particular que se plantee sobre el municipio. La encuesta se realiza de forma totalmente anónima, ya que no se solicita ningún tipo de identificación.

Una vez que el usuario haya terminado de contestar, se le da la opción de escuchar los resultados parciales almacenados hasta el momento para dicha encuesta

4.4.2 Arquitectura

Las acciones principales del ***módulo Encuesta*** se reparten en dos rutinas diferentes. Una de las rutinas almacena los datos de la encuesta en la base de datos (pregunta y posibles respuestas) y crea la gramática correspondiente a las posibles respuestas. Esta rutina se ejecuta por el administrador de la base de datos cada vez que se desee cambiar la encuesta. En la segunda rutina utilizada, cada vez que un usuario elija realizar la encuesta sobre su municipio, se le presenta la pregunta y posibles respuestas, se recogen y almacenan sus resultados, y si lo desea, se le proporcionan los resultados parciales. A continuación se explican detalladamente cada una de estas dos rutinas:

- **Almacenar datos de la encuesta en la base de datos y crear gramática dinámica.**

En el fichero *crear_encuesta.php* se crea la tabla 'encuesta' (Figura 4.50) en la base de datos 'mariagj_*info*'. En ella se almacena el identificador de la encuesta, la pregunta, las cuatro posibles respuestas, los valores acumulativos para cada una de estas respuestas y el número total de votos.

←T→	id	pregunta	respuesta1	respuesta2	respuesta3	respuesta4	valor1	valor2	valor3	valor4	numero_votos
✎ ✕	1	¿A qué partido político tiene pensado votar?	Partido Socialista	Partido Popular	Izquierda Unida	Los Verdes	2	2	2	2	8

Figura 4.50. Tabla 'encuesta'

En este mismo fichero hay dos funciones para gestionar la gramática utilizada en este módulo. Mediante la función *crear_gramática* se crea de forma dinámica un fichero nuevo, *encuesta.xml*, que contiene la gramática con las posibles respuestas para la encuesta. Con la función *borrar_gramática* se borra este fichero. Estas dos funciones se utilizan cuando haya que cambiar la encuesta, eliminando entonces la gramática existente, que estará construida con los valores de la encuesta antigua, y creando dinámicamente una nueva gramática con los nuevos valores que se hayan introducido en la base de datos para la encuesta actual.

- **Presentar la encuesta al usuario y actualizar resultados.**

Una vez que el usuario haya decidido realizar la encuesta, se comienza la rutina en el fichero *encuesta1.php*, donde se le formula la pregunta de la encuesta y se proporcionan las 4 posibles respuestas. Se puede contestar mediante voz o mediante DTMF.

La respuesta es reconocida por el sistema, y en función de ella, se asigna a la variable 'respuesta' un valor numérico del 1 al 4 según la respuesta. Este valor se envía mediante el método *get* y el elemento *submit* al siguiente fichero en la rutina, *encuesta2.php*. En este fichero se recibe el voto, y se aumenta en una unidad el resultado correspondiente en la base de datos, tal y como se muestra en el código de la Figura 4.51.

```
// Se recibe el voto

$voto = $_REQUEST["respuesta"];

//id de la encuesta

$id = 1;

$res="valor".$voto;

$SQLquery = "UPDATE encuesta SET $res = $res+1, numero_votos = numero_votos+1 WHERE
id=$id";

$SQLresult = mysql_query($SQLquery);
```

Figura 4.51. Actualización de resultados después de recibir un voto

Seguidamente, se le pregunta al usuario si quiere escuchar los resultados de la encuesta, y en caso afirmativo se le informa del porcentaje de votos para cada respuesta y del número total de votos hasta el momento.

Finalmente, se accede al fichero *encuesta_enlazar.php*, en el que se le presentan las posibles opciones que tiene para continuar el diálogo:

1- Menú inicio.

2- Salir de la aplicación.

En la Figura 4.52 se observa la relación entre las dos rutinas y los ficheros que forman el *módulo Encuesta*.

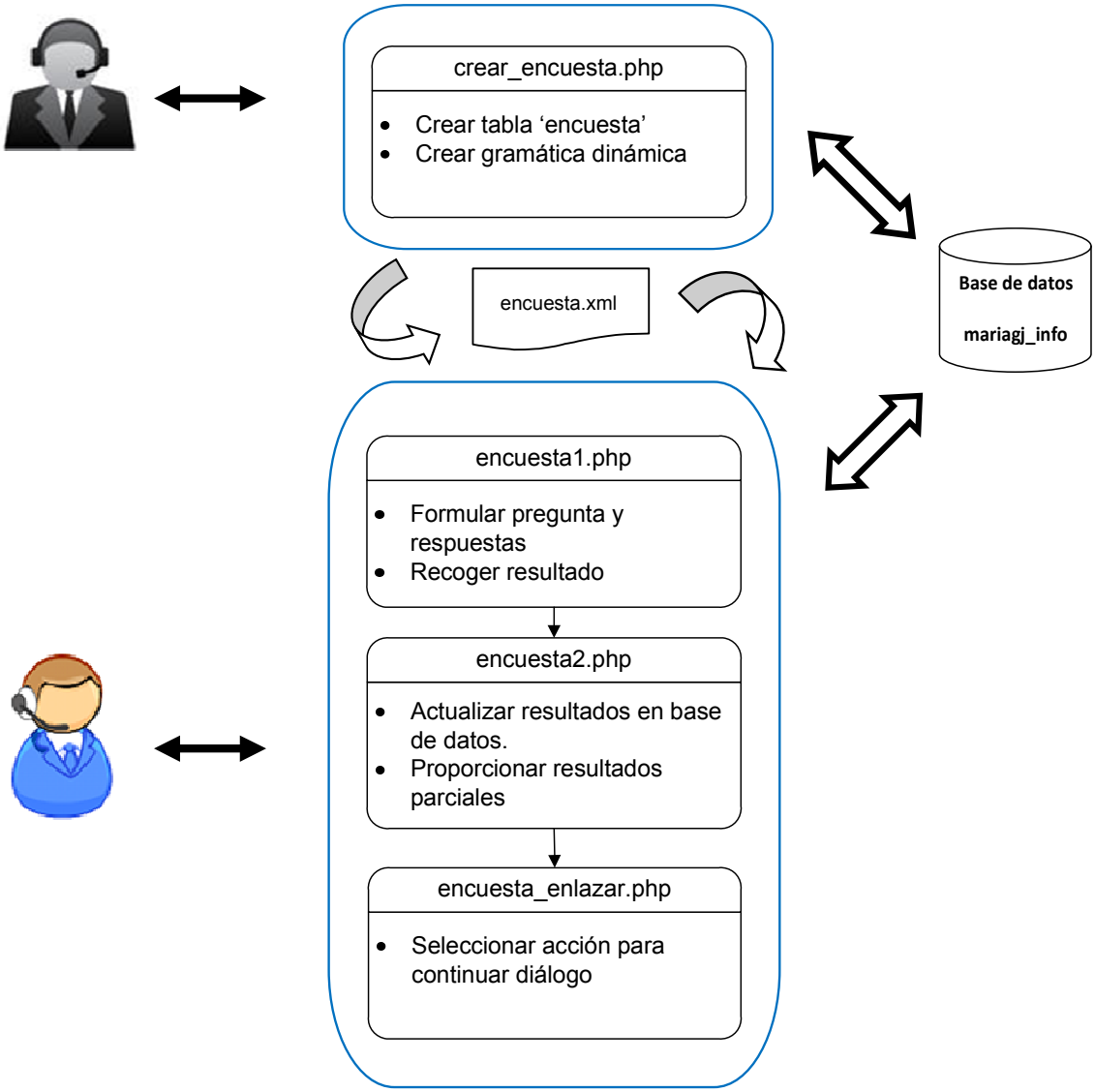


Figura 4.52. Flujo de datos del *módulo Encuesta*

4.4.3 Escenarios de uso

En la Figura 4.53 se presenta un ejemplo de posible diálogo para el *módulo Encuesta.*

S: A continuación usted puede realizar la encuesta municipal y obtener los resultados provisionales.

S: Le formulamos la pregunta y las 4 posibles respuestas.
¿A qué partido político tiene pensado votar?
1: Partido Socialista.
2: Partido Popular.
3: Izquierda Unida.
4: Los Verdes.

U: "Los Verdes".

S: Usted ha seleccionado: Los Verdes.

S: ¿Desea escuchar los resultados de la encuesta?
Diga sí o marque 1, o, diga no o marque 2.

U: Sí.

S: Los resultados de la encuesta, "¿A qué partido político tiene pensado votar?" son:
Respuesta 1: Partido Socialista, 25%.
Respuesta 2: Partido Popular, 25%.
Respuesta 3: Izquierda Unida, 25%.
Respuesta 4: Los Verdes, 25%.
Total de votos emitidos: 8.

S: ¿Qué desea hacer ahora?
1- Volver al inicio del portal de voz de Alcorcón.
2- Salir de la aplicación.

U: Salir aplicación.

S: Usted ha seleccionado salir de la aplicación. Gracias por utilizar nuestro portal de voz. Hasta pronto.

Figura 4.53. Escenario de uso del *módulo Encuesta*

4.5 Módulo Buzón del Ciudadano

4.5.1 Funcionalidad

En el ***módulo Buzón del Ciudadano*** se implementa la funcionalidad de grabar un mensaje de voz por parte del usuario y que este mensaje sea clasificado y almacenado para su posterior tramitación. De esta forma, el ciudadano a cualquier hora y desde

cualquier punto, puede hacer llegar al Ayuntamiento sus solicitudes, quejas, reclamaciones o comentarios.

Este Buzón del Ciudadano sería administrado por la oficina del Defensor del Ciudadano, quienes darían puntual seguimiento a cada una de las solicitudes. Además, en el caso en el que el ciudadano deje sus datos, ya sea teléfono fijo, móvil o correo electrónico, se contactaría con él para dar una respuesta personalizada a su solicitud.

4.5.2 Arquitectura

Las acciones que se realizan para llevar a cabo esta funcionalidad son principalmente dos: grabación del mensaje y su posterior almacenamiento.

- **Grabación del mensaje.**

En primer lugar, se le explica al usuario la forma en que se va a llevar a cabo la grabación, indicándole que debe empezar a grabar su mensaje después de oír el típico pitido, y si desea ser contactado en un futuro, dejar sus datos de contacto. Además, se le comunica que al finalizar de expresar su mensaje, debe mantenerse a la espera, pues se reproducirá el mensaje que ha grabado y se le pedirá confirmación para enviarlo, o la posibilidad de descartarlo y volverlo a grabar.

La grabación de la entrada del usuario se realiza en el fichero *buzon.php*, en el que se utiliza el elemento VoiceXML 'record' con tal fin. A este elemento se le proporcionan varios atributos:

- *name*: nombre identificador que se le da a la grabación.
- *beep*: con este atributo se inserta un tono '*beep*' para indicar al usuario cuando se empieza a grabar.
- *maxtime*: longitud máxima de la grabación en segundos. Se utiliza la variable "msg$.maxtime" para comprobar si este tiempo ha sido superado y, en este caso, se informa al usuario y se vuelve al comienzo del procedimiento para la grabación.
- *finalsilence*: define el tiempo que un usuario puede permanecer en silencio durante la grabación. Si se supera este tiempo, la plataforma interpreta que se ha finalizado de grabar el mensaje.

Por tanto, la sentencia '*record*' con los valores de los atributos utilizados es la mostrada en la Figura 4.54.

```
<record name="grabacion" beep="true" maxtime="20s" finalsilence="4s">
```

Figura 4.54. Sentencia para grabar mensaje en el módulo *Buzón del Ciudadano*

A continuación, se le reproduce al usuario su mensaje, y se accede al fichero *buzon2.php* en el que se solicita su confirmación, preguntándole si quiere enviar este mensaje o descartarlo y volverlo a grabar. En caso que quiera volver a grabarlo, se vuelve al fichero *buzon.php*. Si el usuario confirma enviar el mensaje ya grabado, éste se almacena en la carpeta 'buzon_ciudadano' de la aplicación.

- **Almacenamiento del mensaje.**

Para almacenar el mensaje se utiliza la función *move_uploaded_file* de PHP, que mueve un archivo cargado mediante el mecanismo HTTP POST a una nueva ubicación.

Para poder clasificar la grabación del usuario y posteriormente gestionarla, se le asigna un nombre compuesto por el número de teléfono desde donde se produce la llamada y la fecha de la grabación, esto es, 'callerid-fecha.wav'.

La variable '*callerid*' se obtiene con la expresión '*<var name="callerid" expr="session.connection.remote.uri"/>*'.

La fecha en formato RFC 2822 se consigue mediante la función *date* de PHP, fijando la franja horaria correspondiente a Alcorcón con la función *date_default_timezone_set ("Europe/Paris").* Un ejemplo de este tipo de fecha es *Thu, 14 Apr 2011 16:01:07.*

De esta forma, se almacenan los mensajes perfectamente identificados y clasificados, tal y como se observa en la Figura 4.55.

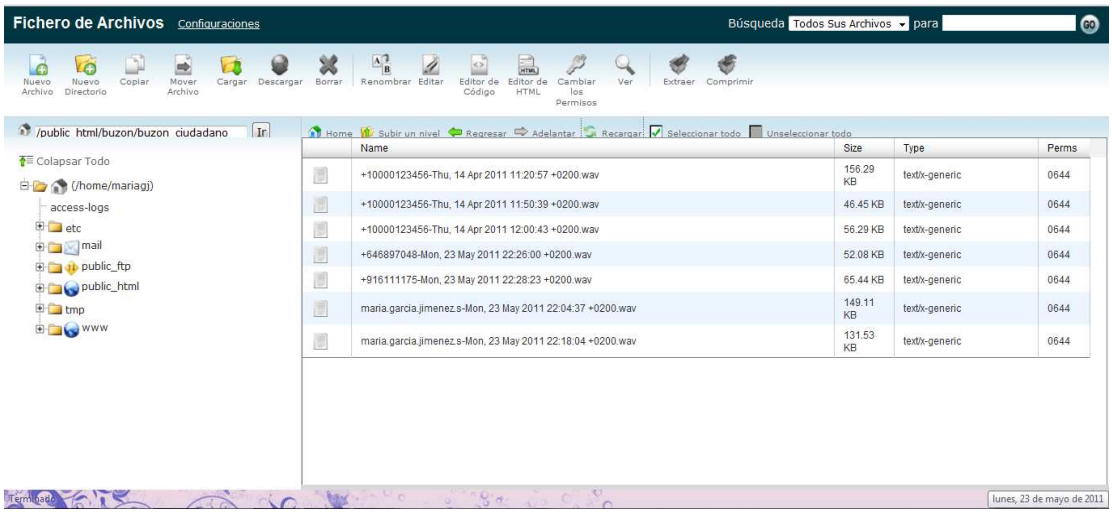


Figura 4.55. Mensajes del *módulo Buzón del Ciudadano* clasificados y almacenados

Finalmente, se comunica al usuario que su fichero ha sido enviado, y se accede al fichero *buzon_enlazar* en el que se le pregunta qué quiere hacer llegado a ese punto:

1- Volver al menú inicial del portal de voz.

2- Salir de la aplicación.

En la Figura 4.56 se observan el flujo de información y la relación entre los ficheros que conforman el *módulo Buzón del Ciudadano*.

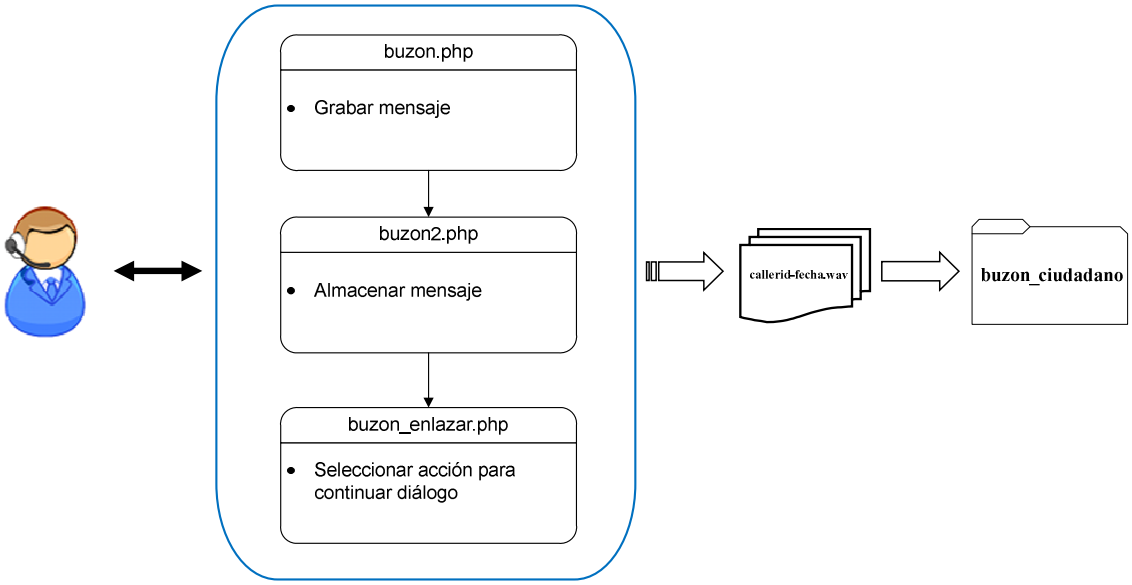


Figura 4.56. Flujo de datos del *módulo Buzón del Ciudadano*

4.5.3 Escenarios de uso

En la Figura 4.57 se presenta un posible ejemplo de diálogo para este *módulo Buzón del Ciudadano*.

S:	Le damos la bienvenida al Buzón del Ciudadano. Desde aquí, podrá hacernos llegar sus sugerencias, quejas y otras observaciones.
S:	Usted podrá grabar su mensaje después de oír la señal. Por favor, en primer lugar indique su nombre, apellidos y teléfono para que podamos responderle adecuadamente, y a continuación su mensaje. Finalmente manténgase a la espera para confirmar su grabación. Gracias.
S:	'beep'
U:	"Buenos días, soy Pepito Pérez y quería dar la enhorabuena por el nuevo servicio del portal de voz de Alcorcón. Gracias."
S:	Su mensaje es el siguiente: "*Buenos días, soy María García y quería dar la enhorabuena por el nuevo servicio del portal de voz de Alcorcón. Gracias.*"
S:	Diga sí o pulse uno para enviar este mensaje. En cambio, diga no o pulse 2 si quiere volver a grabar su mensaje.
U:	Sí.

S:	Hemos guardado su mensaje.
S:	Su grabación se gestionará por nuestro personal, y nos pondremos en contacto con usted. Gracias por colaborar con el Ayuntamiento de Alcorcón.
S:	¿Qué desea hacer ahora? Para volver al inicio del portal de voz de Alcorcón diga inicio o marque 1. O si quiere salir de la aplicación diga salir o marque 2.
U:	Salir.
S:	Usted ha seleccionado salir de la aplicación. Gracias por utilizar nuestro portal de voz. Hasta pronto.

Figura 4.57. Escenario de uso del *módulo Buzón del Ciudadano*

4.6 Módulo Tele-Operador

4.6.1 Funcionalidad

En el ***módulo Tele-Operador*** se transfiere la llamada del usuario al teléfono de la centralita del Ayuntamiento de Alcorcón y, de este modo, el usuario es atendido por un funcionario del Ayuntamiento.

4.6.2 Arquitectura

Esta funcionalidad se implementa en el fichero *operador.php*. Para realizar la transferencia de la llamada se utiliza el elemento de VoiceXML 'transfer', así como de algunos de sus atributos:

- *name*: nombre de la variable transferencia.

- *dest*: especifica el número destino de la transferencia. Al no usar los centros de datos de E.E.U.U., se necesita especificar el número de teléfono en formato E164: +[country code] [number].

- *bridge*: indica si la llamada de salida es una transferencia "*blind*" (una vez que el usuario comienza su transferencia con el número destino, la plataforma se desconecta del usuario) o una transferencia "*bridged*" (la plataforma se mantiene conectada). Voxeo sólo soporta transferencias "bridge", por tanto la aplicación se mantendrá conectada después de haber realizado la transferencia, permitiendo al

usuario volver a la aplicación una vez haya terminado su llamada con el tele-operador de la centralita.

- *connecttimeout*: especifica el tiempo que la plataforma va a emplear para tratar de conectar la llamada al número destino antes de lanzar un evento de "no hay respuesta".

La sentencia con los valores utilizados para estos atributos es la que se muestra en la Figura 4.58.

```
<transfer name="MyCall" dest="tel:+34916648100" bridge="true" connecttimeout="20s">
```

Figura 4.58. Sentencia que transfiere la llamada a la centralita del Ayuntamiento

Para utilizar el elemento '*transfer*' se ha tenido que solicitar permisos especiales en la plataforma Voxeo, en concreto, permisos internacionales de salida de llamada. Hay cuatro posibles casos que pueden ocurrir al hacer la transferencia al número destino. Cada uno de ellos se trata de forma particular:

- La línea está ocupada:

  <if cond="MyCall == 'busy'">

- No hay respuesta en el tiempo especificado por el atributo *connectimeout*.

  <elseif cond="MyCall == 'noanswer'"/>

- La llamada ha finalizado. La centralita ha colgado:

  <elseif cond="MyCall == 'far_end_disconnect'"/>

- La llamada ha sido terminada por el usuario al pulsar una tecla:

  <elseif cond="MyCall == 'near_end_disconnect'"/>

En los dos primeros casos, ya que no se ha podido realizar la transferencia, se accede al fichero *operador_enlazar.php*, en el que se le pregunta al usuario qué quiere hacer a continuación, proporcionándole las siguientes opciones:

1- Intentar de nuevo ser atendido por un operador.

2- Volver al menú inicial del portal de voz.

3- Salir de la aplicación.

En los dos últimos casos, la transferencia se ha realizado con éxito. En el primero de ellos, el usuario ya ha hablado con el operador y el operador cuelga primero. En el segundo caso, es el usuario quien cuelga primero después de hablar con el operador. En ambos casos se sale de la aplicación mediante el elemento '*exit*'.

La Figura 4.59 muestra los ficheros y el flujo de datos del *módulo Tele-Operador.*

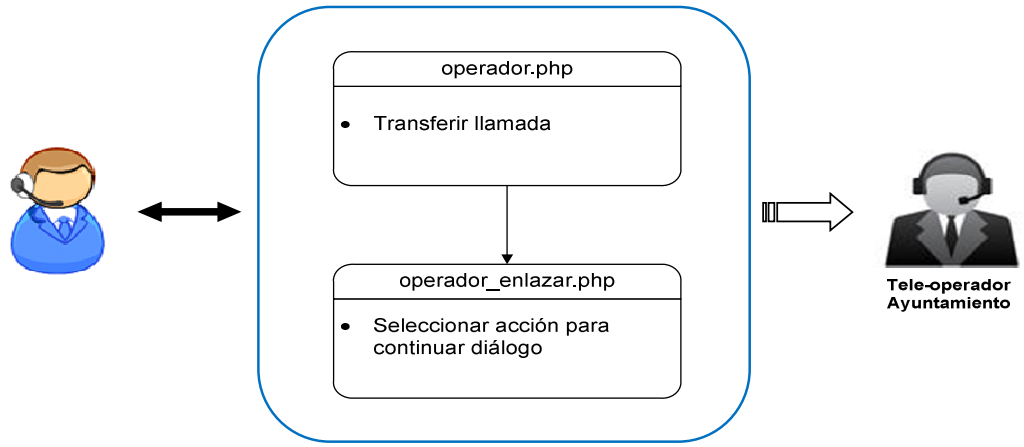


Figura 4.59. Flujo de datos del *módulo Tele-Operador*

4.6.3 Escenarios de uso

Se muestran a continuación dos ejemplos de posibles diálogos para este *módulo Tele-Operador*:

☎ Llamada transferida con éxito (Figura 4.60).

S:	Usted ha elegido ser atendido por un operador.
Tele-operador Ayuntamiento: "…."	

Figura 4.60. Escenario de uso de transferencia de llamada con éxito

☎ Llamada no transferida. Centralita ocupada (Figura 4.61).

S:	Usted ha elegido ser atendido por un operador.
S:	Lo sentimos, la línea está ocupada.
S:	¿Qué desea hacer ahora? Para intentar de nuevo ser atendido por un operador diga operador o marque 1. Para volver al inicio del portal de voz de Alcorcón diga inicio o marque 2. O, si quiere salir de la aplicación diga salir o marque 3.
U:	Salir.
S:	Usted ha seleccionado salir de la aplicación. Gracias por utilizar nuestro portal de voz. Hasta pronto.

Figura 4.61. Escenario de uso de transferencia de llamada sin éxito

Capítulo 5

Evaluación de la aplicación

En este capítulo se describe la evaluación preliminar que se ha llevado a cabo del portal de voz desarrollado para el Proyecto Final de Carrera. Para ello, se ha desarrollado una página web con un cuestionario que recoge la valoración subjetiva de los usuarios que han utilizado el portal de voz. Los resultados se almacenan en una base de datos de la aplicación y posteriormente se han analizado para obtener, de esta forma, la completa evaluación del portal de voz.

5.1 Metodología de evaluación

La evaluación, el estudio del rendimiento y el análisis de la usabilidad de los sistemas de diálogo son procedimientos necesarios para minimizar costes y optimizar resultados en las aplicaciones que hagan uso de ellos.

La evaluación del presente portal de voz se ha llevado a cabo a través de valoraciones de calidad. Para ello, se ha realizado un cuestionario que recoge la opinión subjetiva y el grado de satisfacción de los usuarios, obteniendo así una evaluación cualitativa de la percepción del sistema por parte de los usuarios.

Los aspectos que se han querido analizar son: el grado en el cual el usuario valora que es entendido por el sistema y entiende los mensajes del mismo, la velocidad percibida de la de interacción, la presencia de errores, las seguridad de lo que se debe hacer en cada

momento, la similitud en el comportamiento con un humano y el nivel de satisfacción con el sistema global. Además, información adicional de los usuarios sobre su grado de conocimiento acerca de nuevas tecnologías y uso de sistemas de diálogo sirven para hacerse una idea del perfil de estos usuarios.

El cuestionario elaborado para este fin consta de 10 preguntas (Figura 5.1). Cada pregunta tiene 5 posibles respuestas de las que sólo se puede elegir una de ellas.

1.- Puntúe en una escala de 1 a 5 su conocimiento previo acerca de las nuevas tecnologías para el acceso a información (1= "Bajo", 5= "Alto").

1

2

3

4

5

2.- Puntúe en una escala de 1 a 5 su experiencia previa usando Interfaces de voz o sistemas similares (1= "Bajo", 5= "Alto").

1

2

3

4

5

3.- ¿Cuántas veces ha usado anteriormente Interfaces de Voz?

Nunca

Pocas veces

A veces

Bastante veces

Muchas veces

4.- ¿Qué tal le ha entendido el sistema?

Muy mal

Mal

Regular

Bien

Muy bien

5.- ¿Cómo ha entendido los mensajes generados por el sistema?

Muy mal

Mal

Regular

Bien

Muy bien

6.- En su opinión la interacción fue…

Muy Lenta

Lenta

Adecuada

Rápida

Muy rápida

7.- ¿Ha sido fácil obtener la información que usted solicitaba?

No, ha sido imposible

Sí, pero con gran dificultad

Sí, pero con ciertas dificultades

Sí, ha sido fácil

Sí, ha sido muy fácil

8.- Establezca el nivel de dificultad del sistema para usted.

Muy difícil

Difícil

Normal

Fácil

Muy fácil

9.- ¿Cree usted que el sistema se comportó de manera similar a como lo haría un humano?

Nunca

Casi nunca

A medias

Casi siempre

Siempre

10.- En términos generales, ¿está usted satisfecho con el sistema?

No, nada

Poco satisfecho

Satisfecho

Bastante satisfecho

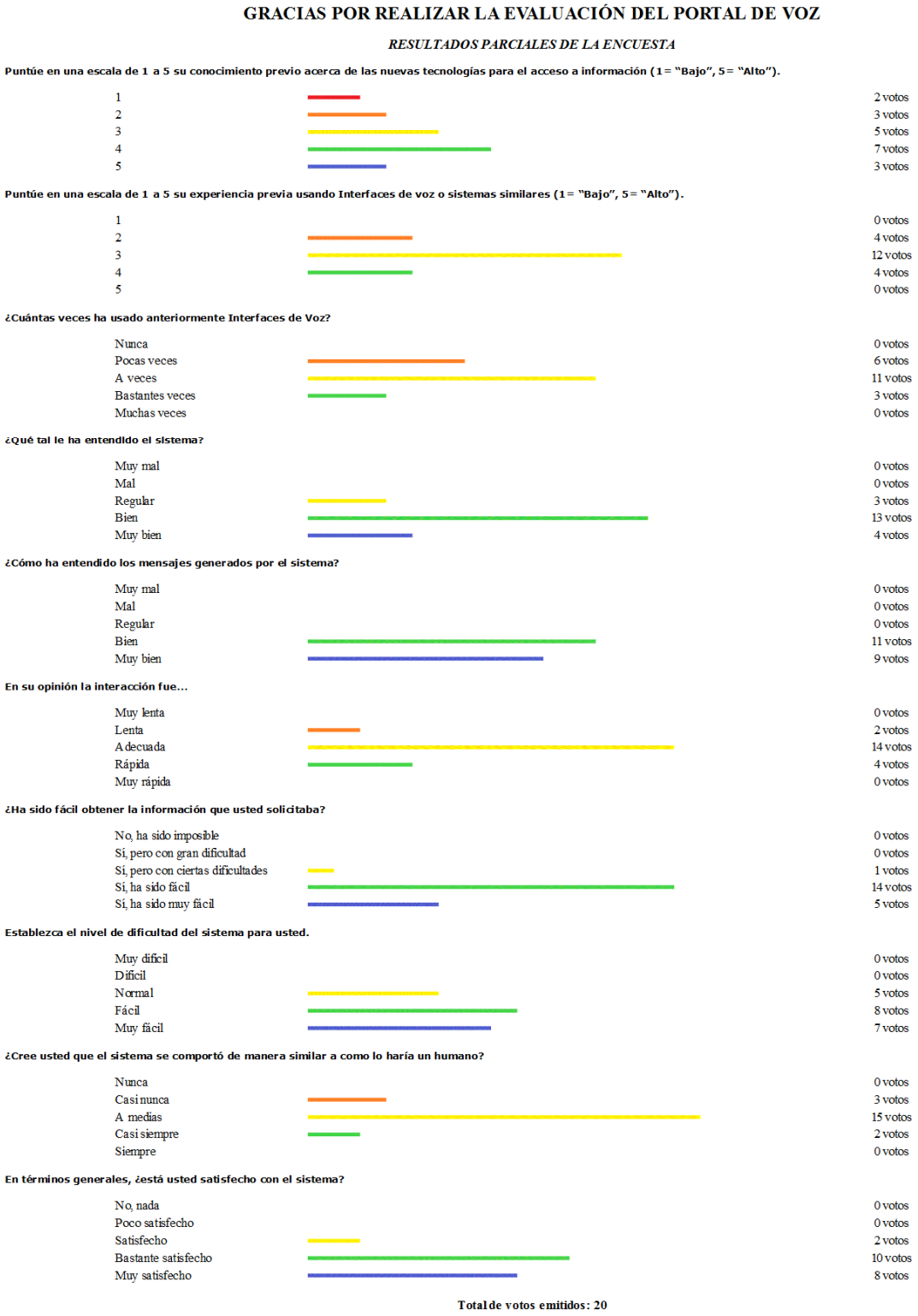
Muy satisfecho

Figura 5.1. Cuestionario desarrollado para la evaluación subjetiva del portal de voz

Para facilitar el registro de las opiniones de los usuarios, se ha desarrollado una página web (http://proyectovxml.x10.mx/evaluacion/evaluacion.php) en la que se muestra el cuestionario tal y como se observa en la Figura 5.2.

Figura 5.2. Página web para realizar la evaluación subjetiva del portal de voz

Una vez que el usuario rellena el cuestionario y pulsa el botón 'Aceptar', es dirigido a otra página web en el que se le da las gracias por colaborar y se muestran los resultados parciales de la encuesta hasta el momento, tal y como se muestra en la Figura 5.3.



GRACIAS POR REALIZAR LA EVALUACIÓN DEL PORTAL DE VOZ

RESULTADOS PARCIALES DE LA ENCUESTA

Puntúe en una escala de 1 a 5 su conocimiento previo acerca de las nuevas tecnologías para el acceso a información (1= "Bajo", 5= "Alto").

1	2 votos
2	3 votos
3	5 votos
4	7 votos
5	3 votos

Puntúe en una escala de 1 a 5 su experiencia previa usando Interfaces de voz o sistemas similares (1= "Bajo", 5= "Alto").

1	0 votos
2	4 votos
3	12 votos
4	4 votos
5	0 votos

¿Cuántas veces ha usado anteriormente Interfaces de Voz?

Nunca	0 votos
Pocas veces	6 votos
A veces	11 votos
Bastantes veces	3 votos
Muchas veces	0 votos

¿Qué tal le ha entendido el sistema?

Muy mal	0 votos
Mal	0 votos
Regular	3 votos
Bien	13 votos
Muy bien	4 votos

¿Cómo ha entendido los mensajes generados por el sistema?

Muy mal	0 votos
Mal	0 votos
Regular	0 votos
Bien	11 votos
Muy bien	9 votos

En su opinión la interacción fue...

Muy lenta	0 votos
Lenta	2 votos
Adecuada	14 votos
Rápida	4 votos
Muy rápida	0 votos

¿Ha sido fácil obtener la información que usted solicitaba?

No, ha sido imposible	0 votos
Sí, pero con gran dificultad	0 votos
Sí, pero con ciertas dificultades	1 votos
Sí, ha sido fácil	14 votos
Sí, ha sido muy fácil	5 votos

Establezca el nivel de dificultad del sistema para usted.

Muy difícil	0 votos
Difícil	0 votos
Normal	5 votos
Fácil	8 votos
Muy fácil	7 votos

¿Cree usted que el sistema se comportó de manera similar a como lo haría un humano?

Nunca	0 votos
Casi nunca	3 votos
A medias	15 votos
Casi siempre	2 votos
Siempre	0 votos

En términos generales, ¿está usted satisfecho con el sistema?

No, nada	0 votos
Poco satisfecho	0 votos
Satisfecho	2 votos
Bastante satisfecho	10 votos
Muy satisfecho	8 votos

Total de votos emitidos: 20

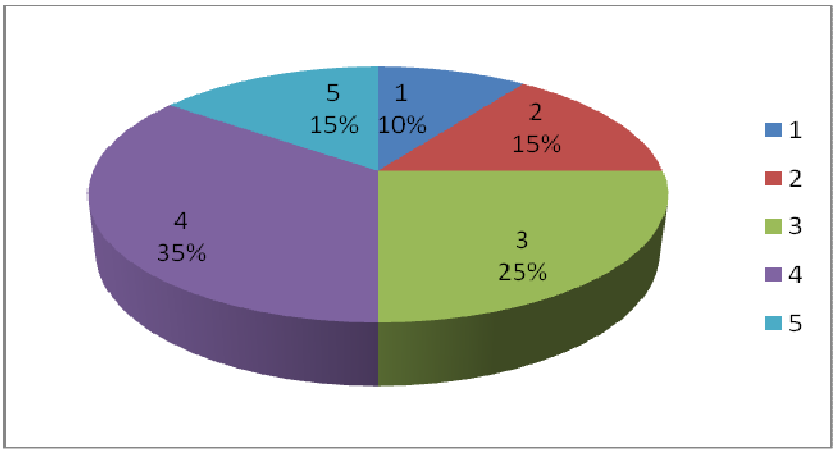
Figura 5.3. Página web con los resultados parciales de la evaluación

Los votos de cada respuesta se almacenan automáticamente en la tabla 'evaluación' de la base de datos '*maryagj_info*', de la que se han obtenido las estadísticas para el análisis de los resultados de la evaluación que se muestran en la siguiente sección de la memoria.

5.2 Resultados de la evaluación

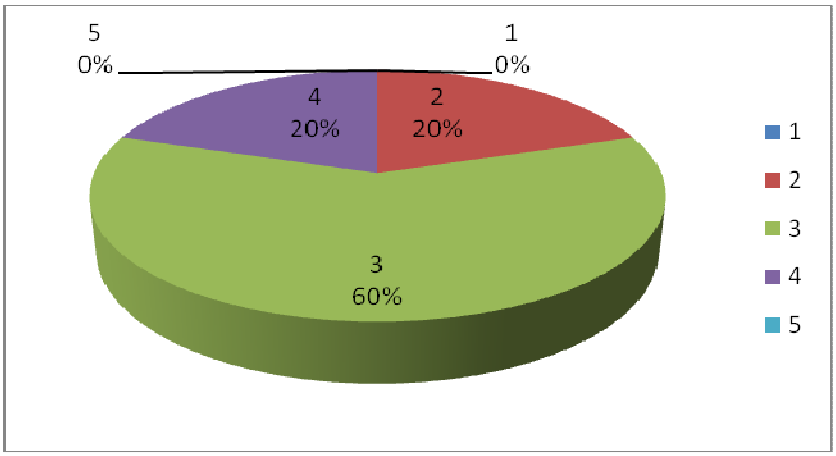
El test de valoración ha sido realizado por 20 usuarios a los que se les explicó las funcionalidades principales del portal de voz y se les pidió que completaran el cuestionario una vez terminaran de usar el portal de voz. Los usuarios eligieron libremente las acciones a realizar y los módulos y submódulos a los que accedieron fueron variados. La Figura 5.4 muestra las estadísticas de los resultados para cada una de las cuestiones que se plantean en el test de valoración subjetiva de la aplicación.

Pregunta 1- Puntúe en una escala de 1 a 5 su conocimiento previo acerca de las nuevas tecnologías para el acceso a información (1= "Bajo", 5= "Alto").



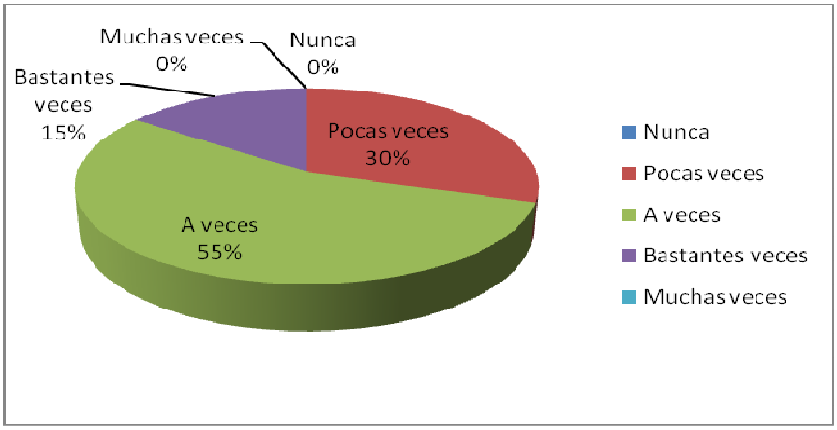
Valor medio	Valor mínimo	Valor máximo	Varianza	Desviación típica
3,3	1	5	1,41	1,18

Pregunta 2- Puntúe en una escala de 1 a 5 su experiencia previa usando Interfaces de voz o sistemas similares (1= "Bajo", 5= "Alto").



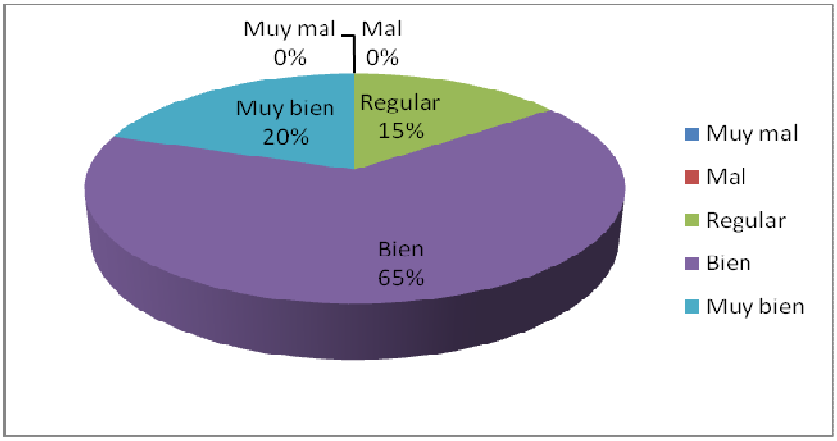
Valor medio	Valor mínimo	Valor máximo	Varianza	Desviación típica
3	2	4	0,4	0,63

Pregunta 3- ¿Cuántas veces ha usado anteriormente Interfaces de Voz?



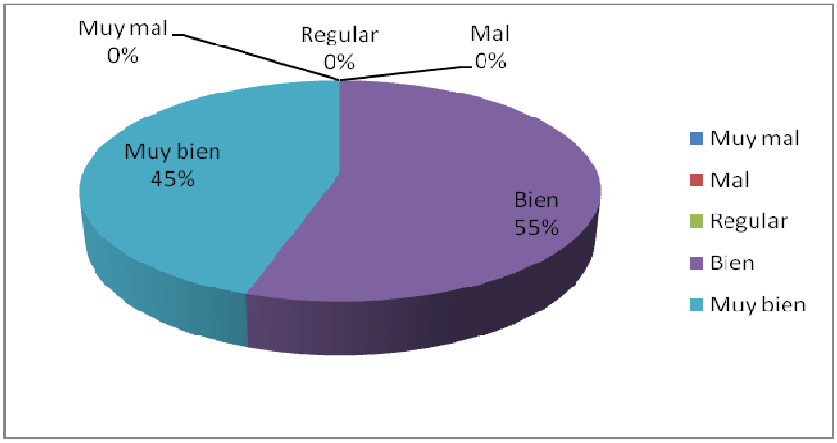
Valor medio	Valor mínimo	Valor máximo	Varianza	Desviación típica
2,85	2	4	0,42	0,65

Pregunta 4- ¿Qué tal le ha entendido el sistema?



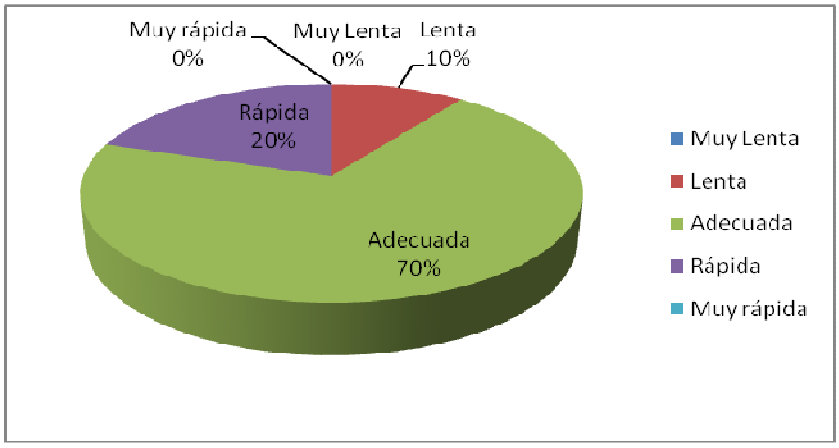
Valor medio	Valor mínimo	Valor máximo	Varianza	Desviación típica
4,05	3	5	0,34	0,58

Pregunta 5- ¿Cómo ha entendido los mensajes generados por el sistema?



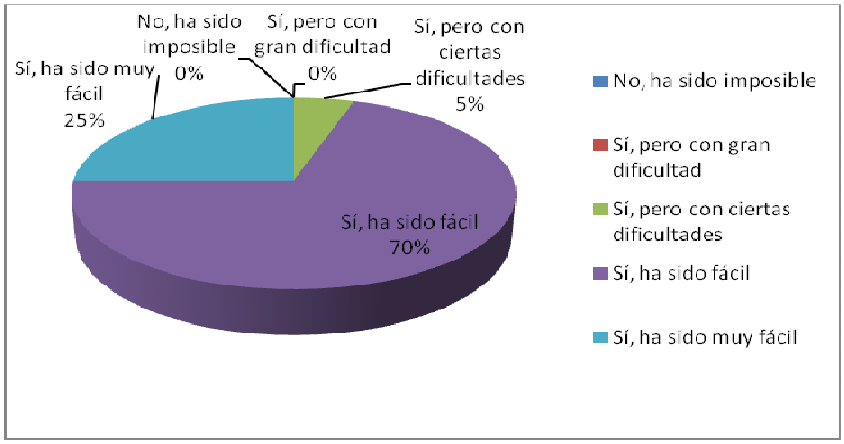
Valor medio	Valor mínimo	Valor máximo	Varianza	Desviación típica
4,45	4	5	0,24	0,49

Pregunta 6- En su opinión la interacción fue…



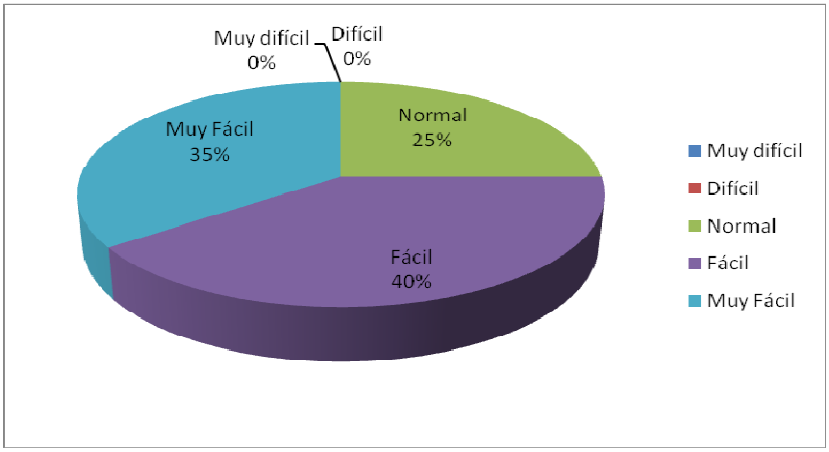
Valor medio	Valor mínimo	Valor máximo	Varianza	Desviación típica
3,1	2	4	0,29	0,53

Pregunta 7- ¿Ha sido fácil obtener la información que usted solicitaba?



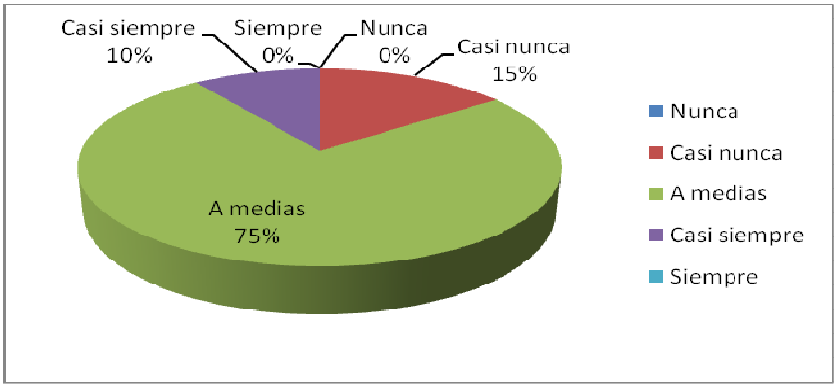
Valor medio	Valor mínimo	Valor máximo	Varianza	Desviación típica
4,2	3	5	0,26	0,50

Pregunta 8- Establezca el nivel de dificultad del sistema para usted.



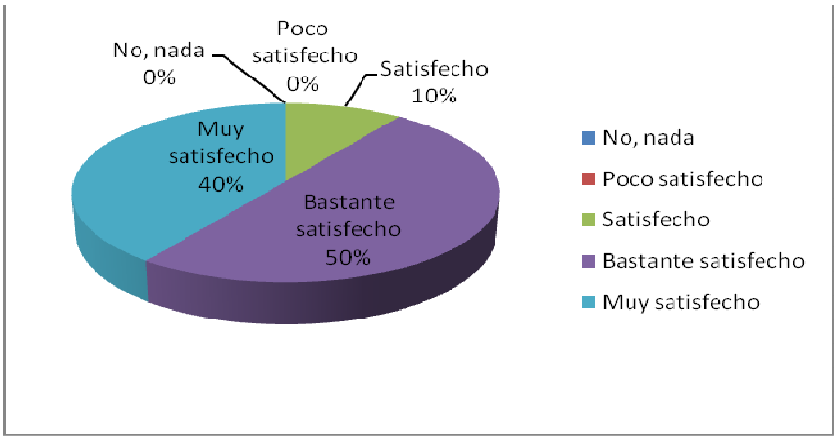
Valor medio	Valor mínimo	Valor máximo	Varianza	Desviación típica
4,1	3	5	0,59	0,76

Pregunta 9- ¿Cree usted que el sistema se comportó de manera similar a como lo haría un humano?



Valor medio	Valor mínimo	Valor máximo	Varianza	Desviación típica
2,95	2	4	0,24	0,49

Pregunta 10- En términos generales, ¿está usted satisfecho con el sistema?



Valor medio	Valor mínimo	Valor máximo	Varianza	Desviación típica
4,3	3	5	0,41	0,64

Figura 5.4. Estadísticas de los resultados de la evaluación subjetiva del portal de voz

Analizando los resultados anteriores, podemos obtener las siguientes conclusiones:

- El nivel de conocimiento de los usuarios a los que se les ha realizado la encuesta sobre nuevas tecnologías y uso de sistemas de diálogo es variado.

- A la gran mayoría de usuarios les ha parecido muy fácil la interacción con el sistema y se valora mayoritariamente como adecuado el ritmo de la misma.

- El sistema ha entendido bien los mensajes de los usuarios y los usuarios han entendido muy bien los mensajes generados por la aplicación.

- En cuanto a la pregunta de la similitud del comportamiento del sistema con un humano, la mayoría creen que el parecido se produce sólo a medias.

- En general, los usuarios obtienen de una forma fácil la información que buscan y están muy satisfechos con el sistema globalmente.

Capítulo 6

Conclusiones y trabajo futuro

En este capítulo se hace un balance del trabajo realizado en el presente Proyecto Final de Carrera. Partiendo de la revisión de los resultados logrados, se comprueba que los objetivos fijados inicialmente han sido cumplidos y se exponen las conclusiones obtenidas. Para finalizar, se presentan los posibles trabajos futuros que se podrían desarrollar sobre la aplicación, de forma que aumentara su eficiencia y el número de funcionalidades.

6.1 Conclusiones

En este Proyecto Final de Carrera se ha desarrollado una aplicación basada en el estándar VoiceXML. Se trata de un portal de voz en el que los usuarios pueden acceder a información de la ciudad de Alcorcón, realizar gestiones y trámites, rellenar encuestas, acceder al Buzón del Ciudadano para dejar sus reclamaciones y solicitudes, y ser transferido al teléfono de la centralita de Alcorcón.

Los servicios ofrecidos por el portal de voz municipal se reparten en módulos por los cuales se conduce al usuario en función de las decisiones que vaya tomando en cada diálogo. Estos módulos están interconectados de tal forma que en una sola llamada es posible realizar más de una acción.

En el ***módulo Información*** los ciudadanos pueden consultar diversos tipos de información, dividida en submódulos de la aplicación:

- Información relacionada con el Ayuntamiento de Alcorcón: equipo de Gobierno, Órganos de Gobierno y datos de las Concejalías del municipio.

- Información de Alcorcón Ciudad: su historia, sus datos geográficos y demográficos y rutas de acceso a la ciudad, además de unas páginas amarillas donde se pueden consultar los datos de contacto de los bares, cafés, restaurantes, tiendas, cines y cartelera, hoteles y hostales de la ciudad.

- Áreas Temáticas: información clasificada según su tipo en 15 áreas temáticas: bienestar social, circulación y transporte, cultura y ocio, deportes, administración, educación, empleo, medio ambiente, parques y jardines, salud y consumo, seguridad, urbanismo, vivienda, hacienda y patrimonio, y cooperación.

- *Noticias*: el ciudadano puede escuchar fecha, título y subtítulos de las noticias municipales.

- *Eventos*: proporciona los eventos que se encuentran publicados en la página web del Ayuntamiento de Alcorcón. Se facilita el área, título, descripción, fecha y lugar de cada uno de los eventos.

- *Información Meteorológica*: se puede consultar el tiempo actual (estado del cielo, temperatura y humedad) y la previsión meteorológica para los siguientes dos días (estado del cielo, temperatura máxima y temperatura mínima).

Además, a través del ***módulo Gestiones y Trámites*** y mediante la identificación por medio del DNI, los habitantes de Alcorcón pueden efectuar diversas gestiones y trámites, como consultar si pertenecen a ciertos listados, comprobar el estado de sus expedientes, reservar una instalación o pedir una cita previa para un servicio municipal.

Otro servicio que proporciona el portal de voz con su ***módulo Encuesta*** es la realización de encuestas municipales. En este módulo se plantea una pregunta con 4 posibles respuestas. Una vez contestada la pregunta, se da la opción al usuario de descubrir los resultados parciales acumulados hasta el momento. Se trata de un sencillo mecanismo en el que se recoge de manera fácil y rápida la opinión ciudadana.

Para que los ciudadanos puedan en cualquier momento exponer sus quejas, solicitudes y reclamaciones se ha añadido el **modulo *Buzón del Ciudadano***. En este módulo se graban los mensajes de los ciudadanos y se almacenan clasificados, para poder atender de forma personalizada cada unos de ellos.

Finalmente, con el ***módulo Tele-Operador*** se ha implementado la opción de poder transferir la llamada al teléfono de la centralita de Alcorcón, para que el usuario sea atendido por un tele-operador del Ayuntamiento en el caso en que lo solicite.

Para poder aplicar a la aplicación desarrollada las funcionalidades y potencia de los **sistemas de diálogo**, en primer lugar se ha realizado un estudio completo de dichos

sistemas. Este estado del arte incluye sus funcionalidades, la arquitectura y los módulos que los conforman y ejemplos existentes en la actualidad.

En cuanto al lenguaje de programación utilizado, **VoiceXML** ha facilitado el desarrollo del portal de voz, permitiendo construir diálogos hablados de forma sencilla y ofreciendo las siguientes funcionalidades: reconocimiento de voz, reconocimiento de entrada DTMF, funciones de telefonía, control del flujo del diálogo, grabación de diálogos, reproducción de voz sintetizada y de ficheros de audio.

Este estándar aprovecha las ventajas de las tecnologías web, como son la integración de servicios de voz y de datos y la compatibilidad con otros lenguajes y minimiza las interacciones cliente/servidor mediante múltiples interacciones por documento XML. Además, separa la interacción del código de usuario de la lógica de servidor, siendo una tecnología independiente de la plataforma que permite la portabilidad y transferencia de datos entre aplicaciones heterogéneas.

La influencia de VoiceXML ha sido decisiva en las nuevas aplicaciones de telefonía y el hecho de estar aprobado por el W3C, y avalado por AT&T, Lucent Technologies, Motorola e IBM (entre más de 350 empresas que en la actualidad son miembros del Foro VoiceXML), hace que sus posibilidades de difusión sean enormes.

Del proceso de implementación del presente portal de voz, como sugerencias de mejora para el desarrollo de aplicaciones con interfaces orales cabe destacar las siguientes consideraciones de diseño:

- Como el usuario no dispone de un apoyo visual al avanzar por los menús, es recomendable ofrecer numerosos mecanismos de ayuda y sistemas de realimentación, que indiquen al usuario qué ocurre en cada momento, además de tener en cuenta el diseñar la aplicación de tal forma que las locuciones no sean excesivamente largas, los menús no tengan gran número de opciones y los niveles en la estructura de navegación sean los mínimos posibles.

- Para que el usuario se habitúe al modo de funcionamiento del servicio, es importante mantener la coherencia y homogeneidad a lo largo de los diálogos, manteniendo un procedimiento de hacer las cosas, por ejemplo, mantener una manera de aceptar y cancelar acciones y no ofrecer un vocabulario distinto en cada sección para acciones parecidas. Este punto es aplicable al uso de pulsaciones DTMF, por ejemplo, que la misma tecla signifique la respuesta positiva: aceptar, sí, avanzar, etc., y otra tecla siempre signifique la respuesta negativa: cancelar, no, retroceder, etc.

- Para el reconocimiento de comandos se ha intentado utilizar expresiones lo más largas posibles, ya que así se favorece la naturalidad de la interacción con el usuario.

- Otra importante característica de la aplicación es que las gramáticas utilizadas se han diseñado para conseguir una mejor manejabilidad de los servicios por parte del usuario, de tal forma que en cada punto se sepa qué opción elegir para encontrar la información buscada o realizar la acción deseada.

- Finalmente, considerando que el usuario va aprendiendo cómo funciona la aplicación según va avanzando por sus menús, la estructura de los diálogos se va

adaptando a la experiencia del usuario, permitiendo la eliminación de explicaciones de los procedimientos a seguir, de modo que se consiga un acceso más rápido y natural.

La plataforma que ha proporcionado la infraestructura y componentes de reconocimiento y síntesis de voz necesarios para crear y utilizar la presente aplicación ha sido **Voxeo**. Las conclusiones a destacar de esta plataforma son la facilidad para la creación y uso de la aplicación de voz, el acceso a ella mediante la asignación de números de teléfonos locales y de *Skype*, además de ofrecer las múltiples herramientas que se han utilizado para el desarrollo y optimización, como son el depurador, el soporte y la completa documentación. Todo ello de forma completamente gratuita.

Como conclusión final, los **objetivos** planteados en un principio **se han cumplido**. Con la realización de este Proyecto Final de Carrera se ha llevado a cabo un estudio completo de los Sistemas de Diálogo, y del lenguaje estándar VoiceXML para poder así aplicar todas las posibilidades de las interfaces orales y las ventajas de VoiceXML en el desarrollo de la nueva aplicación, e implementar las funcionalidades de un portal de voz con el fin de lograr una comunicación lo más efectiva y natural posible.

Mediante el portal de voz se ha elaborado un sistema de acceso a la información y de realización de acciones por parte de los usuarios sin la necesidad de disponer de un navegador, ofreciendo los mismos contenidos que el navegador pero a través de otros dispositivos o medios alternativos como un teléfono, ya sea fijo o móvil, o incluso mediante Skype. De este modo, se proporciona movilidad, flexibilidad y facilidad de acceso, al permitir el uso de la aplicación en entornos y situaciones en los que no podrían ser utilizados un ordenador con teclado y ratón o no existir posible acceso a Internet. Estos entornos podrían ser un automóvil, un lugar al aire libre, etc.

Finalmente, otro de los objetivos más importantes cumplidos es el de facilitar la accesibilidad a los servicios proporcionados por esta aplicación a usuarios que desconocen el uso de un ordenador o de Internet, así como a personas con discapacidades motoras o visuales. Este portal de voz municipal puede promover un mayor conocimiento, acercamiento e integración de estos colectivos dentro del universo de las nuevas tecnologías.

En la actualidad, la mayoría de las aplicaciones de voz utilizadas en los Ayuntamientos para facilitar el acceso a la información y la realización de acciones relacionadas con el municipio se limitan a grabaciones iniciales que te dirigen a una u otra sección para que te atienda personal del Ayuntamiento. Estas aplicaciones se pueden utilizar únicamente mediante la pulsación de las teclas del teléfono. Otros pocos Ayuntamientos añaden cierta tecnología en algunas de sus páginas web que convierte a audio los contenidos web.

Por tanto, este Proyecto Final de Carrera desarrolla e innova un portal de voz municipal pionero en ofrecer una amplia oferta de servicios. Todos estos servicios se pueden realizar de forma desatendida de principio a fin, esto es, no hace falta la transferencia de la llamada a ningún operador del Ayuntamiento. Además, se puede utilizar tanto las teclas del teléfono como la voz en cada unos de los diálogos, facilitando de esta forma la accesibilidad.

La complejidad de la aplicación es elevada debido a la diversidad de cada una de las funciones implementadas, ya que en cada uno de los módulos y submódulos el diseño de las rutinas varía. Además, se debe tener en cuenta la dificultad que supone el hecho de que interactúen en un mismo fichero los lenguajes VoiceXML, PHP y SQL, teniendo en cuenta a la hora de programar las características de cada uno de ellos y el tratamiento de sus respectivas variables. Otro aspecto que supone complejidad adicional es el análisis sintáctico y tratamiento de contenido HTML para extraer y almacenar la información necesaria.

Con este sistema, el Ayuntamiento cumple con la Ley 11/2007 (LAECSP) de acceso electrónico de los ciudadanos a los Servicios Públicos que obliga a fomentar múltiples canales de acceso a la información, en este caso con el teléfono como canal, sin necesidad de desplazarse físicamente a la administración donde se tramitan.

En definitiva, gracias a este portal de voz, los ciudadanos disponen de un nuevo canal útil y accesible por todos que les proporciona de forma fácil y eficaz el acceso telefónico a los servicios de su ciudad.

6.2 Trabajo futuro

Tras la finalización de este proyecto, se detallan a continuación los puntos adicionales sobre los que se podría trabajar para obtener mayores prestaciones en la aplicación desarrollada. Estos trabajos futuros se pueden dividir en dos grandes bloques.

El primer bloque consiste en funcionalidades adicionales que se podrían incorporar al código en los ficheros comunes o en cada uno de los módulos desarrollados:

-El módulo Información podría contener más información desglosada y clasificada en cada una de las áreas ya existentes en el presente sistema.

-Al módulo Gestiones y Trámites se le podrían añadir diversos trámites relacionados con la opción de realizar pagos vía telefónica, proporcionando los mecanismos de seguridad adecuados para llevarlo a cabo mediante un número de tarjeta o una cuenta bancaria.

-En el módulo Tele-operador, al igual que actualmente se realiza una transferencia a la centralita del Ayuntamiento de Alcorcón, se podría facilitar la conexión para poner en contacto al ciudadano con cualquier organismo, oficina, establecimiento de la ciudad o teléfono de emergencia.

-Incluir perfiles de navegación de los ciudadanos. Si se identificara la necesidad del ciudadano a partir de sus interacciones anteriores con la herramienta, se podría optimizar la navegación por el portal de voz para acortar lo máximo posible el acceso a los servicios más visitados. Adicionalmente, teniendo en cuenta el número de accesos previos, podrían modificarse los mensajes proporcionados por el sistema. Por ejemplo, si se trata de un usuario que lo utilizara por primera vez, se le darían completas indicaciones

del funcionamiento para facilitarle el uso y disminuir los errores que se pudieran producir. En cambio, si el usuario ya fuera experto en su utilización, no habría que explicarle qué decir exactamente en cada momento y se le ofrecerían directamente las operaciones que en su historial resultaran más comunes, consiguiendo de esta forma una mayor velocidad en realizar la acción deseada.

El segundo bloque de trabajos futuros consiste en la incorporación de funcionalidades globales para el sistema, así como la utilización de otros tipos de tecnologías que, unidas a las ya utilizadas, proporcionarían funcionalidades adicionales a la aplicación:

Idiomas

Alcorcón, con una población de 173.491 habitantes, tiene una tasa de inmigración del 13.5%. Si la herramienta desarrollada se pudiera utilizar en otros idiomas, extendería y facilitaría el uso a la población inmigrante.

El requisito fundamental para añadir este servicio sería que la aplicación estuviera alojada en una plataforma que proporcionara un reconocedor de voz y un sintetizador de texto en voz que soportara los idiomas añadidos. En este caso, Voxeo con su *Prophecy 9 Multi-Language VXML* permite la utilización de los principales idiomas. En cuanto a la implementación, en el formulario inicial se daría la opción al usuario de elegir el idioma. A partir de este punto se gestionaría el resto de la arquitectura de la aplicación según el idioma elegido.

Herramienta de gestión y cuadro de mando

La presente aplicación requiere un sencillo procedimiento de administración, ya que con la simple ejecución de un fichero PHP se carga la información dinámica en la base de datos, y se crean las gramáticas dinámicas de las páginas amarillas y la encuesta. Para facilitar esta administración del sistema a los funcionarios del Ayuntamiento, se podría añadir una herramienta de gestión y cuadro de mando que simplificara la carga de esta información dinámica y la modificación de los menús, gramáticas e información estática de la base de datos. Adicionalmente, esta herramienta podría proporcionar estadísticas globales de la interacción con el sistema.

Sistemas de diálogo multimodal

Uno de los objetivos fundamentales actualmente en la investigación en los sistemas de diálogo es superar las limitaciones de la interacción basada exclusivamente en el habla.

En una interacción multimodal el usuario no está restringido a utilizar el habla como único canal de comunicación, sino que puede utilizar varios dispositivos de entrada, como por ejemplo un teclado, un ratón, un micrófono, una cámara, una pantalla sensible al tacto, una PDA, etc. Asimismo, el sistema multimodal puede utilizar diversos canales de salida para proporcionar información al usuario como por ejemplo, voz, texto, gráficos o imágenes.

Para la creación de aplicaciones multimodales cabe destacar dos tecnologías que añadirían grandes posibilidades a la presente aplicación desarrollada:

- Servicios de vídeo + voz con VoiceXML

Los nuevos avances en los dispositivos móviles y en las redes de comunicaciones están abriendo la puerta al mundo de los servicios de vídeo, y por consiguiente a los sistemas IVVR (Interactive Voice and Video Response) [IVVR].

El vídeo añade un nuevo canal y dimensión a la interacción hombre-máquina, proporcionando como resultado nuevas posibilidades más expresivas e intuitivas para las aplicaciones, como son:

-Vídeo mail.

-Video-servicios de información.

-Vídeo-Servicios de ocio.

-Vídeo multiconferencias.

-Vídeo call-center.

VoiceXML utiliza URIs para referenciar estos recursos de voz y vídeo, con lo que tan sólo habría que especificar el tipo de contenido usado.

- XHTML + VoiceXML

Otra tecnología para la creación de aplicaciones multimodales es el lenguaje XHTML+Voice [XHTML+VOICE], que permite implementar en el mismo entorno tanto elementos visuales como elementos de voz.

XHTML+Voice, comúnmente conocido como X+V, es un lenguaje XML desarrollado por el W3C para describir interfaces multimodales de usuario. La interacción visual se define mediante la programación de páginas web utilizando XHTML. Los componentes orales se definen mediante la incorporación de un subconjunto del lenguaje VoiceXML. La Interfaz entre la comunicación oral y los componentes visuales de X+V se realiza a través de una combinación de ECMAScript, JavaScript, XML y eventos.

Si se codificara la aplicación en este lenguaje, se podría rellenar un formulario web mediante el teclado y el ratón, o mediante la voz. Esta entrada paralela facilita accesos más rápidos y aporta las ventajas que poseen los sistemas multimodales, como por ejemplo, facilitar el acceso a la aplicación a cualquier usuario, ya que podría navegar a través del dispositivo móvil con la voz o hacer uso de los elementos gráficos y visuales que se le proporcionan. Se crea de este modo un entorno más agradable, dotando de naturalidad y eficiencia al sistema.

Presupuesto

En esta sección se presenta el presupuesto del proyecto. En él se contempla la duración de las distintas fases y tareas, y se incluye un desglose de costes de personal, costes de material y costes totales.

1. Tareas

Fase 1: Planificación.

- Estudio de los sistemas de diálogo.
  - Duración: 21 días.

- Planificación y análisis de requisitos del portal de voz.
  - Duración: 7 días.

- Estudio de las tecnologías necesarias.
  - Duración: 30 días.

Fase 2: Desarrollo.

- Análisis y diseño inicial.
  - Duración: 7 días.

- Implementación del sistema.
  - Duración: 75 días.

- Pruebas unitarias.
  - Duración: 15 días.

- Pruebas de integración y de sistema.
  - Duración: 7 días.

- Evaluación de la aplicación.
  - Duración: 15 días.

Fase 3: Documentación.

- Memoria del Proyecto Final de Carrera.
  - Duración: 30 días.

- Preparación de la presentación.
  - Duración: 7 días.

2. Recursos

Para la realización del presente proyecto se han utilizado los siguientes recursos:

• **Recursos software:**
- Plataforma Voxeo: **0€**
- Skype: **0€**
- Licencia Microsoft Office 2007: **200 €**
- Editor de programación JVoxEdit: **0 €**
- Editor de programación Notepad ++: **0 €**

• **Recursos hardware:**
- Ordenador Portátil : **900 €**
- Auriculares con micrófono: **100€**
- Servidor X10Hosting: **0€**

• **Recursos humanos:**

En la realización de este proyecto han participado dos personas, el director de proyecto y el desarrollador.

Las horas dedicadas al proyecto han sido de una media de 8 horas diarias, contemplando semanas de 7 días.

- Coste de un Ingeniero: 30 €/hora.

3. Resumen de costes

El coste de recursos humanos de la aplicación se resume en la Tabla 5.1.

TAREA	DÍAS	IMPORTE
FASE1	58	14.400 €
FASE2	119	30.960 €
FASE3	37	13.680 €
Subtotal	**214**	**59.040 €**

Tabla 5.1. Detalle de Costes de Recursos Humanos del Proyecto

El coste total del sistema se presenta en la Tabla 5.2:

CONCEPTO	IMPORTE
Recursos Humanos	59.040 €
Recursos Software	200 €
Recursos Hardware	1.000 €
Subtotal	60.240 €
(18% IVA)	10.843 €
TOTAL	**71.083 €**

Tabla 5.2. Detalle de Coste Total del Proyecto

El presupuesto total de este proyecto asciende a la cantidad de **SETENTA Y UN MIL OCHENTA Y TRES EUROS.**

Madrid a 1 de Julio de 2011

Fdo. María García Jiménez

Glosario

- **CGI** (*Common Gateway Interface*)**:** es una importante tecnología de la World Wide Web que permite a un cliente (navegador web) solicitar datos de un programa ejecutado en un servidor web. CGI especifica un estándar para transferir datos entre el cliente y el programa. Es un mecanismo de comunicación entre el servidor web y una aplicación externa cuyo resultado final de la ejecución son objetos MIME. Las aplicaciones que se ejecutan en el servidor reciben el nombre de CGIs.

- **CALLXML**: es un lenguaje basado en etiquetas XML creado por Voxeo Community, que permite implementar aplicaciones telefónicas basadas en voz convencional o voz sobre IP (VoIP) que interactúan con servidores CallXML, los cuales usan el código de la aplicación para controlar las llamadas. CallXML ha sido diseñado para facilitar la implementación de aplicaciones web que puedan interactuar con cualquier tipo de llamada telefónica, soportando la iniciación de llamadas salientes, conferencias, interacción entre múltiples llamadas, funciones encuéntreme-sígame, etc.

- **CCXML** (Call Control Extensible Language): es un lenguaje de marcación basado en XML diseñado para añadir capacidades de control de llamada a sistemas de telefonía de forma estándar. CCXML ha sido diseñado para implementar aplicaciones telefónicas complejas, con posibilidad de lanzar nuevas aplicaciones IVR (Interactive Voice Response) en llamadas subsiguientes. CCXML está impulsado por el W3C.

- **DTMF** (Dual-Tone Multi-Frequency signaling): es un sistema de marcación por tonos, también llamado sistema multi-frecuencial. En este sistema cuando el usuario pulsa en el teclado de su teléfono la tecla correspondiente al dígito que quiere marcar, se envían dos tonos, de distinta frecuencia, uno por columna y otro por fila en la que esté la tecla, que la central descodifica a través de filtros especiales, detectando instantáneamente que dígito se marcó.

- **ECMAScript**: es una especificación de lenguaje de programación publicada por ECMA International. El desarrollo empezó en 1996 y estuvo basado en el popular lenguaje JavaScript propuesto como estándar por Netscape Communications Corporation. Actualmente está aceptado como el estándar ISO 16262. ECMAScript define un lenguaje de tipos dinámicos ligeramente inspirado en Java y otros lenguajes del estilo de C. Soporta algunas características de la programación orientada a objetos mediante objetos basados en prototipos y pseudoclases. La mayoría de navegadores de Internet incluyen una implementación del estándar ECMAScript.

- **IMAP** (Internet Message Access Protocol): es un protocolo de red de acceso a mensajes electrónicos almacenados en un servidor. Mediante IMAP se puede tener acceso al correo electrónico desde cualquier equipo que tenga una conexión a Internet. IMAP tiene varias ventajas sobre POP, que es el otro protocolo empleado para obtener correo desde un servidor. Por ejemplo, es posible especificar en IMAP carpetas del lado servidor. Por otro lado, es más complejo que POP ya que permite visualizar los mensajes de manera remota y no descargando los mensajes como lo hace POP.

- **IVR (**Interactive Voice Response): es un sistema telefónico que capaz de recibir una llamada e interactuar con el humano a través de grabaciones de voz y el reconocimiento de respuestas simples, como "sí", "no" u otras. Se trata de un sistema automatizado de respuesta interactiva, orientado a entregar y/o capturar información a través del teléfono, permitiendo el acceso a servicios de información u otras operaciones.

- **MySQL**: es un sistema de gestión de bases de datos relacionales multihilo, multiusuario y robusto. El software MySQL tiene licencia dual, pudiéndose usar de forma gratuita bajo licencia GNU o bien adquiriendo licencias comerciales de MySQL AB en el caso de no desear estar sujeto a los términos de la licencia GPL. MySQL es una marca registrada de MySQL AB.

- **PHP (**Hypertext Pre-processor): es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente para la interpretación del lado del servidor (*server-side scripting*) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica.

- **POP3** (Post Office Protocol): En informática se utiliza el POP3 en clientes locales de correo para obtener los mensajes de correo electrónico almacenados en un servidor remoto. Es un protocolo de nivel de aplicación en el Modelo OSI.

- **PTSN** (Public Switched Telephone Network): es una red con conmutación de circuitos tradicional optimizada para comunicaciones de voz en tiempo real. Cuando se llama a alguien, se cierra un conmutador al marcar y se establece así un circuito con el receptor de la llamada. PSTN garantiza la calidad del servicio (QoS) al dedicar el circuito a la llamada hasta que se cuelga el teléfono.

- **Script**: es un programa usualmente simple, que por lo regular se almacena en un archivo de texto plano. Los script son casi siempre interpretados, pero no todo programa interpretado es considerado un script. El uso habitual de los scripts es realizar diversas tareas como combinar componentes, interactuar con el sistema operativo o con el usuario.

- **Sendmail**: es un popular "agente de transporte de correo" (MTA - Mail Transport Agent) en Internet, cuya tarea consiste en "encaminar" los mensajes de forma que estos lleguen a su destino. Se afirma que es el más popular MTA, compatible con sistemas Unix y el responsable de la mayoría de envíos del correo de Internet, aunque se critica su alto número de alertas de seguridad además de no ser sencillo de configurar.

- **SIP (**Session Initiation Protocol): es un protocolo desarrollado por el grupo de trabajo MMUSIC del IETF con la intención de ser el estándar para la iniciación, modificación y finalización de sesiones interactivas de usuario donde intervienen elementos multimedia como el video, voz, mensajería instantánea, juegos en línea y realidad virtual.

- **Sistema de diálogo**: son programas informáticos cuya finalidad es interactuar con los usuarios oralmente o de forma multimodal para proporcionar diversos servicios.

- **URI** (**Uniform Resource Identifier**): es una cadena corta de caracteres que identifica inequívocamente un recurso (servicio, página, documento, dirección de correo electrónico, enciclopedia, etc.). Normalmente estos recursos son accesibles en una red o sistema. Algunos URI pueden ser URL, URN o ambos.

- **VoiceXML** (Voice eXtensible Markup Language): es un formato XML estándar del W3C para la especificación de diálogos interactivos entre un humano y una máquina.

- **Voxeo**: es una plataforma de implementación del estándar VoiceXML que cumple por completo con las especificaciones del W3C para VoiceXML 2.0 y 2.1.

- **W3C** (Consorcio World Wide Web): es una asociación internacional formada por organizaciones miembros del consorcio, personal y el público en general, que trabajan conjuntamente para desarrollar estándares web. W3C pretende guiar la web hacia su máximo potencial a través del desarrollo de protocolos y pautas que aseguren el crecimiento futuro de la web.

- **XML** (eXtensible Markup Language): es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG, MathML.

Bibliografía

[ATOS]
Disponible: http://www.es.atosorigin.com/es-es/servicios/soluciones/default.htm [04 de febrero de 2011]

[ATHOSMAIL]
Disponible: http://www.cs.uta.fi/hci/spi/index.html [04 de febrero de 2011]

[BBVA]
Disponible: http://www.naturalvox.com [04 de febrero de 2011]

[BEVOCALCAFE]
Disponible: http://cafe.bevocal.com/ [09 de febrero de 2011]

[CAL08]
Callejas, Zoraida. Desarrollo de Sistemas de Diálogo Oral Adaptativos y Portables: Reconocimiento de Emociones, Adaptación al idioma y Evaluación de Campo. Tesis doctoral. Universidad de Granada, 2008.

[CALLXML]
Disponible: http://docs.voxeo.com/callxml/3.0/home.htm [03 de junio de 2011]

[CCXML]
"Voice Browser Call Control: CCXML Version 1.0". RJ Auburn et al. W3C Proposed Recommendation 10 May 2011. Disponible: http://www.w3.org/TR/ccxml [03 de junio de 2011]

[CINEENTRADAS]
Disponible: http://www.ydilo.com/esp/index.php [04 de febrero de 2011]

[COGNIRON]
Disponible: http://www.cogniron.org/final/Home.php [04 de febrero de 2011]

[COLLAGEN]
Disponible: http://www.merl.com/projects/collagen [04 de febrero de 2011]

[CONITA]
Disponible: http://www.conita.com [09 de febrero de 2011]

[CONQUEST]
Disponible: http://www.conquest-dialog.org [04 de febrero de 2011]

[COMPANIONS]
Disponible: http://www.companions-project.org [04 de febrero de 2011]

[CSS2]
"*Cascading Style Sheets, level 2, CSS2 Specification*", Bos et al. W3C Recommendation, May 1998. Disponible: http://www.w3.org/TR/REC-CSS2/ [08 de febrero de 2011]

[DARPA]
Disponible: http://www.darpa.mil/default.aspx [04 de febrero de 2011]

[DCMI]
"Dublin Core Metadata Initiative ", a Simple Content Description Model for Electronic Resources. Disponible: http://dublincore.org/ [08 de febrero de 2011]

[DIHANA]
David Griol, Lluís F. Hurtado, Encarna Segarra, Emilio Sanchis: A statistical approach to spoken dialog systems design and evaluation. Speech Communication 50(8-9): 666-682 (2008)

[DIXERIT]
Disponible:
http://www.dixerit.co.uk/ [10 de junio de 2011]

[ECMASCRIPT]
" *Standard ECMA-262 ECMAScript Language Specification* ", Standard ECMA-262, diciembre 1999. Disponible:
http://www.ecma-international.org/publications/standards/Ecma-262.htm [08 de febrero de 2011]

[ELOQUANT]
Disponible:
http://www.eloquant.com [09 de febrero de 2011]

[ENC]
Disponible:
http://www.ontsi.red.es/section/index.action?sec=394 [20 de mayo de 2011]

[GRI07]
Griol, David. Desarrollo y evaluación de Diferentes Metodologías para la Gestión Automática del Diálogo. Tesis Doctoral. Universidad Politécnica de Valencia, 2007.

[HEYANITA]
Disponible:
http://www.heyanita.com [09 de febrero de 2011]

[HTML]
"HTML 4.01 Specification ", Dave Raggett et al. W3C Recommendation, December 1999. Disponible:
http://www.w3.org/TR/1999/REC-html401-19991224/ [08 de febrero de 2011]

[HTTP]
" *Hypertext Transfer Protocol -- HTTP/1.1* ", IETF RFC 2616, 1999.
Disponible:
http://www.ietf.org/rfc/rfc2616.txt [08 de febrero de 2011]

[HUR+05]
Lluís Hurtado, Fernando Blat, S. Grau, David Griol, Emilio Sanchis, Encarna Segarra, F. Torres. Sistema de diálogo para el Proyecto DIHANA. Procesamiento del Lenguaje Natural, ISSN 1135-5948, Nº. 35, 2005, págs. 453-454.

[IBM]
Disponible:
http://www.ibm.com [09 de febrero de 2011]

[IKEA]
Disponible:
http://193.108.42.79/ikea-es/cgi-bin/ikea-es.cgi [04 de febrero de 2011]

[ITSPOKE]
Disponible:
http://oldwww.cs.pitt.edu [04 de febrero de 2011]

[IVVR]
Disponible:
http://ivvr-services.blogspot.com/ [20 de mayo de 2011]

[JVOICEXML]
Disponible:
http://www.jvoicexml.sourceforge.net [09 de febrero de 2011]

[JUPITER]
Disponible:
http://groups.csail.mit.edu/sls/research/jupiter.shtml [04 de febrero de 2011]

[LAECSP]
Comisión de Modernización y Calidad de la FEMP. Guía práctica de la Ley 11/2007, de acceso electrónico de los ciudadanos a los Servicios Públicos (LAECSP). (FEMP, 2009).

[LISTEN]
Mostow, J., 2008. Experience from a reading tutor that listens: Evaluation purposes, excuses, and methods. In: Kinzer, C., Verhoeven, L. (Eds.), Interactive Literacy Education: Facilitating Literacy Environments Throught Technology. New York: Lawrence Erlbaum Associates, Taylor and Francis, págs. 117-148.

[LLI06]
Llisterri, J. (2006) "Introducción a los sistemas de diálogo", in LLISTERRI, J.-MACHUCA, M. J. (Eds.) *Los sistemas de diálogo*. Bellaterra - Soria: Universitat Autònoma de Barcelona, Servei de Publicacions - Fundación Duques de Soria (Manuals de la Universitat Autònoma de Barcelona, Lingüística, 45), págs. 11-21.

[LOP]
Disponible:
http://www.ugr.es/~rlopezc/sistemas_dialogo.htm [20 de mayo de 2011]

[LOP05]
López-Cózar, R., and Araki, M.: 'Spoken, Multilingual and Multimodal Dialogue Systems: Development and Assessment' (Wiley, 2005).

[LOQUENDO]
Disponible:
http://www.loquendo.com [09 de febrero de 2011]

[LUCENT]
Disponible:
http://www.lucent.com [09 de febrero de 2011]

[MCT04]
McTear, Michael F.: 'Spoken Dialogue Technology' (Springer, 2004, 1st Edition).

[MERCURY]
Disponible:
http://groups.csail.mit.edu/sls/research/mercury.shtml [04 de febrero de 2011]

[MIMUS]
Pérez, G., Amores, G., Manchón, P., 2006. A multimodal architecture for home control by disabled users. In: Proc. of IEE/ACL Workshop on Spoken Language Technology (SLT). Palm Beach, Aruba, págs. 134-137.

[MySQL]
Disponible:
www.mysql.com [20 de mayo de 2011]

[NUANCE]
Disponible:
http://www.nuance.com [09 de febrero de 2011]

[OLGA]
Disponible:
http://www.nada.kth.se/~osu/olga/e_index.html [04 de febrero de 2011]

[OMVIAMEDIA]
Disponible:
http://www.intervoice.com [09 de febrero de 2011]

[OPENCALL]
Disponible:
http://www.hp.com [09 de febrero de 2011]

[OPENSPEECH]
Disponible:
http://www.spechworks.com [09 de febrero de 2011]

[OPENVXI]
Disponible:
http://www.speech.cs.cmu.edu/openvxi/ [09 de febrero de 2011]

[OPTIMTALK]
Disponible:
http://www.optimsys.cz [09 de febrero de 2011]

[PEGASUS]
Disponible:
http://groups.csail.mit.edu/sls/research/pegasus.shtml [04 de febrero de 2011]

[PUBLICVOICEXML]
Disponible:
http://www.publicvoicexml.org [09 de febrero de 2011]

[READSPEAKER]
Disponible:
http://www.readspeaker.com [20 de junio de 2011]

[RDF-SCHEMA]
"Resource Description Framework (RDF) Schema Specification 1.0 ", Dan Brickley y R.V. Guha. W3C Candidate Recommendation, March 2000.
Disponible:
http://www.w3.org/TR/2000/CR-rdf-schema-20000327/ [08 de febrero de 2011]

[SGRS]
"*Speech Recognition Grammar Specification Version 1.0* ". Hunt y McGlashan. W3C Proposed Recommendation, December 2003. Disponible: http://www.w3.org/TR/2003/PR-speech-grammar-20031218/ [08 de febrero de 2011]

[SSML]
" *Speech Synthesis Markup Language Version 1.0* ". Burnett, Walker y Hunt. W3C Candidate Recommendation, December 2003. Disponible: http://www.w3.org/TR/2003/CR-speech-synthesis-20031218/ [08 de febrero de 2011]

[TELLME]
Disponible:
http://www.tellme.com [09 de febrero de 2011]

[URI]
" *Uniform Resource Identifiers (URI): Generic Syntax* ", IETF RFC 2396, 1998.
Disponible:
http://www.ietf.org/rfc/rfc2396.txt [08 de febrero de 2011]

[VOCALIZA]
Vaquero, C., Saz, O., Lleida, E., Marcos, J., Canalís, C., 2006. VOCALIZA: An application for computer-aided speech therapy in spanish language. In: Proc. of IV Jornadas en Tecnología del Habla. Zaragoza, Spain, pp. 321-326-

[VOCALOCITY]
Disponible:
http://www.vocalocity.net [09 de febrero de 2011]

[VOICEGENIE]
Disponible:
http://www.voicegenie.com [09 de febrero de 2011]

[VOICEXML]
"Voice Extensible Markup Language (VoiceXML) Version 2.0". McGlashan et al. W3C Recommendation 16 March 2004. Disponible: http://www.w3.org/TR/voicexml20/ [08 de febrero de 2011]

[VOXGATEGAY]
Disponible: http://www.motorola.com [09 de febrero de 2011]

[VOXPILOT]
Disponible: http://www.voxpilot.com [09 de febrero de 2011]

[VOXWEBPC]
Disponible: http://www.voxweb.es/spanish/accesibilidad.html [10 de junio de 2011]

[XHTML+VOICE]
"XHTML+Voice Profile 1.0. ", Jonny Axelsson et al. W3C Recommendation, December 2001. Disponible: http://www.w3.org/TR/xhtml+voice/ [20 de mayo de 2011]

[XML]
" *Extensible Markup Language (XML) 1.0* ". Bray et al. W3C Recommendation 6 October 2000. Disponible: http://www.w3.org/TR/2000/REC-xml-20001006 [08 de febrero de 2011]