Procesamiento del Lenguaje Natural, num. 29 (2002), pp.197-203 recibido 25-05-2002; aceptado 02-07-2002

Integración automática de fuentes de conocimiento lingüístico en el desarrollo de sistemas de diálogo

David Pérez-Piñar López
Dto. de Teoría de la Señal y Comunicaciones
ETSI Telecomunicación – Univ. de Vigo
Lagoas/Marcosende – 36200 Vigo
dperez@gts.tsc.uvigo.es

Carmen García Mateo
Dto. de Teoría de la Señal y Comunicaciones
ETSI Telecomunicación – Univ. de Vigo
Lagoas/Marcosende – 36200 Vigo
carmen@gts.tsc.uvigo.es

Resumen: El desarrollo de sistemas de diálogo prácticos, orientados a aplicaciones específicas, requiere un esfuerzo de diseño muy elevado. La práctica en el desarrollo, manejo y refinamiento de sistemas concretos nos ha hecho ver que una de las razones de este problema es la interdependencia entre las fuentes de conocimiento lingüístico, que requieren coherencia en todos sus niveles. Para solucionar este problema sin afectar a la calidad del sistema, presentamos una arquitectura específica para sistemas de diálogo que permite la integración automática de todas las fuentes de conocimiento lingüístico a partir de la especificación formal de la aplicación. Esta arquitectura se caracteriza por su modularidad y su independencia completa de la aplicación. Presentamos también la aplicación de esta metodología a nuestro sistema de diálogo, así como los resultados preliminares obtenidos.
Palabras clave: Sistemas de diálogo; fuentes de conocimiento; VXML

Abstract: The development of practical dialogue systems, designed towards specific applications, requires a very high design effort. Our experience in the development, use and refinement of specific systems has made clear to us that one of the causes of this problem is the close relationship between the various lingüistic knowledge sources involved and the need of coherence between them. In order to solve this problem without affecting the system quality, we present a specific dialogue system arquitecture which integrates automatically all the lingüistic knowledge sources based on the formal application specification. Our proposed arquitecture is completely modular and application-independent. We present also the results obtained in the implementation of our system through the guidelines described.
Keywords: Dialogue systems; Knowledge sources; VXML

1 Introducción

En la actualidad, uno de los obstáculos más problemáticos en el desarrollo de sistemas de diálogo consiste en la especificación del mismo, desde las tareas que puede realizar hasta la interfaz con el usuario. Durante los últimos años se han desarrollado numerosos estudios dirigidos a dotar a estos sistemas de naturalidad en la interacción (Allen et al., 2000), y resumidamente puede extraerse la conclusión de que los sistemas que funcionan (dada la tecnología actual) son aquellos que implementan diálogos prácticos, es decir, diálogos centrados en temas específicos. Sin embargo, el esfuerzo dirigido a facilitar el desarrollo de los sistemas es proporcionalmente muy inferior.

Nuestra experiencia con sistemas de diálogo nos ha hecho ver la necesidad de mejorar los métodos de diseño, no sólo para obtener aplicaciones de calidad, sino también para facilitar la propia tarea del desarrollo (Bernsen et al., 1998).

En este artículo proponemos una arquitectura teórica que permita realizar de manera automática el diseño de un sistema de diálogo práctico. Con esto nos referimos a la capacidad para definir una aplicación completa, desde los con-

ISSN 1135-5948 ©2002 Sociedad Española para el Procesamiento del Lenguaje Natural

ceptos que maneja hasta el vocabulario, sin necesidad de vigilar manualmente la coherencia interna de las fuentes lingüísticas de conocimiento, exigida por la interacción que existe entre ellas.

1.1 Coordinación de fuentes lingüísticas de conocimiento

En la actualidad está ampliamente aceptado que los distintos elementos funcionales de un sistema de diálogo tienen entre sí interdependencias muy fuertes. De hecho, cualquier cambio en una de las fuentes de conocimiento de alguno de los módulos incide, en mayor o menor grado, en el resto de la aplicación.

Podemos clasificar en dos grupos la dependencia a la que nos estamos refiriendo: estática y dinámica. La dependencia estática afecta a elementos de la interfaz que no se modifican durante el uso de la aplicación, sino durante la fase de diseño o refinamiento. Un cambio, por ejemplo, en la gramática del analizador semántico debería reflejarse al menos en el vocabulario y en los modelos de lenguaje del reconocedor.

El caso de la dependencia dinámica se presenta en tiempo de ejecución, y es más problemático desde el punto de vista de la implementación. Hay ejemplos muy numerosos en este sentido; entre ellos puede citarse la inclusión dentro del vocabulario de nuevas palabras, como sucede en nuestro sistema de diálogo, que dispone de capacidad para la búsqueda de mensajes de correo por el remitente. Cuando se recibe un mensaje con un remitente nuevo, su nombre todavía no forma parte del vocabulario de la aplicación, y debe ser incluido.

Por otra parte, la evidencia de la necesidad de integrar todas estas fuentes de conocimiento se enfrenta al problema de la implementación práctica de esta integración. La arquitectura de la aplicación debe permitir estas interacciones, pero de hecho las limita en muchos casos por la complejidad que introducen en el sistema, o bien obliga a diseñar sistemas dependientes de la aplicación específica a la que van dirigidos (Hacioglu and Ward, 2001). La consecuencia directa es que la interdependencia existe, pero no está implementada o está embebida en el código, haciendo que el diseño del sistema sea una tarea tediosa y compleja.

1.2 Objetivos

Nuestro objetivo prioritario consiste en automatizar la integración de las fuentes de conocimiento para facilitar el diseño y el ajuste del sistema. Buscamos al mismo tiempo que esta integración automática no incida negativamente en las prestaciones del sistema. De hecho, pensamos que un sistema bien diseñado puede mejorar las prestaciones en cuanto a tasas de reconocimiento y aciertos en la realización de tareas, porque eliminará muchos errores que se escapan al diseñador, aun en los sistemas más simples.

Como resultado previo presentamos la arquitectura de nuestro sistema de diálogo TelCorreo, al que hemos añadido soporte VXML para la especificación formal del diálogo. Indicaremos cómo se integran en él los elementos externos y las fuentes de conocimiento, para terminar proponiendo un método de diseño nuevo que permite el desarrollo de múltiples aplicaciones bajo una misma arquitectura.

2 Arquitectura clásica de un sistema de diálogo

En la Figura 1 se muestra un esquema simplificado de un sistema de diálogo que sigue la arquitectura clásica. Desde el punto de vista de la implementación, esta arquitectura es modular, en el sentido de que los diferentes elementos que la componen pueden desarrollarse de manera independiente.

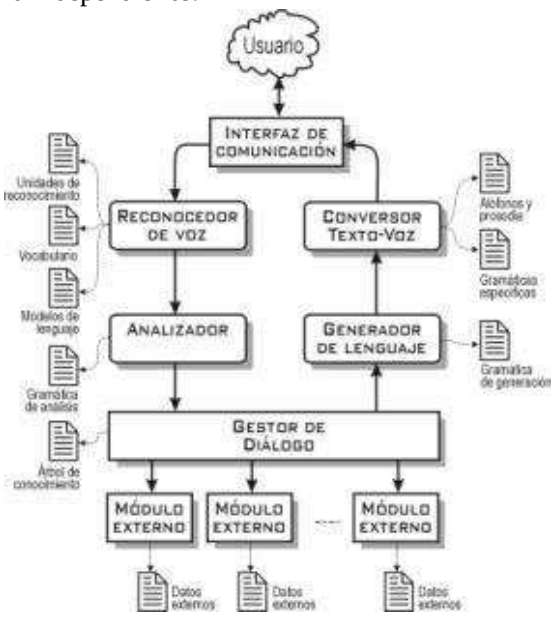

Figura 1: Arquitectura clásica

Las fuentes de conocimiento que se reflejan en el esquema son las básicas en cualquier sistema de diálogo, aunque su formato, su arquitectura o su número pueden variar según la implementación. La relación simplificada es la siguiente:

- Base de datos de unidades de reconocimiento. Contiene las unidades fonéticas que permiten al reconocedor elegir las palabras que ha dicho el usuario de entre las posibles.
- Vocabulario del reconocedor. Contiene las palabras que el reconocedor de voz es capaz de presentar al analizador.
- Modelos de lenguaje del reconocedor. Establecen una relación estadística de probabilidades entre las diversas palabras del vocabulario, para ayudar al reconocedor a seleccionar las que más convengan en cada momento.
- Gramática del analizador sintáctico-semántico. Establece unas reglas, basadas en la sintaxis, la semántica o en ambas, mediante las cuales es posible filtrar la salida del reconocedor y, al mismo tiempo, obtener una representación semántica de lo que el usuario ha dicho.
- Árbol de conocimiento. Contiene toda la información formalizada de la interacción con el usuario en ambos sentidos (entrada y salida), así como la relación entre esta formalización y las acciones que el sistema debe llevar a cabo.
- Bases de datos externas. Su presencia y sus características son muy variables, dependiendo de la aplicación concreta del sistema, pero en la práctica totalidad de los sistemas de diálogo se encuentran datos externos a los que hay que acceder, y que en muchos casos deben formar parte del conocimiento lingüístico del sistema.
- Gramática del generador de lenguaje. Establece las reglas de construcción de las frases con las que el sistema informa al usuario de sus acciones, partiendo de los conceptos formalizados de la aplicación.
- Gramáticas del conversor texto-voz. Aunque pocas veces se presentan como tales, en la mayoría de los casos existen dentro de los propios conversores gramáticas que expanden determinados elementos (fechas, números, direcciones de correo electrónico...)
- Base de datos de alófonos del conversor texto-voz. En este elemento se engloban no sólo las muestras sonoras que permiten construir la señal sintetizada de voz, sino también las reglas de formación de la prosodia, imprescindibles para dotar de naturalidad a la salida del sistema.

La relación entre estos elementos se hace más compleja a medida que se acerca al gestor de diálogo. Así, las bases de datos de unidades de reconocimiento y de alófonos dependen, fundamentalmente, del idioma, pero son prácticamente independientes de la aplicación y del resto de las fuentes de conocimiento. Sin embargo, los elementos de orden superior (vocabulario, modelos de lenguaje, gramáticas y árbol de conocimiento) están íntimamente relacionados, y van perdiendo esa relación con el idioma.

Así se ha puesto de manifiesto en numerosos estudios orientados a mejorar las prestaciones de los sistemas de diálogo, tanto desde el punto de vista de la entrada de información como de la salida (Constantinides et al., 1998). Sin embargo, esa relación entre las diferentes fuentes de conocimiento es difícil de implementar en sistemas que sigan esta arquitectura tradicional. La razón, desde nuestro punto de vista, está en la falta de un elemento aglutinador de la información, capaz de plasmar esas relaciones entre los diferentes elementos de conocimiento.

Efectivamente, cada módulo utiliza la información que le corresponde, y conoce el modo de comunicarse con los demás módulos para que éstos lleven a cabo su tarea. Pero no hay ningún módulo que establezca restricciones a esa comunicación, o lo que es lo mismo, no hay ningún elemento inteligente capaz de asociar entre sí los diversos elementos propios de la comunicación lingüística.

Un ejemplo puede aclarar más los conceptos desarrollados. Cuando se entabla un diálogo entre dos personas acerca de un tema particular, del que se supone que dominan lo suficiente, cada uno de los participantes en el diálogo maneja conceptos, y el esfuerzo de dialogar se centra en la búsqueda de conceptos. La expresión de esos conceptos mediante el lenguaje es, desde el punto de vista de la finalidad del diálogo, una tarea secundaria y aprendida, para la que no es necesario realizar ningún esfuerzo específico.

Es evidente que no hay ningún sistema capaz de formalizar en conceptos nuevos infor-

mación descrita en lenguaje natural (o en cualquier otro tipo de lenguaje). Por ello, el punto central de la cuestión está en encontrar una formalización de dichos conceptos, que debe ser previa al diálogo y que debe, al mismo tiempo, permitir la descripción de las relaciones entre esos conceptos y las formas naturales de expresarlos.

3 Arquitectura centralizada con módulo de gestión lingüística

Para resolver el problema de la coordinación de las fuentes de conocimiento, proponemos una nueva arquitectura que ya se está implantando en algunos sistemas, aunque con otros objetivos. La Figura 2 muestra un esquema de esta arquitectura.

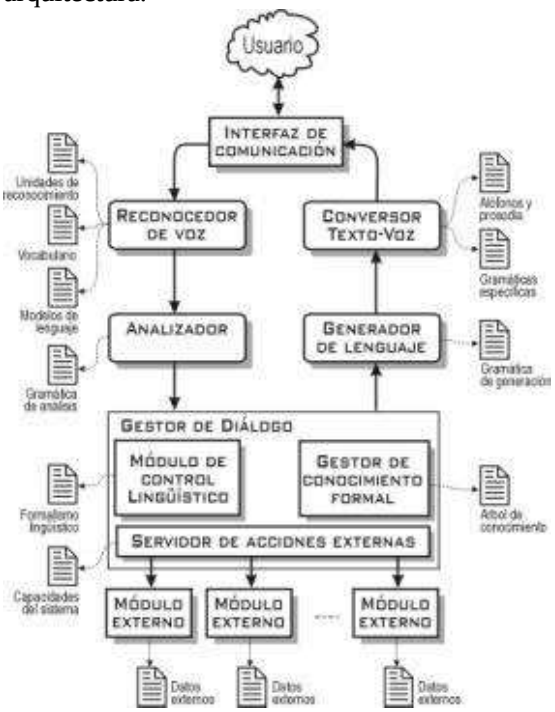

Figura 2: Arquitectura centralizada

Una primera cuestión que surge es la siguiente: ¿Quién manda en el sistema? No es fácil responder a esta pregunta, entre otras cosas porque la solución depende del punto de vista que se adopte.

Desde el punto de vista del diálogo, es evidente que el elemento central que lo dirige todo es el gestor de diálogo. Partiendo de la información contenida en el árbol de conocimiento, el gestor es capaz de determinar cuál debe ser el funcionamiento de los diversos módulos en cada instante del diálogo. Además, esa información es la que restringe la interacción lingüística con el usuario, de modo que además de dirigir el funcionamiento de los demás módulos especifica también sus fuentes de conocimiento, incluso aquellas que están más alejadas de la formalidad de la aplicación (como las bases de datos de alófonos, por ejemplo, que deben ajustarse al idioma de la interacción en el caso de sistemas multilingües).

Sin embargo, desde el punto de vista de la aplicación el gestor de diálogo se convierte en un servicio prestado al usuario. El control global de la aplicación no debe recaer entonces en este módulo, sino en otros que determinan cuándo dialogar y cuándo dejar de hacerlo. Es muy claro el caso de sistemas telefónicos, en los que el usuario determina con sus llamadas el momento de iniciar el diálogo.

Por otra parte, ese conocimiento formal de la aplicación que reside en el gestor de diálogo debe estar disponible no sólo durante su funcionamiento, sino también en la fase de desarrollo, en la que se determinan las fuentes de conocimiento de niveles inferiores. La relación existente entre los conceptos que maneja la aplicación y los formalismos lingüísticos que permiten dialogar sobre ellos puede derivarse de ese conocimiento formal superior, y por tanto desarrollar toda una aplicación a partir de una especificación única.

Esto impone una serie de condiciones en la arquitectura del sistema. Por un lado, todos los módulos deben exponer sus fuentes de conocimiento de forma que se puedan modificar desde fuera, y a ser posible dinámicamente. Por otro lado, es necesario un módulo independiente de generación de esas fuentes de conocimiento. Este nuevo módulo toma como base la información del árbol de conocimiento, y a partir de ella y de fuentes lingüísticas genéricas de conocimiento construye las fuentes de conocimiento de cada módulo.

El acceso dinámico a las fuentes de conocimiento de cualquier nivel es imprescindible en la mayoría de los casos (Constantinides et el., 1998), porque en muchas aplicaciones los datos externos (que deben formar parte del conocimiento lingüístico del sistema) cambian a lo largo de la ejecución. Por eso, el nuevo módulo propuesto debe ser capaz de acceder también a estos datos e incluirlos en los módulos cuando sea preciso en tiempo de ejecución.

De aquí se deriva la necesidad de estandarizar el acceso a los datos externos. En general,

puede especificarse una interfaz estándar entre el sistema y los módulos de acceso a dicha información, así como a cualquier recurso externo relacionado con la aplicación. El gestor de diálogo y el módulo de control lingüístico requieren el conocimiento de las capacidades del sistema, y por tanto esta interfaz debe proveer al sistema completo de un mecanismo en el que informe de los datos que maneja, del modo de manejarlos y de la posible presencia de errores.

4 Resultados preliminares: TelCorreo VXML como banco de pruebas

Como base de pruebas para llevar a cabo los objetivos presentados hasta aquí, se está implementando un sistema de diálogo simple en el que se cumplan todas las condiciones para permitir una definición automática de aplicaciones, siendo al mismo tiempo completamente independiente de la aplicación.

El sistema es una modificación de un servicio vocal telefónico de correo electrónico, denominado TelCorreo (Rodríguez-Liñares et al., 2000), en el que se han implementado las funciones básicas de un cliente de correo electrónico para el acceso por voz al buzón de correo de cada usuario. Entre las características propias del sistema de diálogo pueden destacarse la introducción de la verificación de usuarios por voz y el soporte multilingüe, tanto en reconocimiento como en síntesis de voz, que en la actualidad se extiende a tres idiomas (gallego, castellano y euskera) y que en breve se ampliará a cuatro con la inclusión del catalán. Por otra parte, también se incluye un detector automático de idiomas para la lectura de los cuerpos de los mensajes, que además de los ya expuestos contempla también el inglés y la posibilidad de mensajes mixtos (con texto en dos idiomas).

La modificación del sistema, siguiendo las líneas de desarrollo descritas anteriormente, se ha realizado buscando en primer lugar una estandarización de la especificación formal del diálogo. Dado el auge del lenguaje VXML y su cada vez mayor aceptación, se ha escogido este estándar, también por la relativa facilidad de manejo de las gramáticas que ofrece y por la posibilidad de desarrollar un intérprete completamente independiente, tanto de la plataforma hardware como de la aplicación a que se destina.

En la selección de VXML como formalismo de representación conceptual se han tenido en cuenta los siguientes aspectos:

- Su estandarización y aceptación global da acceso a múltiples herramientas de análisis, verificación y desarrollo de código VXML.
- La posibilidad de crear intérpretes de VXML completamente independientes de la aplicación y del hardware.
- La capacidad para soportar gramáticas, tanto de análisis como de síntesis, que permitan definir una interacción natural con el sistema.
- La posibilidad de desarrollar aplicaciones complejas, y en concreto la capacidad de VXML para formalizar interacciones de iniciativa mixta.
- La facilidad de integración con el intérprete de módulos externos de diversos tipos, que serán dependientes de la aplicación, sin obligar al intérprete a introducir dependencias.
- La posibilidad de utilizar el intérprete como servicio de un módulo de gestión superior.
- La capacidad para introducir módulos inteligentes de procesado del diálogo sin salir del estándar.

El desarrollo del sistema completo sigue la arquitectura centralizada de la Figura 2, partiendo de una base simple a partir de la que se desarrollarán paulatinamente los diferentes módulos. En la Figura 3 se muestra la arquitectura actualmente desarrollada.

Uno de los aspectos más complejos en la independencia de la aplicación consiste en la conexión entre los módulos externos y el gestor de diálogo. En nuestra implementación inicial, como puede verse en la Figura 3, el gestor de diálogo se limita al intérprete VXML, descargando toda la inteligencia en la especificación VXML del diálogo.

La independencia se ha implementado a través del elemento <OBJECT> especificado en el estándar VXML. Este elemento permite realizar tareas no especificadas en el lenguaje, a la vez que permite utilizar variables y contenidos de la aplicación.

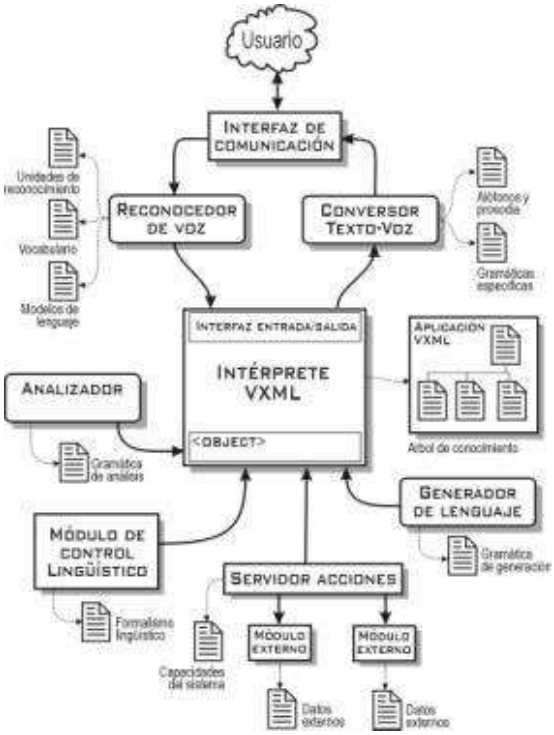

Figura 3: Implementación de la arquitectura propuesta mediante VXML

Su conexión con los módulos externos se realiza mediante sockets. El intérprete se conecta a un servidor de acciones genérico, que a su vez es el encargado de realizar las acciones externas. Para mantener la independencia de la aplicación, este servidor genérico externo se configura mediante un fichero que indica los módulos que están disponibles y el mecanismo de comunicación con cada uno, así como las funciones que ofrecen.

De esta manera, el servidor de acciones es también independiente de la aplicación, y al mismo tiempo su configuración se hace accesible al módulo de control lingüístico, permitiendo así un acceso centralizado para el diseño de la aplicación, y sobre todo para el mantenimiento de las dependencias dinámicas.

La interfaz entre el intérprete y los demás módulos se ha desarrollado siguiendo el mismo esquema de comunicación mediante sockets. Esto permite, por ejemplo, sustituir el reconocedor de voz o el conversor texto-voz por herramientas de línea de comandos sin modificar el intérprete, lo cual facilita mucho las tareas de depuración y desarrollo de las aplicaciones. El formato de los mensajes entre los distintos módulos y el intérprete se ha diseñado de forma que mantenga la independencia de la aplicación, y al mismo tiempo no reduzca las prestaciones del sistema.

El reconocedor de voz del sistema es de desarrollo propio (Cardenal, Diéguez y García, 2002), de habla continua y grandes vocabularios, y tiene la arquitectura necesaria para permitir el acceso a sus fuentes de conocimiento. Del mismo modo, el sistema también incluye un conversor texto-voz de desarrollo propio basado en corpus, denominado Cotovía (Rodríguez et al., 2002), en el que se están desarrollando también módulos específicos para presentar al módulo de control lingüístico sus fuentes de conocimiento.

En paralelo al desarrollo de esta arquitectura, se está implementando un editor VXML orientado a facilitar el diseño de las aplicaciones. El editor está integrado en la misma arquitectura del sistema, y utiliza el módulo de control lingüístico para generar todas las fuentes de información y permitir la depuración en todos los niveles lingüísticos de conocimiento.

5 *Conclusiones y líneas de desarrollo*

Los resultados preliminares de la implementación de la arquitectura propuesta son positivos. Por un lado, se ha establecido un mecanismo experimental de integración semiautomática entre la gramática, el modelo de lenguaje y el vocabulario. En este primer prototipo, el modelo de lenguaje, que es fijo para todos los ámbitos del diálogo, se obtiene a partir de un modelo de lenguaje general, extraído de bases de datos de noticias, reforzado con modelos de lenguaje particulares de la aplicación, creados automáticamente a partir de la gramática. La calidad del sistema en cuanto al reconocimiento de voz no se ha reducido, mientras que se ha conseguido unificar el desarrollo de tres fuentes de conocimiento básicas para la aplicación.

El desarrollo del sistema se orienta ahora hacia la especificación de la integración desde un nivel superior, constituido por el formalismo del diálogo especificado en el lenguaje VXML. Esta nueva especificación supondrá una revisión del mecanismo automático ya implementado, pero permitirá la definición de la aplicación completa desde el punto de vista conceptual. Al mismo, tiempo, se iniciará el estudio de la posibilidad de integrar en este entorno los mecanismos de recuperación de errores en el diálogo.

Bibliografía

Allen, J., D. Byron, M. Dzikovska, G. Ferguson, L. Galescu, y A. Stent. 2000. An Architecture for a Generic Dialogue Shell. *Natural Language Engineering*. To appear.

Bernsen, N., Dybkjaer, H. and Dybkjaer, L. Designing Interactive Speech Systems. 1998. Springer-Verlag, ISBN 3-540-76048-2.

Cardenal, A., Diéguez, J., García-Mateo, C., Fast LM Look-Ahead for Large Vocabulary Continuous Speech Recognition using Perfect Hashing. 2002. *Proceedings of the IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP 2002)*, pág. 705-708. Orlando, USA.

Constantinides, P., Hansma, S., Tchou, C. and Rudnicky, A. A schema-based approach to dialog control. 1998. *Proceedings of ICSLP*, Paper 637.

Hacioglu, K. and Ward, W. Dialog-Context Dependent Language Modeling Using N-Grams and Stochastic Context-Free Grammars. 2001. *Proceedings of ICASSP*, Salt Lake City.

Pellom, B., Ward, W., Pradhan, S., The CU Communicator: An Architecture for Dialogue Systems. 2000. *International Conference on Spoken Language Processing (ICSLP)*. Beijing, China.

Rodríguez, E., Campillo, F., Fernández, E., Méndez, F., Sistema de Conversión Texto-Voz en Lengua Gallega Basado en la Selección Combinada de Unidades Acústicas y Prosódicas. 2002. *XVIII Congreso de la Sociedad Española para el Procesado del Lenguaje Natural (SEPLN)*. Valladolid, España.

Rodríguez-Liñares, L., Cardenal, A., García-Mateo, C., Perez-Piñar, D., Rodríguez, E., Fernández, X., TelCorreo: A Bilingual E-mail Client over the Telephone. 2000. *Proceedings of the Third International Workshop on Text, Speech and Dialogue (TSD 2000), pág. 381-386*. Brno, Czech Republic.

Rudnicky, A. and Xu, W. An agenda-based dialog management architecture for spoken language systems. 1999. *IEEE Automatic Speech Recognition and Understanding Workshop*, pág. I-337.

Xu, W. and Rudnicky, A. Language modeling for dialog system? 2000. *Proceedings of ICSLP*. Paper B1-06.