



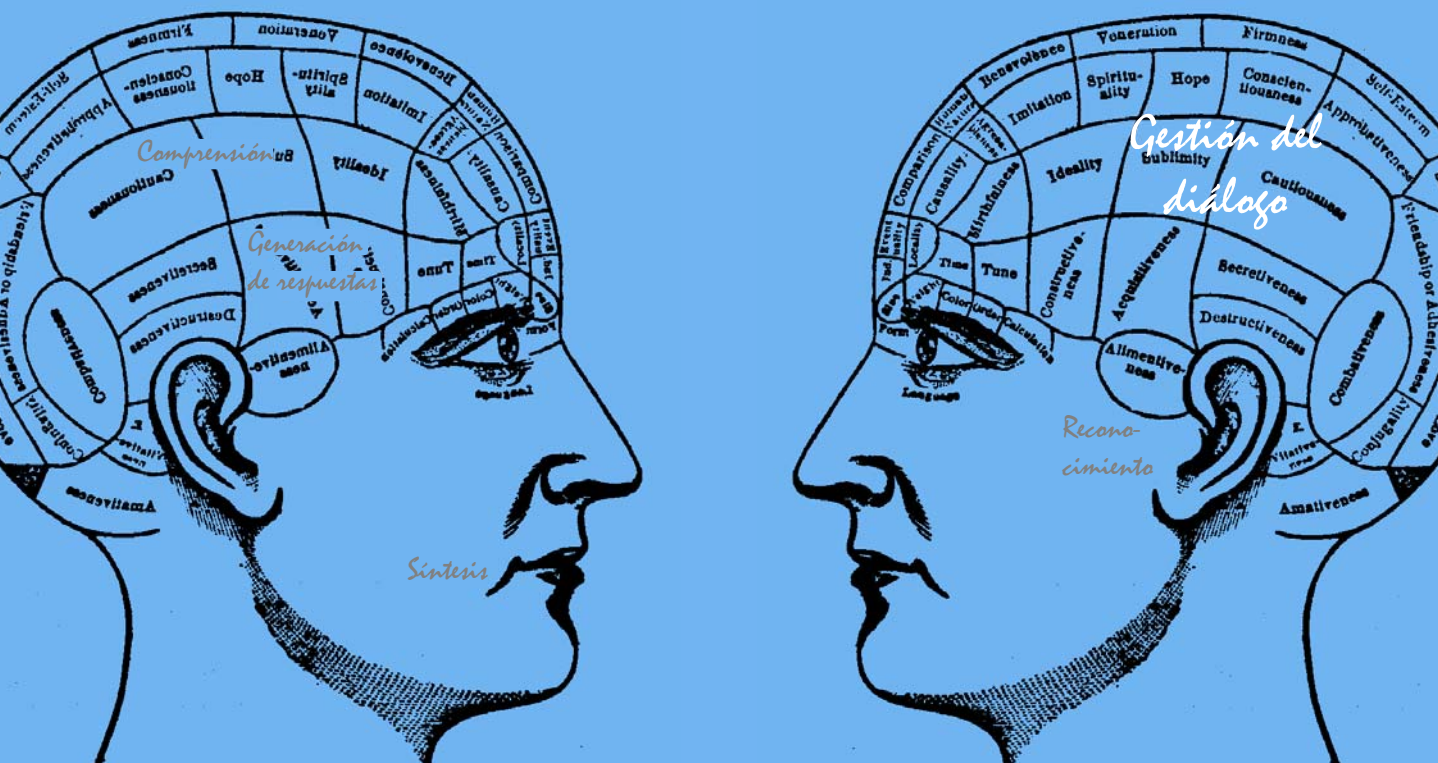
DSIC

Desarrollo y Evaluación de Diferentes Metodologías para la Gestión Automática del Diálogo

David Griol Barres

Dr. D. Lluís F. Hurtado Oliver

Dra. Dña. Encarna Segarra Soriano



Departament de Sistemes Informàtics i Computació
Universitat Politècnica de València
Camí de Vera s/n. 46022 València (Spain)



DSIC

Desarrollo y evaluación de diferentes metodologías para la gestión automática del diálogo

D. David Griol Barres

Tesis Doctoral
dirigida por los doctores:

Dr. D. Lluís Felip Hurtado Oliver
Dra. Dña. Encarna Segarra Soriano

València, noviembre 2007

Y por eso vivo el día
día simple, día claro.
Vivo al menos sin temores,
sin el miedo de gozar.

Cada pueblo, cada puente,
cada cruce me ha enseñado
que con hoy es suficiente
y mañana es demasiado.

ANTONIO VEGA (Estaciones)

Agraïments

En primer lloc, voldria agrair l'ajuda dels meus directors de tesi Lluís i Encarna. Incloc en el mateix sentiment a Emilio. He d'agrair-vos el suport i l'amistat que m'heu donat des del principi, així com la vostra dedicació per tal de dur a bon port el treball mostrat en esta tesi i la possibilitat que m'heu donat de fer-la. Guarde molts records i bons moments, que de segur mantindré sempre al meu cap.

Vull agrair també l'ajuda de la resta de gent del grup de Reconeixement de Formes i Intel·ligència Artificial de la UPV. Voldria recordar-me de la gent amb qui he tingut la immensa sort de conviure diàriament durant aquest temps i que sempre heu tingut una mà per tal d'ajudar-me, un ànim per continuar i una estona per tal de parlar i prendre un cafè. Vaja el meu agraïment a tots vosaltres, heu fet que tot aquest treball haja valgut la pena. També voldria aprofitar per donar ànims a tots aquells doctorands que hagen arribat per una raó o altra a aquesta tesi.

Vull agrair també el treball de tota la gent que participà en el projecte DIHANA, tant a la Universitat de Saragossa, la Universitat del País Basc i la Universitat Politècnica de València, sense aquest treball la tesi no haguera sigut possible. Especialment voldria agrair a José Miguel Benedí l'oportunitat de treballar en el projecte.

A Paco, Sergio, Ramón i Salva per tota la vostra ajuda, idees i sobretot per la vostra amistat.

Per últim, però ocupant-lo tot, aquesta tesi va dedicada als meus pares, als meus germans i a Zoraida, amb els quals he trobat la il·lusió necessària per tal d'acabar-la. Vaja tota la meua estima per a Xuxi, no t'oblidaré mai xicotiua.

Resumen

El objetivo principal de la tesis que se presenta es el estudio y desarrollo de diferentes metodologías para la gestión del diálogo en sistemas de diálogo hablado. El principal reto planteado en la tesis reside en el desarrollo de metodologías puramente estadísticas para la gestión del diálogo, basadas en el aprendizaje de un modelo a partir de un corpus de diálogos etiquetados. En este campo, se presentan diferentes aproximaciones para realizar la gestión, la mejora del modelo estadístico y la evaluación del sistema del diálogo.

Para la implementación práctica de estas metodologías, en el ámbito de una tarea específica, ha sido necesaria la adquisición y etiquetado de un corpus de diálogos. El hecho de disponer de un gran corpus de diálogos ha facilitado el aprendizaje y evaluación del modelo de gestión desarrollado. Así mismo, se ha implementado un sistema de diálogo completo, que permite evaluar el funcionamiento práctico de las metodologías de gestión en condiciones reales de uso.

Para evaluar las técnicas de gestión del diálogo se proponen diferentes aproximaciones: la evaluación mediante usuarios reales; la evaluación con el corpus adquirido, en el cual se han definido unas particiones de entrenamiento y prueba; y la utilización de técnicas de simulación de usuarios. El simulador de usuario desarrollado permite modelizar de forma estadística el proceso completo del diálogo. En la aproximación que se presenta, tanto la obtención de la respuesta del sistema como la generación del turno de usuario se modelizan como un problema de clasificación, para el que se codifica como entrada un conjunto de variables que representan el estado actual del diálogo y como resultado de la clasificación se obtienen las probabilidades de seleccionar cada una de las respuestas (secuencia de actos de diálogo) definidas respectivamente para el usuario y el sistema.

A partir de los diálogos generados mediante el uso de este módulo de simulación se ha ampliado y mejorado el corpus adquirido inicialmente. Además se presentan diferentes técnicas para la generación automática de diálogos, que facilitan la obtención automática de un corpus etiquetado de diálogos y el posterior aprendizaje de un gestor de diálogo.

Los trabajos desarrollados se engloban en el marco del proyecto DIHANA, cuyo principal objetivo fue el desarrollo de un sistema de diálogo para el acceso a un sistema de diálogo mediante el habla espontánea. La tarea definida para el proyecto fue el acceso vocal a un sistema que proporciona información sobre trayectos en tren de recorrido nacional.

En último lugar, las metodologías propuestas en DIHANA para la gestión del diálogo se han adaptado para desarrollar un gestor de diálogo en el ámbito del proyecto EDE-CÁN. Se describe la adaptación realizada y la evaluación de un gestor desarrollado para un sistema de diálogo que facilita la reserva de instalaciones deportivas.

Adicionalmente, se presentan diferentes metodologías basadas en reglas para la gestión del diálogo, así como distintas aproximaciones para el desarrollo de generadores de respuestas en lenguaje natural.

Resum

L'objectiu principal de la tesi que es presenta és l'estudi i desenvolupament de diferents metodologies per a la gestió del diàleg en sistemes de diàleg parlat. El principal repte plantejat amb la tesi consisteix en el desenvolupament de metodologies estadístiques per a la gestió del diàleg, basades en el aprenentatge d'un model a partir d'un corpus de diàlegs etiquetats. En aquest camp es presenten diferents aproximacions per realitzar la gestió, la millora del model estadístic i l'avaluació del sistema de diàleg.

Per a la implementació pràctica d'aquestes metodologies, en l'àmbit d'una tasca específica, ha estat necessari l'adquisició i etiquetat d'un corpus de diàlegs. El fet de disposar d'un gran corpus de diàlegs ha facilitat l'aprenentatge i avaluació del model de gestió desenvolupat. Així mateix, s'ha implementat un sistema de diàleg complet per comprovar el funcionament de les metodologies de gestió proposades en condicions real d'ús.

Per tal d'avaluar les tècniques de gestió de diàleg es proposen diferents aproximacions: l'avaluació mitjançant usuaris reals; la utilització de tècniques de validació creuada mitjançant la definició de particions d'entrenament i d'avaluació en el corpus adquirit; i la utilització de tècniques de simulació d'usuaris. El simulador d'usuari desenvolupat permet modelitzar de forma estadística el procés complet del diàleg. En l'aproximació que es presenta, tant l'obtenció de la resposta del sistema com la generació del torn d'usuari es modelitzen com un problema de classificació, en el que es codifica com a entrada un conjunt de variables que representen l'estat actual del diàleg i com a resultat de la classificació s'obtenen les probabilitats de seleccionar cadascuna de les respostes (seqüència d'actes de diàleg) definides respectivament per a l'usuari i el sistema.

A partir dels diàlegs generats mitjançant l'ús d'aquest mòdul de simulació s'ha ampliat i millorat el corpus adquirit inicialment. A més a més, es presenten diferents tècniques per a la generació automàtica de diàlegs, que faciliten l'obtenció automàtica d'un corpus etiquetat de diàlegs i el posterior aprenentatge d'un gestor de diàleg.

Els treballs realitzats s'engloben en el marc del projecte DIHANA, que tingué com a principal objectiu el desenvolupament d'un sistema de diàleg per accedir a un sistema d'informació mitjançant la parla espontànea. La tasca definida per al projecte va ser l'accés vocal a un sistema que proporciona informació sobre trajectes en tren de recorregut nacional.

En últim lloc, les metodologies definides en DIHANA per fer la gestió del diàleg s'han adaptat per tal d'elaborar un gestor de diàleg per al projecte EDECÁN. Es descriu l'adaptació realitzada i l'avaluació d'un gestor desenvolupat per a un sistema de diàleg que facilita la reserva d'instal·lacions esportives.

Adicionalment, es descriuen diferents metodologies basades en regles per a la gestió del diàleg, així com distintes aproximacions per al desenvolupament de generadors de respostes en llenguatge natural.

Abstract

The main objective defined for this thesis is the study and development of different methodologies to carry out dialog management in spoken dialog systems. The main challenge set in the thesis resides in the development of statistical methodologies for dialog management, based on learning a model from a corpus of labeled dialogs. In this field, the thesis presents different approaches to managing the dialog, the improvement of the statistical model and the evaluation of the dialog system.

For the practical implementation of these methodologies, in the framework of a specific task, the acquisition and labeling of a dialog corpus was necessary. This dialog corpus has facilitated the learning and evaluation of the model developed for dialog management. In addition, a complete dialog system has been developed to evaluate the practical operation of the proposed methodologies for dialog management in real use conditions.

Different approaches are proposed to evaluate the techniques for dialog management: the evaluation by means of real users; the definition of training and test partitions using the acquired corpus; and the evaluation by means of user simulation techniques. The user simulator that has been developed allows the statistical modelization of the complete process of the dialog. In this approach, the processes for obtaining the system answer and generating the new user turn are modeled as a classification problem, in which the input of the classifier is a set of variables that represent the current state of the dialog. The result of the classification can be seen as the probability of selecting each one of the answers (sequence of dialog acts) respectively defined for the user and the system.

The initial corpus has been extended and improved by means of the dialogs generated using this simulation module. In addition, different techniques for the automatic generation of dialogs are described. These techniques facilitate the automatic acquisition of a labeled dialog corpus and the later learning of a dialog manager.

These works have been developed within the framework of the DIHANA project, whose main objective was the development of a dialog system to access an information system using spontaneous speech. The task defined for the project was the oral access to a railway information system.

Finally, the methodologies proposed for the dialog management in the DIHANA project have been adapted to develop a dialog manager for the EDECAN project. The adaptation and evaluation of a dialog manager developed for a dialog system that facilitates the booking of sport facilities is described.

Additionally, different rule-based methodologies for dialog management are presented, as well as different approaches for the development of natural language answer generators.

Índice general

1. Introducción	27
1.1. Introducción. Sistemas de diálogo hablado	27
1.1.1. Reconocimiento automático del habla	32
1.1.2. Comprensión del habla	34
1.1.2.1. Medidas de confianza	34
1.1.3. Gestión de diálogo	35
1.2. Objetivos y líneas de investigación tratadas en la tesis	36
1.3. Estructura de la tesis	37
2. Estado del arte	41
2.1. Sistemas de diálogo hablado	41
2.2. Gestión del diálogo	72
2.2.1. Resumen de metodologías de gestión relevantes	75
2.2.2. Teoría de los Estados de la Información	81
2.2.3. Metodologías estadísticas. Aprendizaje Reforzado	84
2.2.3.1. Modelado del diálogo mediante redes bayesianas	94
2.3. Evaluación de sistemas de diálogo	96
2.4. Simulación de usuarios	104
2.5. Adaptación al usuario y/o a la tarea	112
3. La tarea DIHANA	119
3.1. Introducción	119
3.2. Objetivos del proyecto DIHANA	120
3.3. Adquisición del corpus DIHANA	122
3.3.1. Tipos de escenarios definidos	122
3.3.2. La plataforma de adquisición de diálogos	124
3.3.3. Fases de la adquisición	125
3.3.4. La estrategia del Mago de Oz	126
3.3.5. Ficheros almacenados durante la adquisición	127
3.3.6. Evaluación recibida de los usuarios	128
3.3.7. Características del corpus DIHANA	130
3.4. Etiquetado del corpus DIHANA	130
3.4.1. Etiquetado de los turnos de usuario: representación semántica de la tarea	130
3.4.2. Etiquetado de los turnos de sistema	134

7.4.	Evaluación del gestor de diálogo estadístico basado en redes neuronales . . .	196
7.4.1.	Evaluación de la influencia de la talla del corpus	197
7.4.2.	Evaluación de la influencia del sexo	198
7.4.3.	Evaluación de la procedencia de los diálogos	198
7.5.	Evaluación utilizando la voz y usuarios reales	199
8.	Simulación de usuarios	201
8.1.	Introducción	201
8.2.	Simulador de usuarios estadístico	202
8.2.1.	Codificación definida para el clasificador MLP	204
8.2.2.	Implementación práctica del simulador estadístico	206
8.2.3.	Evaluación del simulador	210
8.3.	Definición de una técnica para la adquisición automática de diálogos	213
8.3.1.	Adquisición de un corpus de diálogos	214
8.3.2.	Ejemplos de diálogos adquiridos	215
8.3.3.	Evaluación del corpus simulado	217
8.4.	Experimentación con las metodologías descritas	218
8.4.1.	Modelo estadístico tanto en el simulador de usuario como en el gestor de diálogo	218
8.4.2.	Selección aleatoria de respuestas en el simulador de usuarios y modelo estadístico en el gestor de diálogo	220
8.4.3.	Modelo estadístico en el simulador de usuarios y selección aleatoria de respuestas en el gestor de diálogo	221
9.	Adaptación del gestor del diálogo al usuario y a nuevas tareas	223
9.1.	Introducción	223
9.2.	El proyecto EDECÁN	223
9.3.	La tarea EDECÁN	226
9.4.	Representación semántica de la tarea	227
9.4.1.	Etiquetado de los turnos de usuario	228
9.4.2.	Etiquetado de los turnos de sistema	229
9.5.	Representación de las fuentes de información	231
9.6.	Gestor de diálogo desarrollado	232
10.	Conclusiones	235
A.	Adquisición Corpus DIHANA	241
A.1.	Ejemplo de ficha definida para la adquisición del corpus DIHANA	241
A.2.	Ficheros almacenados durante la adquisición de cada diálogo	245
A.2.1.	Fichero .fra.xml. Frases acústicamente balanceadas	245
A.2.2.	Fichero .rec.xml. Módulo de reconocimiento automático del habla	246
A.2.3.	Fichero .com.xml. Módulo de comprensión semántica	247
A.2.4.	Fichero .dia.xml. Módulo gestor de diálogo	248
A.2.5.	Fichero .que.xml. Consultas a la base de datos	250
A.2.6.	Fichero .piz.xml. Registro del diálogo	251
A.3.	Resultados de la encuestas	253
A.3.1.	Encuesta que rellenaron los solicitantes	253

A.3.2. Estadísticas generales de las encuestas y evaluación de los escenarios	255
B. Sistema de diálogo DIHANA	261
B.1. Formato de los paquetes XML definidos	261
B.1.1. Módulo de entrada (RAH)	261
B.1.2. Módulo de comprensión (CH)	261
B.1.3. Gestor de diálogo (GD)	262
B.1.4. Módulo generador de respuestas (GR)	263
B.1.5. Módulo de síntesis de voz (CTV)	264
B.1.6. Módulo de base de datos (BD)	264
B.2. Categorías definidas en el corpus DIHANA	264
B.2.1. Ciudades	264
B.2.2. Servicios	265
B.2.3. Tipo-viaje	265
B.2.4. Tipo-tren	265
B.2.5. Precio	265
B.2.6. Número-relativo-orden	265
B.2.7. Dirección	265
B.2.8. Clase billete	266
B.3. Gestión del diálogo: Implementación en LUA de las redes neuronales	266
C. Generación de respuestas	269
C.1. Plantillas de respuestas definidas para la adquisición del corpus DIHANA	269
C.1.1. No entendido	269
C.1.2. No válido	269
C.1.3. Completar (confirmación/pregunta de concepto)	269
C.1.4. Confirmación explícita (de concepto y atributos)	270
C.1.5. Confirmación implícita	271
C.1.5.1. Duración	271
C.1.5.2. Horarios	274
C.1.5.3. Precio	276
C.1.5.4. Servicios	283
C.1.5.5. Tipo de tren	283
C.1.6. Confirmación mixta Seguro	284
C.1.6.1. Concepto Horario	284
C.1.6.2. Concepto Precio	284
C.1.6.3. Concepto Duración	284
C.1.6.4. Frame Tipo de tren	285
C.1.6.5. Concepto Servicios	285
C.1.6.6. Segunda parte de la confirmación mixta	285
C.1.7. Confirmación mixta Concepto Inseguro	285
C.1.7.1. Primera parte de la confirmación	285
C.1.7.2. Segunda parte de la confirmación	285
C.1.8. Concepto Seguro Hora, Precio, Duración, Tipo de tren o Servicios	286
C.1.8.1. Primera parte de la confirmación	286
C.1.8.2. Segunda parte de la confirmación	286

C.1.9. Nueva Consulta	286
C.2. Plantillas de respuestas definidas tras el etiquetado del corpus.	287
C.2.1. Preguntas	287
C.2.2. Confirmaciones explícitas	287
C.2.2.1. Conceptos	287
C.2.2.2. Atributos	287
C.2.3. Confirmaciones mixtas	288
C.2.4. Respuestas a consultas a la base de datos	289
C.2.4.1. Respuesta-Hora_salida	289
C.2.4.2. Confirmación,Respuesta-[Hora_salida],Hora_salida	290
C.2.4.3. Confirmación,Respuesta-Hora_salida,Hora_salida	290
C.2.4.4. Respuesta-Hora_llegada	290
C.2.4.5. Confirmación,Respuesta-[Hora_llegada],Hora_llegada	291
C.2.4.6. Respuesta-Precio	291
C.2.4.7. Confirmación,Respuesta-Precio,Precio	297
C.2.4.8. Confirmación,Respuesta-[Precio],Precio	298
C.2.4.9. Respuesta-Tipo_tren	298
C.2.4.10. Confirmación,Respuesta-[Tipo_tren],Tipo_tren	298
C.2.4.11. Confirmación,Respuesta-Tipo_tren,Tipo_tren	299
C.2.4.12. Respuesta-Tiempo_recorrido	299
C.2.4.13. Respuesta-Servicio	301
C.2.4.14. Confirmación,Respuesta-[Servicio],Servicio	301
C.2.5. Otras	301
D. Desarrollo de sistemas de diálogo mediante VoiceXML	303
D.1. Introducción a VoiceXML	303
D.1.1. Orígenes de VoiceXML	303
D.1.2. Estructura y funcionamiento	304
D.1.3. Conceptos básicos	306
D.1.3.1. Diálogos y subdiálogos	306
D.2. Constructores de diálogo	307
D.2.1. Formularios	307
D.2.2. Menús	308
D.3. Modelado del lenguaje en VoiceXML	308
D.3.1. Gramáticas	308
D.3.1.1. Gramáticas para el reconocimiento de voz	308
D.3.1.2. Ámbito de las gramáticas	309
D.3.1.3. Activación de las gramáticas	309
D.4. Recursos disponibles	310
D.4.1. IBM Voice Toolkit para WebSphere Studio	311
D.4.1.1. Instalación del Voice Toolkit	311
D.4.1.2. Reconocimiento de voz	312
D.5. Aplicación práctica: Desarrollo de una centralita en castellano	313
D.6. Implementaciones de las especificaciones de interfaces vocales del W3C	315
D.6.1. VoiceXML 2.0 y 2.1	315
D.6.2. Especificaciones para las gramáticas de reconocimiento de voz	317

D.6.3. Interpretación semántica de gramáticas de reconocimiento de voz .	318
D.6.4. CCXML (Call Control XML)	318

Bibliografía	319
---------------------	------------

Índice de tablas

3.1. Estadísticas de las preguntas formuladas a los informantes que adquirieron el corpus DIHANA	129
3.2. Estadísticas de los escenarios formulados a los informantes que adquirieron el corpus DIHANA	129
3.3. Principales características del corpus DIHANA	130
3.4. Principales características del corpus BASURDE	130
3.5. Conceptos dependientes de la tarea	131
3.6. Conceptos no dependientes de la tarea	131
3.7. Atributos definidos para la representación semántica en la tarea DIHANA .	132
6.1. Reducción del espacio de estados tras la introducción del <i>DR</i> en DIHANA	171
6.2. Ejemplos de funciones de activación de una red neuronal	181
7.1. Resultados de la evaluación realizada para los modelos de gestión basados en reglas	192
7.2. Principales características de las particiones definidas para la evaluación de los gestores estadísticos	193
7.3. Resultados de la evaluación del tratamiento de las situaciones no vistas . .	195
7.4. Evaluación de la medida $\%error$ para la función distancia y el clasificador MLP	196
7.5. Evaluación del modelo estadístico basado en el MLP	197
7.6. Comparativa entre la utilización de la función Distancia y el MLP	197
7.7. Evaluación del modelo estadístico frente a diversas tallas del corpus de entrenamiento	197
7.8. Evaluación del modelo estadístico teniendo en cuenta el sexo de los usuarios	198
7.9. Evaluación del modelo estadístico teniendo en cuenta la procedencia de los diálogos	198
7.10. Evaluación del gestor de diálogo estadístico mediante pruebas con voz . . .	199
8.1. Reducción del espacio de estados tras la introducción del <i>UR</i> en DIHANA	204
8.2. Agrupación de escenarios definida para la evaluación del corpus simulado .	211
8.3. Evaluación del corpus adquirido mediante el simulador de usuarios	211
8.4. Estadísticas de la adquisición del corpus mediante la técnica de simulación de diálogos	215
8.5. Resultado de la evaluación de los gestores de diálogo M1 y M2	217
8.6. Modelo estadístico en el simulador de usuario y en el gestor de diálogo. Modelo general	219

8.7. Modelo estadístico en el simulador de usuario y en el gestor de diálogo. Modelos específicos	220
8.8. Selección aleatoria de respuestas en el simulador de usuarios y modelo estadístico en el gestor de diálogo. Modelo general	220
8.9. Selección aleatoria de respuestas en el simulador de usuarios y modelo estadístico en el gestor de diálogo. Modelos específicos	221
8.10. Modelo estadístico en el simulador de usuarios y selección aleatoria de respuestas en el gestor de diálogo. Modelo general	221
8.11. Modelo estadístico en el simulador de usuarios y selección aleatoria de respuestas en el gestor de diálogo. Modelos específicos	222
9.1. Resultados de la evaluación del gestor de diálogo desarrollado para el proyecto EDECÁN	233

Índice de figuras

1.1.	Diagrama de acciones de un sistema de diálogo	28
1.2.	Arquitectura modular de un sistema de diálogo hablado	30
2.1.	Arquitectura definida en el Galaxy Communicator	44
2.2.	Arquitectura Olympus para el desarrollo de sistemas de diálogo	49
2.3.	Arquitectura definida en la plataforma ATLAS	53
2.4.	Arquitectura definida para la plataforma JASPIS	54
2.5.	Arquitectura definida en el proyecto TRIPS	56
2.6.	Modelo de desarrollo del paradigma PARADISE	99
3.1.	Ejemplo de escenario tipo A	123
3.2.	Ejemplo de escenario tipo B	123
3.3.	Ejemplo de escenario tipo C	123
3.4.	Ejemplo de escenario abierto (izquierda) y cerrado (derecha)	124
3.5.	Plataforma de adquisición de diálogos (interfaz generado para el Mago de Oz)	125
3.6.	Diagrama que muestra los estados definidos para la estrategia del Mago de Oz en DIHANA	127
3.7.	Ejemplo del etiquetado definido para el corpus DIHANA	137
4.1.	Arquitectura definida para el sistema de diálogo DIHANA	140
4.2.	Ejemplo de paquetes XML definidos para la comunicación en el sistema de diálogo DIHANA	142
4.3.	Esquema del proceso de comprensión desarrollado para el proyecto BA-SURDE	144
4.4.	Ejemplo de traducción a lenguaje semántico (1ª fase de comprensión)	144
4.5.	Ejemplo de traducción a frames (2ª fase de comprensión)	146
4.6.	Ejemplo de un árbol de respuestas de petición de atributos	150
4.7.	Ejemplo de un árbol de respuestas de confirmación	151
4.8.	Generación de la sentencia SQL	153
4.9.	Modulo gestor de las comunicaciones del sistema de diálogo DIHANA	154
5.1.	Ejemplos de las técnicas de simplificación semántica definidas	159
5.2.	Atributos obligatorios definidos para cada uno de los conceptos en DIHANA	160
5.3.	Inicialización del diálogo	161
5.4.	Solicitud del tipo de consulta	161
5.5.	Solicitud de los atributos obligatorios	161

5.6.	Confirmación explícita basada en las medidas de confianza	162
5.7.	Resultado de la consulta a la base de datos	162
5.8.	Finalización del diálogo	162
5.9.	Inicialización del diálogo	163
5.10.	Solicitud del tipo de consulta	164
5.11.	Confirmación de los atributos	164
5.12.	Solicitud de los atributos obligatorios	164
5.13.	Confirmación de los atributos	165
5.14.	Resultado de la consulta a la base de datos	165
5.15.	Finalización del diálogo	165
6.1.	Estructura del registro del diálogo (<i>DR</i>) definido para la tarea DIHANA .	170
6.2.	Matriz de pesos para el cálculo de la distancia entre secuencias	173
6.3.	Ejemplo de cálculo de las secuencias con menor distancia	173
6.4.	Ejemplo de diálogo generado con el gestor estadístico y la función distancia	174
6.5.	Ejemplo de la función <i>g</i> definida en el MGGI	180
6.6.	Red neuronal artificial	181
6.7.	Esquema gráfico de un perceptrón (izquierda) y de un MLP (derecha) . . .	182
6.8.	Esquema gráfico de la regla delta (izquierda) y de los términos involucrados en la regla delta generalizada (derecha)	183
6.9.	Algoritmo de Backpropagation	184
6.10.	Gestor estadístico desarrollado con la aplicación de las técnicas de trata- miento de las situaciones no vistas	186
6.11.	Ejemplo de diálogo obtenido con el gestor basado en el clasificador Naive Bayes	187
6.12.	Esquema del gestor de diálogo desarrollado para el proyecto DIHANA . . .	188
6.13.	Ejemplos de diálogos obtenidos con el gestor basado en la utilización del MLP	190
7.1.	Medidas definidas para la evaluación de la metodología de gestión estadística	194
8.1.	Diagrama de bloques del sistema de diálogo desarrollado para el proyecto DIHANA	202
8.2.	Estructura de campos definida para el Registro del Usuario (<i>UR</i>) en DIHANA	205
8.3.	Conjunto de objetivos definidos en los escenarios de la tarea DIHANA . . .	206
8.4.	Ejemplo de diálogo generado con el simulador de usuarios	208
8.5.	Ejemplo de diálogo generado con el simulador de usuarios	209
8.6.	Evolución de la medida <i>#no-vistas</i> con respecto a la incorporación de nue- vos diálogos simulados	212
8.7.	Evolución de las medida <i>#error</i> con respecto a la incorporación de nuevos diálogos simulados	213
8.8.	Evolución de las medidas <i>%estrategia</i> y <i>%correcta</i> con respecto a la incor- poración de nuevos diálogos simulados	213
8.9.	Ejemplo de diálogo con un único objetivo adquirido mediante el simulador de diálogos	215
8.10.	Ejemplo de diálogo con dos objetivos adquirido mediante el simulador de diálogos	216

9.1.	Esquema del gestor de diálogo desarrollado para el proyecto EDECÁN . . .	227
9.2.	Ejemplo de diálogo para la tarea EDECÁN	228
9.3.	Conjunto de conceptos y atributos definidos para el etiquetado de los turnos de usuario	229
9.4.	Conjunto de conceptos y atributos definidos para el etiquetado de los turnos de sistema	230
9.5.	Registro del diálogo definido para la tarea EDECÁN	231
A.1.	¿Entendió al sistema cuando éste le hablaba?	255
A.2.	¿El sistema comprendió lo que usted le decía?	256
A.3.	¿Fue adecuado el ritmo de interacción?	256
A.4.	¿Supo usted cómo actuar en cada momento del diálogo?	257
A.5.	¿Con qué frecuencia el sistema fue lento en su respuesta?	257
A.6.	¿El sistema se comportó del modo esperado durante la conversación? . . .	258
A.7.	Resultados globales de los escenarios	258
A.8.	Resultados de los escenarios cerrados	259
A.9.	Resultados de los escenarios abiertos	259
B.1.	Script en LUA para la clasificación de muestras mediante un MLP	266
B.2.	Script en LUA para el aprendizaje de un MLP	267
D.1.	Arquitectura de VoiceXML	305
D.2.	Constructor de pronunciaciones de Voice Toolkit	313
D.3.	Tabla de fonemas definida por la IPA	315

Capítulo 1

Introducción

En este capítulo se precisa el contexto en el que se enmarca la tesis doctoral, exponiendo brevemente qué se entiende por sistema de diálogo, cuáles son los módulos principales que los constituyen y cuáles son los aspectos fundamentales que conforman la gestión del diálogo. En último lugar se precisan los objetivos definidos para la tesis doctoral y se detalla la estructura de capítulos que conforma este documento.

1.1. Introducción. Sistemas de diálogo hablado

El diccionario de la Real Academia Española de la Lengua define el diálogo como la *plática entre dos o más personas que alternativamente muestran sus ideas o afectos*. La acción de conversar es el modo más natural para resolver un gran número de acciones cotidianas entre los seres humanos: obtener una determinada información, contratar un servicio, solicitar un pedido, conocer el estado de un determinado proceso, etc.

Por este motivo, un interés histórico dentro del campo de las Tecnologías del Habla ha sido utilizar estas tecnologías en aplicaciones reales, especialmente en aplicaciones que permitan a una persona utilizar su voz para obtener información mediante la interacción directa con una máquina o para controlar un determinado sistema. El objetivo es disponer de sistemas que faciliten la comunicación persona-máquina del modo más natural posible, es decir, a través de la conversación.

Un sistema de diálogo puede, de esta forma, entenderse como un sistema automático capaz de emular a un ser humano en un diálogo con otra persona, con el objetivo de que el sistema cumpla con una cierta tarea (normalmente suministrar una cierta información o llevar a cabo una determinada tarea).

Descartando el más simple de los casos, este tipo de aplicaciones requieren una secuencia de interacciones entre la persona y la máquina para conseguir que el usuario consiga su propósito. Así, el objetivo del usuario se alcanza gradualmente con la interacción con la máquina a lo largo de varios turnos de diálogo. Este hecho obliga a dotar al sistema de la complejidad necesaria para que pueda: referenciar durante el diálogo la información que haya aparecido anteriormente, tomar la iniciativa para reconducir el diálogo dentro del dominio en el que se ha definido, solicitar información necesaria para cumplir el objetivo solicitado, requerir aclaraciones cuando exista la duda sobre la información aportada por el usuario, etc.

Construir una aplicación informática que pueda mantener una conversación con una persona de manera natural sigue siendo hoy en día un reto, dada la gran cantidad de fuentes de conocimiento que son necesarias y las limitaciones de las tecnologías utilizadas para obtener información del usuario. No obstante, los constantes avances de la investigación en Tecnologías del Habla han permitido que sean factibles actualmente sistemas de comunicación persona-máquina mediante la voz, capaces de interactuar con cierto grado de flexibilidad (iniciativa mixta en el desarrollo del diálogo). Estos sistemas están siempre orientados a tareas de información muy específica (dominios semánticos restringidos).

Disponiendo de un interfaz hablado, se libera al usuario de utilizar otros canales tradicionales como el teclado, el ratón o la pantalla. Dado este interfaz vocal, el número de entornos y tareas en los que pueden aplicarse estos sistemas es enorme: sistemas que proporcionen información sobre horarios y precios de transportes públicos, sistemas de información meteorológica, servicios de banca electrónica, aplicaciones accesibles desde los vehículos, sistemas que faciliten el acceso a la información a personas con discapacidades, acceso a servicios y control de máquinas vía telefónica (fija o móvil), portales de voz, etc.

La Figura 1.1 resume las acciones básicas que debe realizar un sistema de diálogo para cumplir la finalidad global para la que fue diseñado.

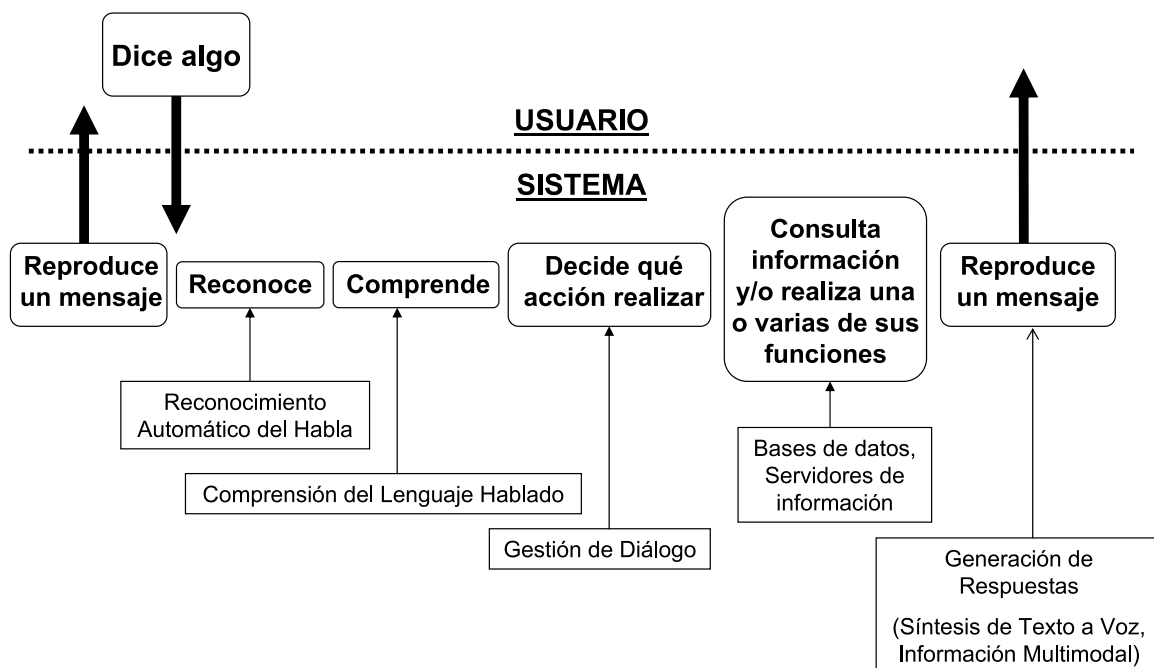


Figura 1.1: Diagrama de acciones de un sistema de diálogo

Tal y como se observa en esta figura, el sistema genera un mensaje inicial, normalmente para dar la bienvenida o informar al usuario sobre las características y funcionalidades del sistema. Tras cada intervención del usuario, el sistema debe realizar un conjunto de acciones básicas que se repiten cíclicamente como respuesta a cada acción del usuario:

- *Reconocer* la secuencia de palabras mencionadas por el usuario.
- Extraer el *significado* de dichas palabras, es decir, comprender qué información es útil en el dominio del sistema.

- Realizar operaciones de *acceso a base de datos* u otros recursos del sistema, en los que se almacena la información que solicita el usuario o se registran las operaciones que desea conocer.
- Decidir qué acción o acciones deben realizarse a continuación de cada solicitud del usuario, es decir, qué *respuesta* debe suministrar el sistema.
- Reproducir un mensaje *hablado* que informe al usuario qué acción ha seleccionado el sistema.

De este modo, dado el gran número de operaciones que deben realizarse, es habitual un desarrollo modular de los sistemas de diálogo hablado, lo que permite desglosar las dificultades entre los diferentes componentes del sistema. Un sistema de estas características se puede describir en términos de los siguientes módulos:

- *Módulo de Reconocimiento Automático del Habla*, reconoce la señal vocal pronunciada por el usuario y proporciona la secuencia de palabras reconocida más probable (o las k más probables).
- *Módulo de Comprensión del Habla*, a partir de las(s) secuencia(s) de palabra(s) reconocida(s), el sistema obtiene una representación semántica de su significado.
- *Gestor de Diálogo*, considera la interpretación semántica de la petición del usuario, la historia del proceso de diálogo, la información de la aplicación disponible en ese punto y el estado del sistema, y determina la siguiente acción que debe tomar el sistema siguiendo la estrategia del diálogo.
- *Módulo de Consulta a la Base de Datos de la Aplicación*, recibe peticiones de consulta a la base de datos por parte del gestor de diálogo, las procesa y devuelve el resultado al gestor.
- *Módulo de Generación de Respuestas*, recibe la respuesta del sistema en forma de cierta representación formal y 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. La respuesta del sistema proporcionada por el generador de respuestas puede incorporar otras modalidades de información (vídeo, tablas con datos, gestos a reproducir por un avatar...).
- *Sintetizador de Texto a Voz*, componente que recibe la respuesta del sistema como texto en lenguaje natural y genera la correspondiente señal de audio, que será la respuesta que llegará al usuario.

La figura 1.2 muestra la arquitectura modular descrita para el desarrollo de sistemas de diálogo hablado.

En cuanto a las metodologías utilizadas para el desarrollo de los diversos módulos que componen el sistema de diálogo, cabe destacar la utilización de técnicas basadas en reglas o la aplicación de métodos estadísticos, basados usualmente en el aprendizaje de un modelo a partir de un corpus de diálogos.

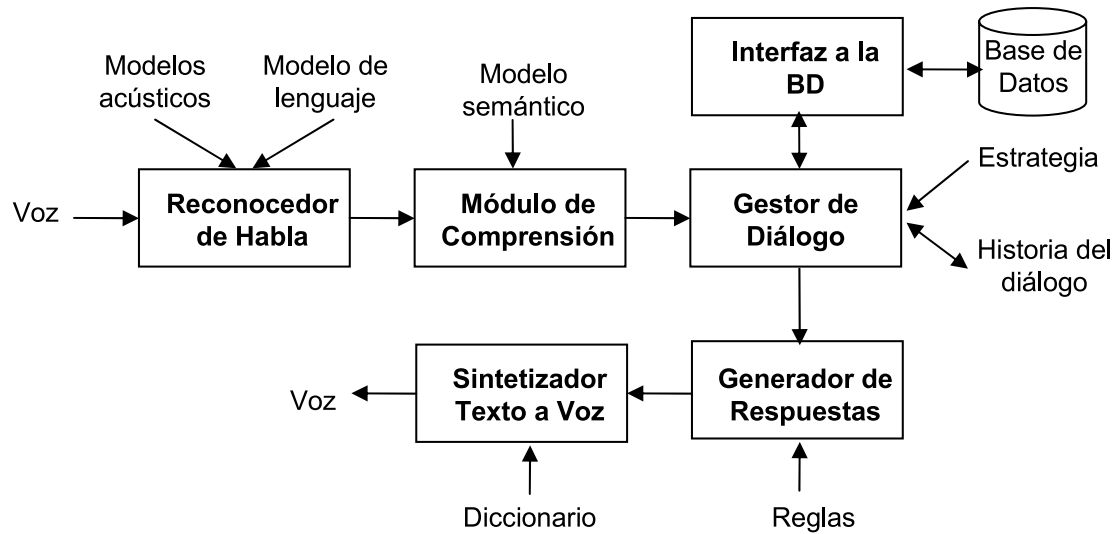


Figura 1.2: Arquitectura modular de un sistema de diálogo hablado

Los sistemas que se desarrollan utilizando métodos basados en reglas tienen como principales inconvenientes el elevado coste del diseño del modelo (dada la necesidad de que participe personal especializado) y la limitación en su aplicación, ya que el cambio en la tarea del sistema implica el diseño de un nuevo modelo. Los sistemas que se desarrollan según el paradigma inductivo, usando métodos de aprendizaje a partir de datos, presentan la dificultad previa de la adquisición de un corpus. Éste debe proporcionar un número de muestras suficientemente grande que permita inferir, aplicando técnicas de aprendizaje automático, un modelo que describa adecuadamente la tarea considerada.

La motivación para entrenar modelos a partir de datos reales en lugar de basarse en las técnicas tradicionales es clara. Los avances en el campo de los sistemas de diálogo hacen que los procesos de diseño, implementación y evaluación de las estrategias de gestión del diálogo sean cada vez más complejos, lo que ha posibilitado que el foco de interés de la comunidad científica se desplace de forma creciente de los métodos heurísticos a las técnicas basadas en modelos aprendidos a partir de datos. Las aproximaciones estadísticas que modelan mediante probabilidades los diferentes procesos, estimando los parámetros del modelo a partir de corpus de diálogos, se han introducido en todas las áreas de investigación de los sistemas de diálogo durante la última década (Young, 2002a) (Young, 2002b). Estos modelos pueden entrenarse a partir de diálogos reales, pudiendo modelar la variabilidad en los comportamientos de los usuarios. El objetivo final es desarrollar sistemas con un comportamiento más robusto, mayor portabilidad, escalables y con mayor facilidad para realizar su adaptación al usuario o a nuevos dominios.

Una de las técnicas más utilizadas para la adquisición de un corpus de diálogos y el posterior desarrollo de sistemas de diálogo consiste en un método de simulación en el que una persona realiza funciones (típicamente de la gestión de diálogo) que simulan el comportamiento del sistema a desarrollar. Esta técnica, denominada Mago de Oz (Fraser y Gilbert, 1991), facilita la evaluación de la utilidad y necesidades del sistema durante las fases iniciales de desarrollo. Adicionalmente a la adquisición del corpus de datos necesario para entrenar los modelos estadísticos, otra utilidad de esta técnica consiste en tener la

posibilidad de redefinir o verificar las decisiones de diseño del sistema, pudiendo detectar puntos débiles antes de la interacción del sistema en un contexto real. Los principales inconvenientes de la técnica de Mago de Oz se basan en la dificultad de simular el comportamiento real del sistema y el esfuerzo necesario para llevarla a la práctica (reclutar un número suficiente de usuarios que permitan evaluar el sistema y adquirir los datos necesarios, desarrollo de escenarios que indiquen los objetivos de los diálogos a desarrollar, etc.). Llevar a cabo esta técnica con éxito requiere además del entrenamiento adecuado de la persona encargada de simular el comportamiento del sistema (*Mago*), de forma que se apliquen las acciones adecuadas ante idénticas situaciones en el diálogo. Asimismo, suele ser necesario el desarrollo de interfaces que faciliten la labor del Mago y reduzcan el tiempo de generación de la respuesta del sistema.

El etiquetado de los corpus adquiridos, necesario para poder realizar el aprendizaje de modelos estadísticos, puede llevarse a cabo a diferentes niveles: morfosintáctico, sintáctico, prosódico o pragmático (Gibbon, Moore, y Winski, 1997). El etiquetado morfosintáctico se basa en asignar una categoría (usualmente gramatical) a cada una de las palabras. Cada una de estas categorías puede disponer de un conjunto de subcategorías asociadas. El etiquetado sintáctico se basa en establecer un enlace entre cada frase del corpus y una estructura jerárquica que representa las relaciones entre las palabras. La anotación prosódica divide la frase en unidades para tener en cuenta la pronunciación de palabras, pausas, ritmos, etc. En la anotación pragmática, el objetivo es obtener una descripción abstracta de la intención del usuario que ha llevado a cabo el diálogo. Este etiquetado suele realizarse en términos de *actos de diálogo*. En esta aproximación, los diálogos se dividen en turnos (puede definirse un turno de diálogo como una o varias intervenciones del usuario o del sistema). Cada uno de estos turnos se divide en fragmentos a los que se les asigna una etiqueta que representa el significado o intencionalidad del mismo (por ejemplo: respuesta, pregunta, confirmación, etc.). Existe un gran número de aplicaciones disponibles para llevar a cabo el etiquetado de los corpus: DAT (Allen y Core, 1997), MMAX (Müller y Strube, 2003), MATE (Carletta y Isard, 1999), AGTK (Bird y Liberman, 2001), Alembic (Day et al., 1997), Callisto (*callisto.mitre.org*), AnnoTag (Reithinger y Kipp, 1998), DiET (Netter et al., 1998), Nite (Carletta et al., 2003), Nb (Flammia y Zue, 1995), MUP (Glass y Eugenio, 2002), EMU (Cassidy y Harrington, 2004), etc.

El desarrollo de los sistemas de diálogo actuales ha sido posible gracias a las prestaciones alcanzadas en los diversos módulos implicados, en gran medida, gracias a los avances en el reconocimiento automático del habla, que constituye el primer eslabón del sistema.

Otras líneas de investigación, dentro del campo de los sistemas de diálogo, en las que se están realizando trabajos actualmente son la incorporación de diferentes tipos de información tanto a la entrada como la salida del sistema (sistemas multimodales), la adaptación de las características del sistema a los diferentes perfiles del usuario y el desarrollo e investigación de metodologías que faciliten la evaluación de los sistemas y la obtención de estrategias de diálogo óptimas. Áreas importantes en las que se están introduciendo paulatinamente los sistemas de diálogo están relacionadas con los dominios de respuesta a preguntas (*question answering*), recuperación de información (*information retrieval*), traducción automática, robótica o domótica.

En todo sistema de diálogo, una vez determinada cuál será la siguiente intervención del sistema en términos de un lenguaje conceptual, realizar su conversión a texto en lenguaje natural (generación de respuesta) y de éste a voz (síntesis del habla) es un problema,

comparativamente, menor. Los principales retos se encuentran en el desarrollo de los otros módulos, dado que los problemas de la comprensión del lenguaje natural y del control del diálogo por parte del sistema presentan mayores dificultades. Anteriormente estos problemas se abordaban con una modelización del conocimiento lingüístico, formalizado mediante reglas deducidas por usuarios expertos en la tarea considerada. En esta vía de investigación, más tradicional, se siguen desarrollando todavía un gran número de trabajos. La aplicación de los métodos estadísticos aprendidos a partir de un corpus de diálogos etiquetado ha comenzado a dar resultados en tiempos mucho más recientes y constituyen actualmente una vía de investigación innovadora.

1.1.1. Reconocimiento automático del habla

A lo largo de las últimas décadas, en el campo del reconocimiento automático del habla, se ha progresado desde el reconocimiento de palabras aisladas dentro de conjuntos reducidos de vocabulario hasta el reconocimiento de habla continua y con conjuntos de vocabulario cada vez mayores. Estos avances han supuesto que la comunicación con los sistemas de diálogo pueda efectuarse cada vez más de una forma más natural.

Diferentes aspectos que suelen tenerse en cuenta a la hora de establecer clasificaciones de los reconocedores son el tipo de usuarios permitidos (sistemas independientes o dependientes del usuario), el estilo de habla soportado (reconocedores de palabras aisladas, palabras conectadas o de habla continua) o el tamaño del vocabulario (pequeño, medio o gran vocabulario).

La complejidad de la tarea de reconocimiento radica en diversos problemas: la variabilidad acústica (cada persona pronuncia los sonidos de manera diferente cuando habla), la confusión acústica (muchas palabras suenan de forma similar, lo que dificulta su distinción), el problema de la coarticulación (las características de los sonidos pronunciados pueden variar en función de los sonidos vecinos), las palabras fuera del vocabulario, los fenómenos propios del habla espontánea (interjecciones, pausas, dudas, falsos comienzos, repeticiones de palabras, autocorrecciones...) y las condiciones del entorno (ruido, distorsiones del canal, limitaciones de ancho de banda...). Por este motivo, cuando se utilice un sistema de reconocimiento automático, se debe analizar la calidad de las palabras reconocidas o de los conceptos comprendidos por el sistema con el fin de detectar posibles errores o zonas de gran ambigüedad. Esta necesidad es aún más importante en los sistemas de diálogo, donde una mala interpretación de la frase pronunciada puede llevar al sistema a realizar un comportamiento erróneo, ya que la salida del reconocedor es el punto de partida del resto de módulos del sistema.

Entre las diferentes técnicas utilizadas para el desarrollo de reconocedores, sin duda, la aproximación estadística es actualmente la más utilizada. En esta aproximación, el problema del reconocimiento puede entenderse como encontrar la secuencia de palabras W pronunciadas dada una secuencia de datos acústicos A . Esta secuencia puede determinarse siguiendo la expresión:

$$W = \max_W P(W|A)$$

Utilizando la regla de Bayes, la expresión anterior puede reescribirse de la siguiente forma:

$$P(W|A) = \frac{P(A|W)P(W)}{P(A)}$$

donde $P(A|W)$ se denomina modelo acústico (probabilidad de obtener la secuencia acústica A cuando se ha pronunciado la secuencia de palabras W) y $P(W)$ es proporcionada por el modelo de lenguaje (probabilidad de pronunciar la secuencia de palabras). Dado que la probabilidad de la secuencia acústica es independiente de la secuencia de palabras, la expresión puede escribirse del siguiente modo:

$$W = \max_W P(A|W)P(W)$$

Para llevar a la práctica esta aproximación, la solución más utilizada actualmente consiste en el modelado de las unidades acústicas mediante modelos ocultos de Markov (HMM), como es el caso de reconocedores ampliamente utilizados por la comunidad científica como HTK (Hidden Markov Model Toolkit) (htk.eng.cam.ac.uk) o Sphinx (cmusphinx.sourceforge.net/html/cmusphinx.php).

El éxito de los HMM se basa principalmente en la existencia de algoritmos de aprendizaje automático de los parámetros del modelo (Rabiner, Juang, y Lee, 1996), así como en su capacidad para representar el habla como un fenómeno secuencial en el tiempo. Se han estudiado múltiples aproximaciones, como son los modelos discretos, semicontinuos o continuos, así como diversas topologías de los modelos.

Uno de los elementos imprescindibles para disponer de un reconocedor de habla continua es el modelo de lenguaje. Dado que la percepción de la acústica es a menudo insuficiente, incluso para las personas, para reconocer la secuencia de fonemas o palabras pronunciadas, es necesario tener un modelo de concatenación de palabras. Los modelos de lenguaje más utilizados son los basados en N-gramas (Bahl, Jelinek, y Mercer, 1990) (Jelinek, 1990) y los basados en gramáticas regulares (Segarra, 1993) o independientes del contexto (Jelinek, Lafferty, y Mercer, 1992). Las gramáticas suelen ser adecuadas para tareas reducidas, ya que permiten más precisión en el tipo de restricciones que imponen, pero son incapaces de representar la gran variabilidad del habla natural. Los N-gramas, sin embargo recogen de una forma más sencilla las concatenaciones entre palabras pero son muy adecuados cuando se dispone de un número suficiente de muestras de entrenamiento. En ambos casos la existencia de técnicas de aprendizaje automático (Inferencia Gramatical) para las Gramáticas (Fu y Booth, 1986), técnicas de estimación de parámetros para los N-gramas y técnicas de suavizado permite la obtención de modelos adecuados para el reconocimiento del habla. En el caso de HMM como representación acústico-fonética y de N-gramas como modelos de lenguaje, se construye una red de estados en que las palabras están representadas por la concatenación de modelos de los fonemas que las componen.

Los avances también son significativos en cuanto a la correcta identificación de los fonemas en entornos ruidosos. Actualmente, los métodos estadísticos proporcionan las mejores tasas de reconocimiento en estos entornos de acústica deficiente, como es el caso de la locución telefónica.

1.1.2. Comprensión del habla

En cuanto al proceso de comprensión, puede entenderse como un cambio en el lenguaje de representación, de lenguaje natural a un lenguaje semántico, de forma que se mantenga el significado del mensaje. Al igual que en el reconocedor de voz, el módulo de comprensión puede trabajar con varias hipótesis (tanto de reconocimiento como de comprensión) y con medidas de confianza.

Para afrontar el problema de la comprensión existen actualmente dos grandes aproximaciones: la comprensión basada en reglas y la comprensión basada en modelos estadísticos estimados a partir de datos.

Las alternativas basadas en reglas extraen la información semántica a partir del análisis sintáctico-semántico de las frases, utilizando gramáticas definidas para la tarea, o a partir de la detección de palabras (o secuencias de palabras) clave, con significado semántico. Algunos analizadores, con el objetivo de mejorar la robustez del análisis, combinan los aspectos sintácticos y semánticos de la tarea. Otras técnicas se basan en aplicar un análisis a dos niveles, en el cual se utilizan gramáticas para llevar a cabo un análisis detallado de la frase y extraer la información semántica relevante. Además, existen sistemas que utilizan analizadores basados en reglas aprendidas de forma automática a partir de un corpus de entrenamiento utilizando técnicas de procesamiento del lenguaje natural.

En el caso de los métodos estadísticos, el proceso se basa en la definición de unidades lingüísticas con contenido semántico y en la obtención de modelos a partir de muestras etiquetadas. Este tipo de análisis (Minker, 1998) (Segarra et al., 2002) emplea un modelo probabilístico para identificar los conceptos, marcadores y valores de los casos, para representar las relaciones entre los marcadores de los casos y sus valores y para decodificar semánticamente las pronunciaciones del usuario. El modelo se genera durante una fase de entrenamiento (aprendizaje), donde sus parámetros capturan las correspondencias entre las entradas de texto y su representación semántica. Una vez el modelo de entrenamiento se ha aprendido, se emplea a modo de decodificador para generar la mejor representación semántica de la entrada. De este modo, el proceso de comprensión se realiza de forma similar al reconocimiento del habla. Mediante el algoritmo de Viterbi puede interpretarse como un proceso de traducción de una frase de entrada (secuencia de palabras) en una frase de salida (secuencia de unidades semánticas).

La definición del lenguaje semántico se basa en un gran número de casos en la utilización del concepto de *frame*. En esta aproximación, la representación generada por el módulo de comprensión contiene *conceptos* (los diferentes tipos de consultas que puede realizar el usuario) y *atributos* (información que debe aportar el usuario para completar o modificar la consulta requerida al sistema). De este modo, todo mensaje enviado por el módulo de comprensión al gestor del diálogo tras cada intervención del usuario es un *frame*.

1.1.2.1. Medidas de confianza

Con la finalidad de que los módulos posteriores del sistema puedan corregir los posibles errores generados por el módulo de comprensión, la salida generada por el reconocedor puede no limitarse a una única frase, sino, como se ha comentado previamente, a múltiples hipótesis en forma de grafo de palabras (Aubert y Ney, 1995) o de las k mejores frases (las más probables para el reconocedor).

Una alternativa a estos métodos consiste en la utilización de medidas de confianza (típicamente un valor real entre 0 y 1) que representen la fiabilidad de cada una de las palabras. El objetivo de la estimación de medidas de confianza es evaluar la calidad de las palabras reconocidas y de los conceptos semánticos extraídos en la fase de comprensión de lenguaje natural, facilitando en lo posible la detección y corrección de errores por parte del gestor de diálogo. Las medidas de confianza obtenidas en estos dos módulos tienen como objetivo evaluar su comportamiento de forma que el gestor de diálogo pueda medir la calidad de la información recibida y en consecuencia, elegir la acción concreta a realizar: rechazar la frase, preguntar otra vez, o pedir confirmación de alguno de los datos obtenidos (Sturm, den Os, y Boves, 1999b) (San-Segundo et al., 2001a). De este modo, se posibilita que los módulos siguientes del sistema puedan operar con varias alternativas y tengan en cuenta en su funcionamiento la fiabilidad de las palabras.

Según la resolución de las medidas de confianza, pueden clasificarse en tres niveles diferenciados:

- Nivel de palabra: en este caso el objetivo es detectar palabras mal reconocidas. Para ello, se utilizan parámetros obtenidos del módulo de reconocimiento de voz, procedentes de la decodificación y del modelo de lenguaje.
- Nivel de concepto: en este caso se pretende detectar conceptos erróneos dentro de una frase determinada. Estas medidas de confianza son muy importantes para la gestión de diálogo, puesto que la información semántica se utiliza para realizar la gestión y decidir cuales van a ser las acciones del sistema en su interacción con el usuario. En este caso suelen utilizarse parámetros obtenidos del reconocedor de voz y del módulo de comprensión.
- Nivel de frase: en este nivel, el objetivo es detectar, por un lado, frases fuera del dominio de la aplicación, y por otro, frases del dominio con problemas en el reconocimiento que no tienen ninguna información semántica o concepto correcto. Se pretende detectar frases que no van a ser correctamente reconocidas y comprendidas por el sistema desarrollado, evitando realizar interpretaciones erróneas.

Aunque las medidas de confianza asociadas a las palabras en el proceso del reconocimiento son la medida más frecuentemente utilizada para denotar la fiabilidad de la información que se suministra al gestor de diálogo (Wessel, Macherey, y Schlüler, 1998) (Zhang y Rudnicky, 2001) (San-Segundo et al., 2001b) (Hazen, Seneff, y Polifroni, 2002) (López-Cózar et al., 1999) (Pérez-Piñar y García, 2005a), existen diferentes trabajos en los que se muestra además la utilidad de las medidas de confianza asociadas a conceptos y atributos durante el proceso de comprensión (Torres et al., 2005) (Bouwman y Hulstijn, 1998) (García et al., 2003) (Sama et al., 2005).

1.1.3. Gestión de diálogo

El gestor del diálogo (GD) es el elemento central del sistema de diálogo, dado que interacciona tanto con los módulos que componen la entrada del sistema como con los que generan la salida del mismo. La tarea principal del GD es controlar el flujo del diálogo y decidir las acciones a llevar a cabo como respuesta a la intervención del usuario. Para cumplir este objetivo, el GD debe interpretar la representación semántica del turno de

usuario en el contexto del diálogo, detectar y corregir errores producidos en los procesos de reconocimiento y comprensión, solicitar información a los módulos que controlan el acceso a las bases de datos del sistema y generar la respuesta del sistema apropiada al estado actual del diálogo.

Para completar esta serie de tareas, es necesario que el GD almacene la historia del diálogo y disponga de una representación que indique el estado actual del mismo. Además, requiere la definición de una estrategia que indique el comportamiento del sistema. El funcionamiento del sistema de diálogo tiene una gran dependencia de la calidad de la estrategia definida para el GD. No obstante, su definición no es nada trivial debido a la gran variabilidad de intervenciones que pueden presentar los usuarios que utilicen el sistema. Tres aspectos clave para la definición de la estrategia son la decisión del grado de iniciativa con que se le dota al sistema (iniciativa del sistema, estrategias de iniciativa mixta e iniciativa por parte del usuario), la elección de la estrategia de confirmación de datos del usuario (confirmaciones explícitas, implícitas o mixtas) y la definición de mecanismos para la detección y corrección de errores, por ejemplo, mediante la definición de medidas de confianza en los procesos de reconocimiento y comprensión, a partir de las cuales el GD pueda estimar la fiabilidad de la información que recibe.

En cuanto a las aproximaciones propuestas por los diferentes grupos de investigación para llevar a cabo la gestión del diálogo, cabe destacar las dos líneas que se han comentado anteriormente: utilización de métodos basados en reglas y desarrollo de metodologías estadísticas.

Actualmente, tanto los métodos estadísticos como los basados en reglas permiten desarrollar sistemas capaces de comprender automáticamente las intervenciones de las personas, expresadas en su lenguaje natural. En cambio, el problema del control o gestión del diálogo por parte del sistema aún no ha sido satisfactoriamente resuelto aplicando métodos estadísticos, debido a la ausencia de corpus de talla suficiente y a la mayor dimensión del espacio de sucesos objeto de modelización. Los sistemas que, hoy en día, gestionan automáticamente los diálogos persona-máquina suelen basarse en la aplicación de reglas o bien son sistemas híbridos que integran ambos tipos de métodos.

Conforme la complejidad de los sistemas es mayor, las estrategias basadas en la definición de reglas fijadas a mano presentan dificultades mayores para precisar el conjunto de reglas que serían necesarias y la imposibilidad de comprobar si el sistema funcionará correctamente en todos los posibles escenarios del diálogo. Los modelos estadísticos permiten realizar un mejor modelado de la variabilidad de comportamientos del usuario, presentando el inconveniente de su dependencia con respecto a la calidad y cantidad de las muestras disponibles en los corpus adquiridos. Este hecho hace que la definición de metodologías que modelen la gestión del diálogo de forma puramente estadística, mediante el aprendizaje de un modelo a partir de un corpus de muestras etiquetadas, haya sido el principal reto establecido para el trabajo desarrollado en esta tesis.

1.2. Objetivos y líneas de investigación tratadas en la tesis

La principal área de investigación en la que se enmarca esta tesis doctoral es el desarrollo de metodologías para realizar la gestión del diálogo en sistemas con interfaz oral, en

concreto, mediante el habla espontánea. Dentro de esta línea de investigación, se ha prestado un especial énfasis a la elaboración de estrategias de gestión basadas en el aprendizaje automático de un modelo a partir de un corpus de datos etiquetado.

Los diferentes trabajos llevados a cabo abordan los principales problemas que pueden presentarse en la gestión del diálogo y que conforman los principales retos establecidos para la tesis. Entre estos retos cabe destacar el desarrollo de aproximaciones de gestión basadas únicamente en un modelo aprendido a partir de un corpus de datos; la elaboración de metodologías para el tratamiento de las situaciones no incluidas en el modelo aprendido; y la definición y desarrollo de diferentes métodos y medidas para la evaluación del funcionamiento del gestor de diálogo.

Las soluciones desarrolladas para el tratamiento de estas líneas de investigación se han implementado con el objetivo de facilitar la adaptación de las metodologías propuestas para afrontar cambios en las tareas abordadas. En la evaluación de las diferentes metodologías desarrolladas, se ha prestado una especial atención al estudio de modelos para la simulación de usuarios.

De este modo, las líneas de investigación principales que se definieron para la presente tesis doctoral se materializan en los siguientes objetivos:

1. Estudio y desarrollo de diferentes metodologías estadísticas para el desarrollo de gestores de diálogo.
2. Estudio e implementación de diferentes metodologías para la evaluación de sistemas de diálogo.
3. Estudio y desarrollo de diferentes modelos para la simulación de usuarios.
4. Definición de metodologías que permitan la estandarización de los sistemas de diálogo y su adaptación a diferentes tareas.

1.3. Estructura de la tesis

En cuanto a la estructura del documento, la tesis está comprendida por un total de diez capítulos. Una vez presentados en este capítulo los objetivos que fueron definidos para la tesis e introducido el contexto en el que se enmarca la misma, el capítulo segundo aborda de manera más detallada el estado de arte relativo a los sistemas de diálogo hablado. En primer lugar, se presentan los diferentes sistemas más representativos desarrollados por grupos y centros de investigación. Seguidamente, se incide en las soluciones propuestas por los diferentes grupos para la resolución de los problemas relativos a la gestión de diálogo, evaluación de los sistemas, la utilización de técnicas de simulación de diálogos y modelado del usuario para realizar dicha evaluación y obtener estrategias óptimas de gestión.

El capítulo tercero se dedica a la descripción de la tarea DIHANA (Benedí, Varona, y Lleida, 2004), proyecto en el cual se enmarca la tesis doctoral. El objetivo definido para el proyecto DIHANA (diciembre 2002 - noviembre 2005) fue el estudio y desarrollo de un sistema robusto de diálogo para el acceso a sistemas de información mediante habla espontánea en diferentes entornos. La tarea seleccionada fue el acceso a un sistema de información sobre horarios, precios, tipos de tren, tiempos de recorrido y servicios de

trenes españoles de largo recorrido. Uno de los principales intereses científicos del proyecto fue la adquisición de un gran corpus oral de diálogo en lengua castellana. Dado que los modelos de diálogo estadísticos que se presentan en la tesis se aprendieron a partir de este corpus, gran parte del capítulo se dedica a la descripción del proceso seguido para realizar su adquisición mediante la técnica del Mago de Oz y su posterior etiquetado.

El capítulo cuarto resume las características principales del sistema de diálogo implementado para el proyecto DIHANA, necesario para poder evaluar en la práctica las metodologías desarrolladas en el ámbito de la gestión de diálogo. En el capítulo se describe la arquitectura modular definida para el sistema, las características más relevantes de los diversos módulos y el formato XML desarrollado para facilitar la comunicación entre dichos módulos.

El capítulo quinto presenta dos aproximaciones basadas en reglas para la gestión de diálogo, desarrolladas de forma paralela a la adquisición y etiquetado del corpus DIHANA. La primera de estas metodologías implementa la estrategia definida para que el Mago de Oz gestionase el diálogo durante la adquisición del corpus. La segunda aproximación trata de simplificar esta estrategia, diferenciándose principalmente en el método aplicado para llevar a cabo las confirmaciones de los datos marcados con baja fiabilidad. De este modo, se evalúan diferentes estrategias para la confirmación de los datos aportados por el usuario en la gestión del diálogo. En este capítulo se introducen además definiciones comunes al conjunto de gestores de diálogo desarrollados para la tesis: el registro del diálogo, la utilización de medidas de confianza y los métodos de simplificación de la información semántica proporcionada por el módulo de comprensión.

El capítulo sexto describe el núcleo central del trabajo desarrollado en el marco de la tesis: el desarrollo de modelos estadísticos para la gestión del diálogo. Dada la importancia del gestor de diálogo como auténtico cerebro del sistema de diálogo y la complejidad que presenta la aplicación de modelos estadísticos para el modelado de su comportamiento, este capítulo puede considerarse el de mayor relevancia en la tesis. La metodología propuesta se basa en la modelización estadística de las secuencias de actos de diálogo, del usuario y del sistema. Su aprendizaje se ha realizado a partir del corpus DIHANA. Para obtener la respuesta final del gestor, mediante el uso del modelo desarrollado, se han definido y evaluado dos estrategias: la obtención de las probabilidades del modelo mediante una estimación por máxima verosimilitud y la obtención de la nueva respuesta del sistema a través de un proceso de clasificación.

El capítulo séptimo describe diferentes técnicas y medidas utilizadas generalmente para realizar la evaluación de sistemas de diálogo, mostrándose los resultados obtenidos en la evaluación de los gestores de diálogo desarrollados. Las técnicas propuestas se pueden clasificar en dos grupos: la evaluación mediante usuarios reales y la utilización de particiones del corpus y validación cruzada, realizando la evaluación turno a turno.

El capítulo octavo presenta el simulador de usuario desarrollado para evaluar y mejorar el comportamiento del gestor estadístico presentado en el capítulo anterior. El modelo de usuario se basa en el modelo definido para llevar a cabo la gestión del diálogo, teniendo un comportamiento completamente estadístico. En el capítulo se describe el modelo propuesto y se presentan los resultados de su evaluación y de la evaluación conjunta de la interacción del simulador con el gestor de diálogo. Asimismo, se presenta una metodología para la simulación diálogos y la obtención automática de un corpus etiquetado.

El capítulo noveno se dedica a la descripción de la adaptación de las metodologías de

gestión propuestas para afrontar nuevas tareas. La adaptación se ha llevado a cabo en el ámbito del proyecto de investigación EDECÁN (Lleida et al., 2006), cuyo objetivo es aumentar la robustez de un sistema de diálogo de habla espontánea a través del desarrollo de tecnologías para la adaptación y personalización del mismo a distintos contextos acústicos y de aplicación. Una de las tareas seleccionadas para el sistema es la reserva de instalaciones deportivas en la Universitat Politècnica de València. En el capítulo se resumen las principales características de la tarea y se exponen los resultados de la adaptación de la metodología estadística desarrollada en DIHANA para la gestión del diálogo.

A continuación se presentan las conclusiones del trabajo desarrollado en la tesis y las líneas de investigación en las que se está trabajando actualmente.

Seguidamente, se incorporan una serie de anexos en los que se amplía con mayor detalle (en forma de texto, gráficos y tablas) la información presentada en diferentes capítulos de la tesis: adquisición del corpus DIHANA, desarrollo del sistema de diálogo diseñado para el proyecto, resultados de la evaluación de los gestores de diálogo y el modelo de simulación de usuario. En último lugar, se recopilan las referencias bibliográficas citadas a lo largo de la tesis.

Capítulo 2

Estado del arte

El segundo capítulo de la tesis se dedica a presentar un estado del arte de los principales campos de investigación en los que se enmarca el trabajo desarrollado. Se han distinguido un total de cinco apartados:

- Sistemas de diálogo hablado.
- Gestión del diálogo.
- Evaluación de sistemas de diálogo.
- Técnicas de simulación de usuarios.
- Adaptación al usuario y al dominio.

2.1. Sistemas de diálogo hablado

La investigación en las tecnologías de los sistemas de diálogo, tal y como los conocemos hoy en día, se remonta a finales de la década de los ochenta como resultado de dos grandes proyectos con financiación gubernamental: el programa de sistemas de lenguaje hablado DARPA en los Estados Unidos y el programa Esprit SUNDIAL en Europa. El dominio del programa DARPA fue la consulta de información relativa a vuelos (Air Travel Information Services, ATIS). En el proyecto participaron un conjunto de laboratorios de investigación de Estados Unidos. El principal objetivo de la investigación fue el estudio y desarrollo de las tecnologías relativas al reconocimiento del habla y comprensión del lenguaje, bajo el dominio de la reserva de vuelos utilizando el canal telefónico (DARPA, 1992) (ARPA, 1994). Dado que todos los participantes del proyecto utilizaron la misma base de datos, fue posible comparar el funcionamiento de los diferentes prototipos desarrollados, llevándose a cabo un gran esfuerzo para realizar evaluaciones periódicas de los diferentes sistemas. Actualmente, el corpus de diálogos ATIS sigue siendo un recurso utilizado por desarrolladores y evaluadores de sistemas de diálogo.

La tarea ATIS sirvió de marco para el desarrollo de los primeros proyectos, como los llevados a cabo en AT&T, concretamente el proyecto AMICA (Pieraccini, Levin, y Eckert, 1997), donde se aplicaron diferentes métodos estocásticos en un sistema de diálogo con iniciativa mixta. ATIS también fue el punto de partida de las investigaciones del MIT

y de la CMU, dentro de este proyecto se desarrollaron los sistemas CMU ATIS (Ward, 1991) y MIT ATIS (Seneff, Hirschman, y Zue, 1991) (Zue et al., 1993).

El proyecto SUNDIAL (Speech Understanding and Dialogue) (Peckham, 1993), financiado por la Comunidad Europea, tuvo como dominio la consulta de horarios de trenes y aviones en inglés, francés, alemán e italiano. El objetivo del proyecto fue construir sistemas de diálogo en tiempo real capaces de mantener una conversación con el usuario siguiendo una estrategia cooperativa. Además de tratar aspectos de reconocimiento y comprensión del habla, un tema de estudio en el que se centró la investigación fue el modelado del diálogo hablado, desarrollándose diferentes aproximaciones para realizar la gestión del diálogo.

La investigación llevada a cabo en SUNDIAL condujo a un gran número de proyectos con financiación europea centrados en el modelado del diálogo, como VerbMobil (Bos et al., 1999), DISC (Bernsen y Dybkjaer, 1997) y ARISE (Den et al., 1999). También cabe mencionar el sistema Philips (Aust et al., 1994) (Aust et al., 1995), sistema comercial bajo el dominio de consultas sobre horarios de trenes desarrollado a partir de la investigación llevada a cabo en SUNDIAL. Dentro del proyecto europeo ARISE se desarrollaron seis sistemas en paralelo: dos prototipos italianos basados en la tecnología desarrollada por el CSELT (Castagneri, Baggia, y Danieli, 1998) (Baggia, Castagneri, y Danieli, 2000), un prototipo francés desarrollado por el LIMSI (Lamel et al., 2000) (Den et al., 1999) y dos prototipos en holandés y uno en francés basados en la tecnología Philips.

Más recientemente, entre los programas de investigación de mayor importancia, cabe destacar DARPA Communicator. En este programa, con financiación gubernamental y centrado en el desarrollo de tecnologías del habla, participaron centros de investigación en Estados Unidos y Europa. El objetivo establecido fue el desarrollo de una nueva generación de sistemas de diálogo, utilizando como entrada del sistema no únicamente la voz sino existiendo también la posibilidad de incorporar otros tipos de información multimodal. Los sistemas desarrollados dentro de este programa soportan interacciones complejas con el usuario, desde el punto de vista que tanto el sistema como el usuario pueden iniciar la conversación, cambiar de tema o interrumpir al otro participante del diálogo. Los dominios de aplicación (en los que se incluye la planificación de viajes) requieren usualmente el acceso a múltiples fuentes de información, representando un avance con respecto a los sistemas desarrollados en los programas ATIS o SUNDIAL.

Los investigadores de la CMU desarrollaron el sistema Carnegie Mellon Communicator (www.speech.cs.cmu.edu/Communicator) (Rudnicky et al., 1999), que permite obtener información de itinerarios complejos que incluyen reservas múltiples de vuelos, hoteles y alquiler de coches. La arquitectura del sistema se basa en la definición de módulos agentes de dominio que se encargan de la gestión de la información más específica. Paralelamente, investigadores de la Universidad de Colorado desarrollaron el sistema CU Communicator (Ward y Pellom, 1999), que aborda la misma tarea.

De los diversos proyectos que siguieron, sobresalen los sistemas implementados por el grupo del LIMSI: el sistema PARIS-SITI, dedicado a facilitar información turística sobre la capital francesa, el sistema MASK (Multimodal Multimedia Service Kiosk) (Gauvain et al., 1995) y el sistema sobre información de trenes RAILTEL (Billi y Lamel, 1997).

Entre los grupos de investigación internacionales dentro del campo de las tecnologías de diálogo hablado, en Estados Unidos cabe destacar: el Grupo Spoken Language Systems del MIT (groups.csail.mit.edu/sls), el grupo CSLU del Institute of Science and Techno-

logy de Oregon (*cslu.cse.ogi.edu*), el Sphinx Group de la CMU (*www.speech.cs.cmu.edu*), el grupo CSLR (Center for Spoken Language Research) de la Universidad de Colorado (*cslr.colorado.edu*), el grupo Conversational Interaction and Spoken Dialogue Research de la Universidad de Rochester (*www.cs.rochester.edu/research/cisd*), etc. Entre las compañías estadounidenses que participaban activamente en este campo de investigación cabe destacar AT&T (*www.att.com*), los laboratorios Bell (*www.bell-labs.com*), Microsoft (*www.microsoft.com*), IBM (*www.ibm.com*), etc. En el ámbito europeo existen también un gran número de centros de investigación: el grupo Natural Interactive Systems Laboratory de Odense (Dinamarca) (*www.nis.sdu.dk*), el LIMSI Spoken Language Processing Group en París (*www.limsi.fr/TLP*), el Centre for Speech Technology de la Universidad de Edimburgo (*www.cstr.ed.ac.uk*), el grupo Speech Communication and Technology del KTH en Estocolmo (*www.speech.kth.se/speech*), el grupo Language Technology del DKFI en Alemania (*www.dfki.de*), CSELT en Italia (*www.csel.it*), el Speech Research Group de la Universidad de Cambridge (*mi.eng.cam.ac.uk/research/speech*), etc.

A nivel nacional cabe destacar los siguientes grupos: Pattern Recognition and Human Language Technology (PRHLT) del Instituto Tecnológico de Informática de la Universitat Politècnica de València (*www.iti.upv.es/~prhlt*); el Centro de Tecnologías y Aplicaciones del Lenguaje y del Habla (TALP) (*www.talp.upc.es*) del Departamento de Teoría de Señal y Comunicación y el Grupo de Procesado de Lenguaje Natural del Departamento de Lenguajes y Sistemas Informáticos (GPLN) (*www.lsi.upc.edu/~nlp/web*) de la Universidad Politècnica de Catalunya; el Grupo de Investigación en Señales, Telemática y Comunicaciones de la Universidad de Granada (*ceres.ugr.es*); el Grupo de Tratamiento de Señal del Departamento de Teoría de la Señal y Comunicaciones de la Universidad de Vigo (*www.gts.tsc.uvigo.es/web*); el Grupo de Procesado de Señales de la Universidad Politécnica de Madrid (*www.ssr.upm.es*); el Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Sevilla (*www.us.es/julietta*), el Laboratorio de Comunicación Oral ROBERT WAYNE NEWCOMB del Departamento de Arquitectura y Tecnología de Sistemas Informáticos de la Universidad Politécnica de Madrid (*labaudio.datsi.fi.upm.es*), el Grupo de Tecnologías de las Comunicaciones del Instituto de Investigación en Ingeniería de Aragón en el Centro Politécnico Superior de la Universidad de Zaragoza (*i3a.unizar.es*); el Grupo de Reconocimiento de Formas y Tecnologías del Habla del Departamento de Electricidad y Electrónica en la Facultad de Ciencias de la Universidad del País Vasco (*grah.ehu.es/grupo_voz*); el Grupo de Tecnología del Habla del Departamento de Ingeniería Electrónica de la ETSI de Telecomunicación en la Universidad Politécnica de Madrid (*www-gth.die.upm.es*) y el Grup de Reconeixement de Formes i Intel·ligència Artificial (RFIA) (*www.dsic.upv.es/users/rfia*) del Departament de Sistemes Informàtics i Computació de la Universitat Politècnica de València.

Tal y como se ha comentado en el capítulo de introducción de esta tesis, el desarrollo de sistemas de diálogo donde la iniciativa pueda ser compartida entre el usuario y el sistema, requiere un alto grado de desarrollo de las tecnologías involucradas: representación robusta de la voz, reconocimiento del habla, tratamiento del habla espontánea, tratamiento y comprensión del lenguaje y modelización del diálogo. A continuación se citan unas breves características de los sistemas de diálogo hablado más destacados. Estos sistemas se han agrupado teniendo en cuenta el grupo de investigación que los ha desarrollado. En el apartado referente a la gestión de diálogo se amplía la información relativa al desarrollo de gestores de diálogo en el ámbito de los diferentes proyectos.

- **Grupo Spoken Language Systems del MIT Computer Science and Artificial Intelligence Laboratory**, Cambridge (E.E.U.U.)

Los investigadores del Spoken Language Systems Group (SLS) del MIT desarrollaron una arquitectura para la implementación de sistemas de diálogo, que denominaron GALAXY (Seneff et al., 1998) (Seneff, Lau, y J.Polifroni, 1999). Se trata de la arquitectura de referencia en el marco del programa DARPA Communicator. Se basa en el modelo cliente-servidor, en el que cada uno de los módulos del sistema actúa como servidor de una determinada tarea (reconocimiento, comprensión, etc.).

En la arquitectura se define un módulo (*Hub*) encargado de centralizar la comunicación entre los diversos módulos. El hub recibe los envíos de los diferentes módulos y los redirecciona a su destinatario. La representación utilizada para llevar a cabo esta comunicación utiliza el formato de frames. La Figura 2.1 muestra los módulos que conforman la arquitectura, definiéndose módulos dependientes del dominio del sistema (por ejemplo, el gestor de diálogo y el módulo de comprensión del lenguaje) e independientes del mismo (como el servidor de audio). En esta arquitectura, el flujo de información viene controlado por la programación que se realice en el hub central.

Basándose en esta arquitectura, se han desarrollado sistemas de diálogo para facilitar información en tareas de información meteorológica, planificación de viajes aéreos, información de carreteras, etc. Ejemplos de estos sistemas son VOYAGER (Glass et al., 1995), sistema de información para viajeros, PEGASUS (Zue et al., 1994), sistema de acceso a un sistema de reservas on-line, o JUPITER (Zue et al., 2000), sistema de información meteorológica. A continuación, se resumen las características de los sistemas más relevantes desarrollados por el MIT.

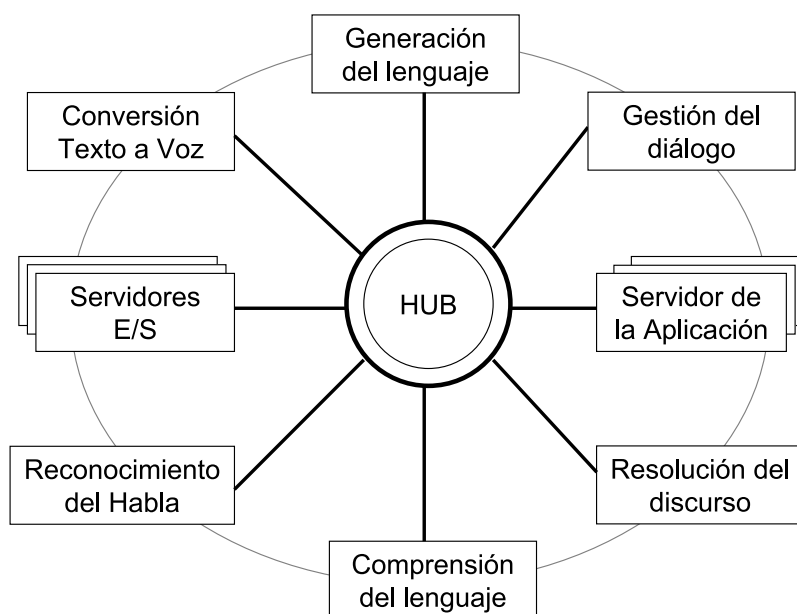


Figura 2.1: Arquitectura definida en el Galaxy Communicator

- **DINEX**: DINEX (Seneff y Polifroni, 1996) es un sistema conversacional que proporciona información sobre restaurantes de Boston (nombre, dirección, horario, rango de precios, tipo de cocina, reserva...) y cómo llegar a ellos. El sistema obtiene la información de tres fuentes: una guía de restaurantes de Boston, una guía de viajes de Boston y las Páginas Amarillas DINEX. Utiliza información multimodal: el usuario interactúa con la voz y mediante clicks del ratón en mapas y listas; el sistema puede responder mediante voz sintetizada e incluso mostrar por pantalla la página web o menús del restaurante seleccionado.

En cuanto a los módulos más relevantes que conforman el sistema, se utiliza el reconocedor automático del habla SUMMIT (Glass, 2003), el sistema de comprensión TINA (Seneff, 1992) y el módulo de generación de respuestas GENESIS (Baptist y Seneff, 2000).

- **JUPITER**: JUPITER es un sistema de diálogo que proporciona información meteorológica de diferentes ciudades alrededor del mundo (temperatura, velocidad del viento, etc.). Desde su puesta en marcha en mayo de 1997 ha registrado más de 100.000 llamadas, con una tasa de reconocimiento de palabras del 90 % y una tasa de comprensión correcta del 80 % de las consultas realizadas dentro del dominio del sistema (vocabulario de 2000 palabras).

Los principales retos tecnológicos tratados por el sistema JUPITER son:

- Buscador virtual: Dado que el teléfono es el único canal de comunicación con el usuario, JUPITER posibilita el acceso a información meteorológica procedente de Internet sin necesidad de que el usuario disponga de esta conexión.
- La interacción se lleva a cabo únicamente con la voz, con lo que los esfuerzos se han centrado en generar las respuestas con el mayor grado de amigabilidad posible y en la mejora de la calidad de la voz sintética.
- Comprensión de contenidos: Al utilizar la información contenida en la web, es necesario realizar tareas de recuperación de la información (*information retrieval*), comprensión y síntesis de información procedente de múltiples fuentes de la web.
- Sistema multilingüe: Se prevé posibilitar la comprensión de lenguas diferentes al inglés (español, japonés y chino mandarín).
- Reconocimiento y comprensión del habla robustos, tanto para tratar los retos que presenta la voz telefónica como para el modelado del lenguaje dependiente del dominio.

En cuanto al funcionamiento básico del sistema, la base de datos con información se actualiza tres veces al día. Esta información se remite al módulo de comprensión TINA, que determina y etiqueta los tipos de información meteorológica disponibles en cada zona. El sistema utiliza el reconocedor SUMMIT, un módulo de gestión del diálogo basado en reglas, el generador de respuestas GENESIS y el sintetizador de voz ENVOICE (Yi, Glass, y Hetherington, 2000).

- *MERCURY*: El sistema de diálogo MERCURY (Seneff y Polifroni, 2000), creado en 1999, posibilita la búsqueda de información de horarios y precios de vuelos (trayectos entre más de 200 ciudades con posibilidad de escalas) por vía telefónica. El vocabulario del sistema es de unas 1000 palabras. El sistema utiliza información aérea procedente de la red de viajes SABRE, gestión de diálogo basada en reglas y las tecnologías descritas para el reconocimiento automático del habla, generación de respuesta y síntesis de texto a voz.
- *PEGASUS*: Sistema de diálogo, también mediante el canal telefónico, que proporciona información sobre vuelos en Estados Unidos. Esta información es en tiempo real, actualizándose continuamente a lo largo del día. Los diferentes módulos del sistema se basan en las tecnologías desarrolladas por el MIT.
- *PENATES*: El sistema PENATES (groups.csail.mit.edu/sls/applications/penates.shtml) proporciona información sobre cerca de mil restaurantes en el área metropolitana de Boston. El sistema responde a consultas sobre tipos de cocina, rangos de precio, horarios de apertura, acceso a discapacitados, direcciones y números de teléfono. La principal característica diferenciadora de PENATES con respecto al resto de sistemas del MIT es que sus modelos han sido entrenados enteramente a partir de un corpus de usuarios simulados. Además, emplea módulos de comprensión del lenguaje y gestión del diálogo independientes del dominio.
- *VOYAGER*: El sistema VOYAGER proporciona información turística y de viajes en el área metropolitana de Boston. Ejemplos de consultas que pueden realizar los usuarios del sistema son la búsqueda de lugares de interés, visualización de mapas y fotografías aéreas, petición de información sobre rutas locales y solicitud de datos en tiempo real sobre el tráfico. Desde su primera versión en 1989, VOYAGER se ha utilizado como banco de pruebas para el desarrollo de los sistemas del MIT. Actualmente soporta información multilingüe (inglés, japonés e italiano). El sistema utiliza la tecnología descrita en los proyectos anteriores. Para consultas en tiempo real sobre el tráfico en una ruta determinada, el sistema utiliza la información obtenida del sistema SmartRoute, pudiendo informar sobre los incidentes que puedan existir a lo largo de dicha ruta.
- *ORION*: ORION (Seneff, Chuu, y Cyphers, 2000) es un sistema conversacional que realiza tareas off-line para contactar posteriormente con el usuario en instantes prefijados y comunicarle la información oportuna. Para ello, ORION solicita al usuario la información necesaria para llevar a cabo la tarea requerida, incluyendo el número de teléfono del usuario y el instante en el que desea que el sistema contacte con él. De este modo, la actividad del sistema se reparte entre la solicitud por parte del usuario de nuevas tareas (vía telefónica o mediante un interfaz gráfico) y la realización de aquellas que están pendientes.

Se trata de un sistema todavía en desarrollo. Se espera que el sistema responda a los tipos de consulta soportados por los sistemas ya desarrollados por el MIT (consultas sobre el tráfico, información de vuelos, información meteorológica), además de otras funcionalidades como la simple reproducción de un mensaje hablado a modo de avisos programados.

El usuario puede acceder al sistema de forma telefónica o mediante un interfaz web, en el que puede visualizar o modificar sus tareas pendientes. Se han desarrollado dos gestores de diálogo (basados en reglas) que se encargan, respectivamente, de registrar las nuevas tareas solicitadas y de ejecutarlas.

- *WHEELS*: El sistema WHEELS (Meng et al., 1996) realiza búsquedas en una base de datos de anuncios clasificados de automóviles. Estas búsquedas pueden realizarse vía telefónica o mediante comandos escritos. El sistema proporciona respuestas sintetizadas y tablas en pantalla. Es un sistema bilingüe (inglés y español). Entre las características más relevantes cabe mencionar la posibilidad que el usuario pueda solicitar ayuda dinámica en cualquier momento del diálogo y la capacidad del sistema para responder a preguntas específicas sobre la información que se está mostrando en pantalla.
- *WebGALAXY*: El sistema WebGALAXY (Lau et al., 1997) proporciona información de vuelos, datos meteorológicos e información turística y de tráfico en Boston. Los usuarios interactúan con el sistema mediante la voz o entradas escritas. El proyecto, que dio comienzo en 1996, tuvo como principal objetivo añadir a la arquitectura GALAXY la posibilidad de consultar información almacenada en la web.

■ **Carnegie Mellon University (CMU), Pittsburgh (E.E.U.U.)**

- *Ariadne* (Denecke, 2002): Se trata de una plataforma para el desarrollo rápido de prototipos de sistemas de diálogo hablado independiente de la tarea y del idioma. El programador, para ello, debe definir la estrategia de diálogo y adaptarla a sus requerimientos. Para su definición se utilizan ficheros JavaScript.
 - *BusLine* (Bennett et al., 2002): Se trata de una aplicación desarrollada en el estándar VoiceXML (McGlashan et al., 2004), utilizando la plataforma Tellme (www.tellme.com/platform), que facilita información de autobuses de distintas zonas de Pittsburgh. Utiliza la tecnología de reconocimiento automático del habla de Nuance (www.nuance.com) y el sintetizador de texto a voz Lernout&Hauspie's.
 - *NBA Game Update Line* (Bennett et al., 2002): Aplicación desarrollada en VoiceXML, mediante la plataforma Tellme, que proporciona información sobre resultados y estadísticas de partidos recientes de la NBA.
 - *Reading Tutor (project LISTEN)* (www.cs.cmu.edu/~listen): Se trata de un tutor de lectura dirigido a niños. El funcionamiento básico consiste en mostrar por pantalla cuentos e historias y escuchar si la pronunciación que se realiza de los mismos es correcta. El tutor interviene cuando el lector comete errores, duda en alguna palabra, solicita ayuda o el programa tiene la sensación de que la lectura se realiza con dificultad. Para ello, el sistema utiliza una versión adaptada del reconocedor Sphinx-II de la CMU. Funciona sobre el sistema operativo Windows.
 - *CMU Communicator*: Se trata de uno de los sistemas de diálogo hablado clásicos desarrollado, como se ha comentado previamente, en el ámbito de un proyecto financiado por el programa DARPA Communicator. Utiliza una iniciativa
-

del diálogo mixta, proporcionando información sobre viajes en avión. El sistema actúa como un agente de viajes, siendo capaz de proveer información sobre viajes con escalas, reservas de hoteles y alquiler de coches. Utiliza información en tiempo real obtenida de la web (vuelos) y es capaz de operar en un rango de 500 ciudades.

La arquitectura del sistema se basa en el modelo GALAXY, comentado en los sistemas de diálogo desarrollados por el MIT. Utiliza el reconocedor Sphinx-II y el módulo de comprensión semántica Phoenix (Ward y Issar, 1994). Este último módulo se basa en una gramática semántica, escrita a mano, para extraer la secuencia de unidad de conceptos semánticos a partir de la respuesta generada por Sphinx. La respuesta generada por Phoenix se suministra a un módulo, denominado Helios, encargado de proporcionar al gestor de diálogo una medida de la fiabilidad de la información suministrada por los dos módulos anteriores, utilizando para ello las medidas de confianza generadas por Sphinx.

El sistema gestiona el diálogo usando planificaciones y agendas. El gestor de diálogo, denominado Agenda, está controlado por una serie de manejadores que son ordenados dinámicamente a medida que el usuario va interactuando con el sistema. Cada uno de estos manejadores puede enlazarse con conceptos definidos para realizar la comprensión semántica, realizar llamadas al módulo de generación de respuestas o solicitar información a módulos dependientes de la tarea (por ejemplo, consultas a la base de datos de vuelos). El generador de respuestas en lenguaje natural utiliza una metodología mixta basada en plantillas y en la generación estocástica de respuestas. Finalmente, se utiliza el sintetizador Festival (www.cstr.ed.ac.uk/projects/festival) para realizar la conversión de texto a voz.

La primera versión del sistema data del año 1999. Desde entonces, se han realizado multitud de cambios, como la incorporación de las funcionalidades de reservas de hoteles y de coches, la elaboración de un componente estocástico para la generación de respuestas y el desarrollo de nuevos modelos de lenguaje. Uno de los principales objetivos del sistema es facilitar la adaptación sencilla a otras tareas. Para ello, se utilizan módulos agentes del dominio que contienen la información más específica de la tarea.

- Sistemas desarrollados utilizando *Olympus/RavenClaw*:

Olympus/RavenClaw (Bohus et al., 2007b) es una arquitectura para desarrollar sistemas de diálogo que tiene sus orígenes en el CMU Communicator. Cada módulo del sistema funciona como un servidor independiente y se comunica a través del hub Galaxy.

La Figura 2.2 muestra el conjunto de módulos que componen la arquitectura. En la configuración por defecto, *Olympus* utiliza el reconocedor automático del habla Sphinx-II, el sintetizador de texto a voz Festival, el módulo de comprensión Phoenix, el módulo para tratamiento de medidas de confianza Helios, el generador de respuestas en lenguaje natural Rosetta (basado en plantillas de respuestas) y el gestor del diálogo Ravenclaw. Pueden utilizarse diferentes sintetizadores, como festival, Theta o Swift. Cualquiera de estos módulos puede reemplazarse por otro, posibilitándose asimismo la incorporación de nuevos

módulos desarrollados en C, Java, Python, Lisp o Perl.

Ravenclaw (Bohus y Rudnicky, 2003) es un servidor para gestionar el diálogo desarrollado en C++. El diálogo se representa mediante un árbol, con condiciones previas y posteriores a cada uno de los nodos, y acciones a realizar tras visitar cada uno de ellos. Estos nodos son responsables de realizar tareas del diálogo, como solicitar una determinada información al usuario, presentar información o acceder a bases de datos. Durante la ejecución, los nodos del árbol se sitúan en una pila y se van visitando en el orden especificado por las condiciones previas especificadas para cada uno de ellos. El diseñador del gestor debe especificar manualmente las reglas a aplicar en cada nodo.

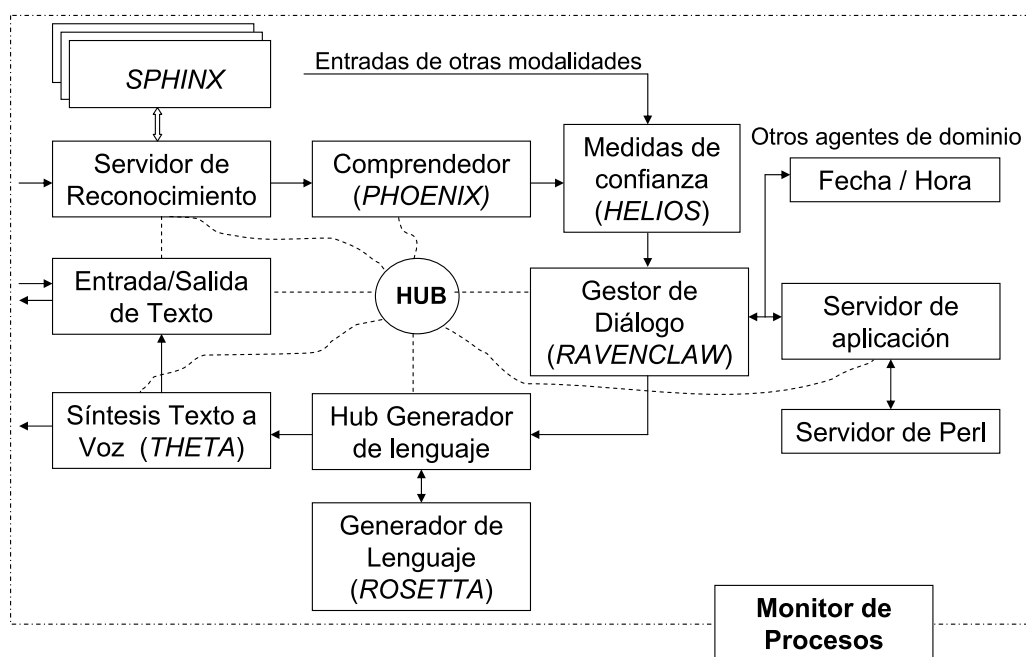


Figura 2.2: Arquitectura Olympus para el desarrollo de sistemas de diálogo

A continuación se mencionan las características básicas de un conjunto de sistemas de diálogo desarrollados utilizando la arquitectura Olympus/RavenClaw:

- *Conquest* (Bohus et al., 2007a): Sistema de diálogo hablado que proporciona información sobre conferencias y congresos. Se desarrolló para los congresos Interspeech-2006 y IJCAI-2007. Utiliza el reconocedor Sphinx-II, un gestor del diálogo basado en RavenClaw, un generador de respuestas basado en plantillas en lenguaje natural y el sintetizador Cepstral (www.cepstral.com).
- *Let's Go!* (Raux et al., 2005): Se trata de un sistema de diálogo hablado que proporciona información sobre la red de autobuses metropolitana de Pittsburgh. El principal objetivo del proyecto fue crear un sistema de diálogo con un comportamiento básico que posibilite maximizar el número de usuarios y potenciar su uso por parte de ancianos o personas con dificultad con el inglés.

La arquitectura del sistema está compuesta por un acceso telefónico, el reconocedor Sphinx-II con los modelos acústicos obtenidos del corpus de usuarios del CMU Communicator (se utilizan dos modelos diferentes dependiendo del género del usuario, ejecutándose en paralelo y seleccionando aquel que proporciona los mejores resultados), el módulo de comprensión semántica Phoenix y un gestor del diálogo basado en RavenClaw.

- *RoomLine* (www.ravenclaw-olympus.org/roomline.html): RoomLine es un sistema que proporciona asistencia para la reserva de salas de conferencia en la *School of Computer Science* de la CMU. El sistema dispone de una base de datos en la que se almacenan las características (tamaño, localización, equipamiento...) de 13 salas de conferencia. Durante el transcurso del diálogo, el objetivo del sistema es localizar la sala que satisfaga de mejor forma los requerimientos solicitados por el usuario. Actualmente existe una versión descargable del sistema en la que se utiliza una base de datos simulada, el reconocedor Festival y el sintetizador Cepstral.
 - *LARRI* (Bohus y Rudnický, 2002): Sistema de diálogo hablado multimodal que proporciona asistencia y guía al personal de mantenimiento de los aviones F18.
 - *Intelligent Procedure Assistant (IPA)* (Aist et al., 2003): Sistema multimodal desarrollado con el objetivo de guiar a los astronautas de la Estación Espacial Internacional durante la ejecución de tareas de mantenimiento y verificaciones del funcionamiento de dispositivos
 - *Vera* (www.ravenclaw-olympus.org/vera.html): Sistema que utiliza el canal telefónico, siendo su dominio suministrar mensajes a un usuario externo y realizar llamadas con la función despertador.
 - *MeetingLine* (www.ravenclaw-olympus.org/meetingline.html): Sistema de diálogo que utiliza el canal telefónico para acceder a información pregrabada sobre reuniones y citas.
 - *TeamTalk* (www.ravenclaw-olympus.org/teamtalk.html): Interfaz de diálogo hablado que facilita la comunicación entre un operador humano y un equipo de robots. En la tarea propuesta, el operador suministra órdenes a los diferentes robots para que localicen determinados objetos y los conduzcan a una determinada localización.
 - *Sublime* (www.ravenclaw-olympus.org/sublime.html): Sistema de diálogo multimodal desarrollado para funcionar en una PDA y realizar la gestión de información personal contenida en ella.
 - *Madeleine* (www.ravenclaw-olympus.org/madeleine.html): Sistema de diálogo que utiliza texto, tanto como entrada como salida. El dominio seleccionado es el diagnóstico médico.
 - *RavenCalendar* (www.ravenclaw-olympus.org/ravencalendar.html): Sistema de diálogo multimodal para realizar consultas a un calendario (*Google calendar*) y realizar consultar sobre anotaciones realizadas (citas, notas, clases) o incorporar nuevos eventos.
-

- *Proyecto USI (Universal Speech Interface):*

El proyecto USI (Rosenfeld et al., 2000), también conocido como *Speech Graffiti*, tiene como principal objetivo el diseño de un interfaz universal (independiente de las aplicaciones) para la comunicación hablada de forma que pueda utilizarse para facilitar el desarrollo de sistemas de diálogo. Para ello, se ha analizado la comunicación de las personas con diversos dispositivos, con el objetivo de desarrollar un interfaz universal a utilizar para que la comunicación hablada persona-máquina sea lo más efectiva y eficiente posible.

El objetivo de Speech Graffiti es crear un protocolo de interacción con el sistema que proporcione una alternativa intermedia entre el lenguaje natural y los diálogos en los que la iniciativa recae en el sistema. Para conseguir este objetivo, se definen estructuras de comunicación independientes del dominio del sistema y palabras clave que gobiernan la interacción del usuario con un servicio determinado (por ejemplo, “starting over” para borrar la historia previa del diálogo). La idea fundamental es que un usuario previamente instruido en las construcciones básicas del protocolo de comunicación pueda interactuar fácilmente con cualquier aplicación compatible con Speech Graffiti.

En (Tomko y Rosenfeld, 2004) se realiza un estudio comparativo entre un mismo sistema con dos interfaces de comunicación: utilizando lenguaje natural o Speech Graffiti. Los resultados muestran que un 74 % de los usuarios prefieren Speech Graffiti, aún siendo los porcentajes de éxito de los diálogos muy similares en ambos interfaces.

La principal ventaja de Speech Graffiti reside en su utilización en entornos donde el reconocimiento de la voz sea dificultoso, siendo una alternativa interesante al lenguaje natural. Los principales inconvenientes son la generación de diálogos que pueden llegar a ser poco naturales y la necesidad de formar previamente al usuario.

Utilizando el paradigma USI se han desarrollado los sistemas que se presentan a continuación (www.speech.cs.cmu.edu/usi):

- *MovieLine*: Se trata de un sistema con iniciativa mixta implementado con la misma tecnología que el CMU Communicator (Sphinx-II, Phoenix, gestor del diálogo basado en el proyecto Universal Speech Interface de la CMU y sintetizador Festival). Proporciona información actualizada semanalmente sobre la cartelera de cines en Pittsburgh. Además del acceso telefónico, permite la interacción con el usuario a través de la web.
- *ApartmentLine*: Sistema desarrollado también bajo el paradigma USI teniendo como dominio el acceso a información sobre alojamiento en Pittsburgh.
- *FlightLine*: Sistema que proporciona información sobre salidas, llegadas y terminales de vuelo.
- *Gadget*: Sistema desarrollado para el control remoto de dispositivos: sistemas Hifi, aparatos domésticos...

- **Centre for Speech Technology (CTT), KTH, TMH, Telia Research**, Estocolmo (Suecia).

- *Adapt* (Gustafson et al., 2000): Se trata de un sistema de diálogo multimodal para realizar la búsqueda de apartamentos dentro del mercado inmobiliario de Estocolmo. El sistema, que utiliza información real obtenida de la web, permite que el usuario puede interactuar mediante la voz o mediante pulsaciones en un mapa interactivo. Utiliza agentes animados.
 - *August* (Gustafson, Lindberg, y Lundeberg, 1999): Sistema de diálogo desarrollado para atender a los visitantes del Centro Cultural de Estocolmo dentro del programa de Capital Cultural Europea durante 1998 y 1999. Utiliza un agente animado que interactúa con el usuario a través de la voz, texto, gestos y movimientos de cabeza. Se desarrolló con la intención que proporcionase información sobre varios dominios (el más sencillo de ellos es información sobre restaurantes).
 - *Higgins* (Skantze, Edlund, y Carlson, 2006): El objetivo del proyecto HIGGINS es la investigación y el desarrollo de un sistema de diálogo con la finalidad de estudiar y evaluar diferentes estrategias para el tratamiento de errores. Entre los módulos que constituyen la arquitectura del sistema cabe destacar un simulador de entornos en tres dimensiones.
 - *Olga* (Beskow, Elenius, y McGlashan, 1997): Se trata de una figura animada tridimensional con la que el usuario puede comunicarse mediante el habla. El sistema ayuda al usuario a encontrar la información presente en las bases de datos utilizadas (consejos sobre la utilización de hornos microondas). Además, puede tomar su propia iniciativa, por ejemplo, dando consejos al usuario. El gestor de diálogo utiliza un modelo basado en reglas y plantillas.
 - *Proyecto de diálogo WAXHOLM* (Carlson y Hunnicutt, 1996): WAXHOLM proporciona información sobre el tráfico naval en el archipiélago de Estocolmo. Fue desarrollado para la compañía Washolm, que dispone de una flota de una veintena de barcos que conectan doscientos puertos. La arquitectura del sistema es similar a ATIS, disponiendo de una serie de módulos capaces de actuar como servidores independientes. Además, proporciona información multimodal: voz, tablas de texto e imágenes.
 - *Sistemas desarrollados mediante la plataforma ATLAS* (Melin, 2001).

ATLAS es una librería desarrollada en Java con la finalidad de facilitar el desarrollo de aplicaciones multilingües y multimodales, y en especial sistemas de diálogo. El diseño de la plataforma se basa en un modelado en capas del sistema, siendo ATLAS una capa intermedia entre la capa dependiente de la aplicación y la capa que contiene los diferentes módulos del sistema de diálogo (Figura 2.3). Su objetivo es implementar la mayoría de las funcionalidades del sistema que sean independientes de la tarea, así como definir funciones para dar soporte a los módulos dependientes de la misma.

Se han desarrollado sintetizadores de texto a voz, reconocedores y sistemas de verificación del usuario compatibles con esta plataforma.

 - *CTT-Bank* (Melin, Sandell, y Ihse, 2001): Sistema desarrollado para realizar operaciones bancarias a través de la voz. Los usuarios deben identificarse con el sistema mencionando su nombre y un número de identificación
-

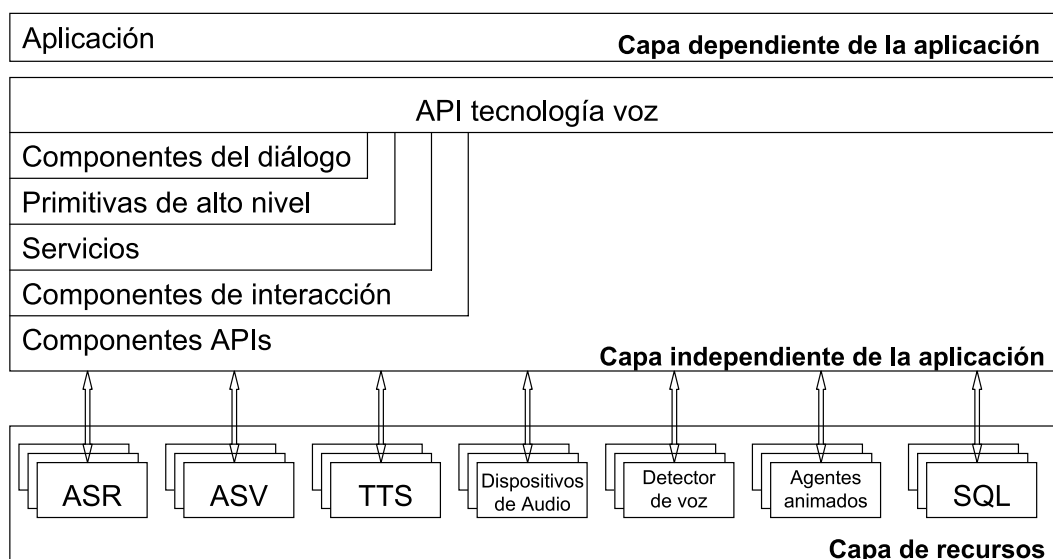


Figura 2.3: Arquitectura definida en la plataforma ATLAS

personal. Una vez el usuario se ha validado, puede consultar el estado de sus cuentas de ahorro, solicitar un listado de las últimas transacciones y transferir fondos entre cuentas. La capa dependiente de la aplicación define un conjunto de componentes de diálogo para implementar los servicios bancarios y las operaciones de identificación del usuario. La capa de recursos contiene un conjunto de reconocedores del habla entrenados cada uno de ellos con una gramática especializada.

- *PER (Prototype Entrance Receptionist)* (Packucs y Melin, 2001): Se trata de un recepcionista virtual multimodal y multilingüe que interactúa en la entrada del Departamento de Voz, Música y Sonido del CTT. Su funcionalidad es posibilitar el acceso de los empleados al departamento mediante el uso de la voz, sin necesidad de llaves. Utiliza verificación del usuario (lectura de un código que se muestra en pantalla) para autenticar a los empleados. El sistema incluye un agente animado, un gestor de diálogo que informa únicamente si se permite el acceso dando la bienvenida al usuario o solicita que se repita la secuencia mencionada, un dispositivo para desbloquear la puerta y un conjunto de sensores para detectar la presencia del usuario.
- *Picasso Impostor Trainer* (Elenius, 2001): Se trata de un sistema desarrollado para evaluar la habilidad de los hablantes para imitar las voces de otros usuarios. Para ello, cuenta con una base de datos de grabaciones de usuarios, un reconocedor automático del habla, un módulo de identificación del usuario, un detector de voz y un acceso telefónico. El usuario interactúa con el sistema tratando de imitar las grabaciones que le reproduce el sistema. Como resultado, el sistema le indica una serie de estadísticas que evalúan la semejanza entre la voz imitada por el usuario y la disponible en la base de datos.

- *Hörstöd* (Johansson, 1997): Es un sistema desarrollado para investigar si las personas con problemas en el oído pueden ser ayudadas durante una conversación telefónica utilizando las transcripciones generadas por un reconocedor de fonemas. Utiliza la mayor parte de componentes mencionados para el sistema Picasso.

▪ **Speech-based and Pervasive Interaction Group**, Tampere (Finlandia)

El grupo de investigación Speech-based and Pervasive Interaction de la Universidad de Tampere ha desarrollado una plataforma, denominada JASPIS (Turunen et al., 2005), para implementar sistemas de diálogo multilingües, distribuidos, adaptados al usuario y al entorno.

La arquitectura se basa en la utilización de tres elementos diferenciados: agentes, gestores y evaluadores.

El nivel superior de la arquitectura está formado por los gestores, conectados a un gestor central mediante una topología en estrella basada el modelo cliente-servidor. Los agentes son componentes que implementan las acciones de interacción con el usuario: reproducir mensajes del sistema, tomar decisiones durante el diálogo, etc. Los evaluadores se utilizan para verificar diferentes aspectos de los agentes, con la finalidad de determinar si son apropiados para tareas específicas del sistema. La información del sistema se almacena en bases de datos compartidas que pueden ser consultadas por todos los componentes de la arquitectura mediante el gestor de la información. La Figura 2.4 muestra la arquitectura de la plataforma JASPIS.

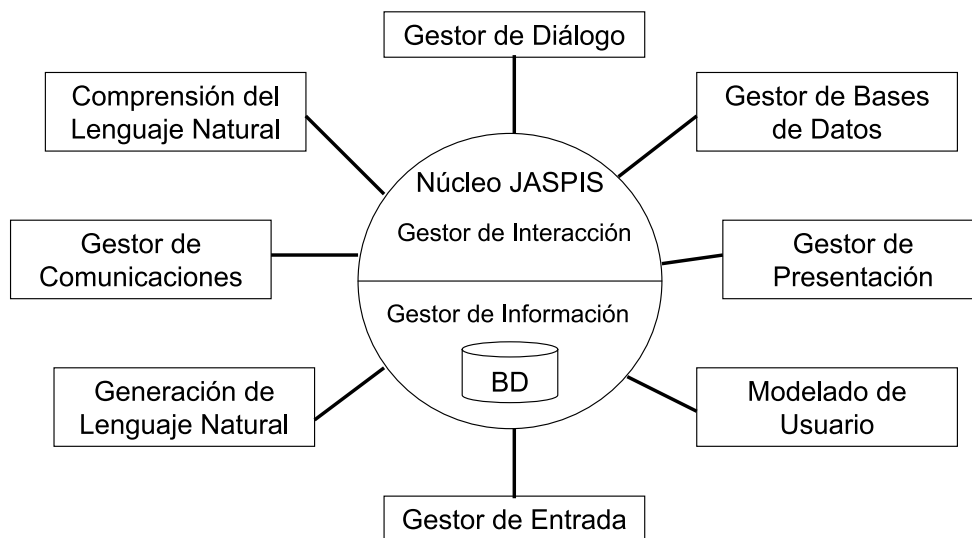


Figura 2.4: Arquitectura definida para la plataforma JASPIS

Mediante la arquitectura JASPIS se han desarrollado los siguientes sistemas:

- *Mailman / AthosMail* (Turunen y Hakulinen, 2000): Es un cliente de correo multilingüe (inglés y finlandés) diseñado para facilitar la lectura de los correos electrónicos mediante el uso del teléfono.

- *Busman*: Facilita información sobre horarios de autobuses en finlandés.
 - *Doorman* (Mäkelä et al., 2001): Sistema que proporciona información y guía a las personas que visitan el departamento a modo de portero virtual. Entre las funcionalidades del sistema se encuentran la lectura de los correos electrónicos, realización de operaciones como la apertura de puertas y suministro de información sobre la situación de despachos y personal del departamento.
- **Universidad de Pittsburgh**, Pittsburgh (E.E.U.U.)
- *ITSPOKE* (Spoken Dialogue for Intelligent Tutoring System): ITSPOKE (Litman y Silliman, 2004) es un proyecto de la Universidad de Pittsburgh con el principal objetivo de permitir a los estudiantes interactuar con un tutor virtual utilizando la voz. Se trata de una evolución del sistema Why2-Atlas (VanLehn et al., 2002) en la que se incorpora un interfaz hablado para la comunicación con el usuario. El sistema dispone de un interfaz gráfico con el que se inicia el diálogo, mostrándose en pantalla una pregunta sobre un concepto físico que el usuario debe desarrollar utilizando un cuadro de texto. Una vez analizada la respuesta escrita, el sistema realiza preguntas para clarificar esta respuesta inicial o solicitar información adicional (utilizando preguntas predefinidas teniendo cuenta los diferentes conceptos del dominio del sistema). Si el sistema detecta que la respuesta a alguna de las preguntas no es correcta, puede solicitarle al alumno una nueva respuesta escrita.
- Los módulos fundamentales del sistema son el reconocedor Sphinx-II; los módulos Why2-Atlas para el análisis sintáctico, semántico y gestión de diálogo (basada en modelos de estados finitos), y el sintetizador Cepstral.
- **Rochester University**, Rochester (E.E.U.U.)
- *TRAINS*: El proyecto TRAINS (Allen et al., 2000) de la Universidad de Rochester tuvo como principal objetivo el desarrollo de un sistema de diálogo con iniciativa mixta capaz de interactuar con tiempos de respuesta reducidos. El dominio seleccionado fue proporcionar información sobre rutas y horarios de trenes de mercancías. Al usuario se le proporcionaba un mapa en pantalla mostrando ciudades, conexiones y localizaciones de trenes. Mediante la voz el sistema le informaba de un conjunto de ciudades destino donde se requerían trenes. El objetivo del usuario era encontrar el conjunto de rutas más eficiente. Para ello, era necesario su entrenamiento previo. El sistema utilizaba el reconocedor Sphinx-II y el sintetizador TrueTalk. Durante el transcurso del proyecto se elaboraron varios prototipos (TRAINS91, TRAINS93, TRAINS95, TRAINS96).
 - *TRIPS*: El proyecto TRIPS (Ferguson y Allen, 1998) amplía el trabajo desarrollado en TRAINS, utilizando escenarios más complejos dentro del mismo dominio que TRAINS. El sistema desarrollado incorpora la posibilidad de comunicaciones mediante la voz, utilizando texto o mediante un interfaz gráfico. La arquitectura definida utiliza, al igual que DARPA Communicator, una serie de módulos que se comunican entre si mediante un *hub* central. Para realizar
-

esta comunicación se definió un lenguaje propio denominado KPML (*Knowledge Query and Manipulation Language*). El sistema consta de tres componentes fundamentales: el gestor de interpretaciones (*Interpretation Manager*), el agente de comportamiento (*Behavioral Agent*) y el gestor de generación (*Generation Manager*). La función del gestor de interpretaciones es procesar la entrada del usuario a partir de la interpretación semántica (secuencia de actos de diálogo) de la respuesta proporcionada por el reconocedor. El agente del comportamiento planifica que comportamiento va a seguir el sistema tras el análisis de la acción del usuario. El gestor de generación proporciona la respuesta final del sistema. La Figura 2.5 muestra la arquitectura desarrollada para el sistema TRIPS.

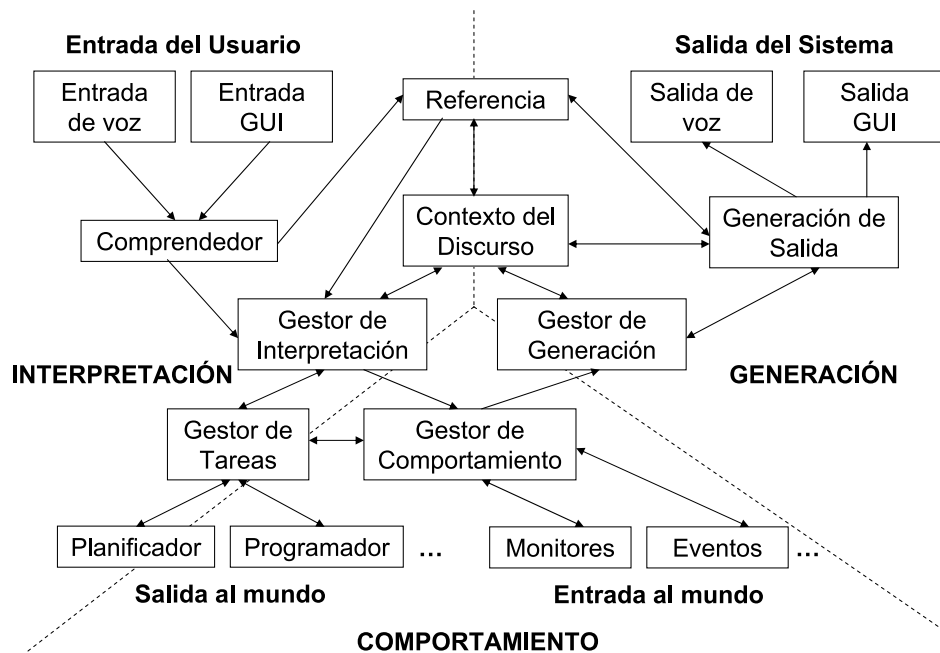


Figura 2.5: Arquitectura definida en el proyecto TRIPS

- **Center for the Study of Language and Information (CSLI), Stanford University, Stanford (E.E.U.U.)**

Los sistemas desarrollados por el grupo CSLI de la Universidad de Standford se basan en la utilización de la arquitectura OAA (Open Agent Architecture) (Cheyer y Martín, 2001), el reconocedor automático del habla Nuance, el sistema de comprensión del habla GEMINI (Dowding et al., 1993) y el sintetizador Festival.

El proyecto *WITAS* (Doherty et al., 1998) tiene como dominio el desarrollo de un sistema de diálogo para el control de un vehículo aéreo (un pequeño helicóptero teledirigido). El diálogo tiene en cuenta las órdenes recibidas y las condiciones del entorno. Entre los requerimientos impuestos al sistema se encuentran ser asíncrono, disponer de iniciativa mixta e interactuar en un entorno dinámico. La interacción con el sistema se lleva a cabo mediante un interfaz multimodal que permite combinar en un mismo diálogo voz, texto, gráficos, gestos, vídeos en tiempo real y datos extraídos de sensores; así como intercalar diferentes objetivos en un mismo diálogo.

- **Microsoft Research**, (research.microsoft.com)

El objetivo del proyecto Conversational Architectures de Microsoft Research (Horvitz y Paek, 1999) es desarrollar sistemas de diálogo que se asemejen lo máximo posible a la conversación entre personas. Para ello, se parte de la suposición inicial de que un sistema de diálogo ha de ser robusto independientemente de los errores que puedan cometerse en los diferentes procesos. El diálogo se modela como un proceso de toma de decisiones bajo cierta incertidumbre, en el que el sistema realiza un análisis de costes/beneficios para determinar qué acción llevar a cabo. El objetivo final es desarrollar un modelo computacional que represente las dependencias entre la incertidumbre existente y la teoría de decisión, utilizada para identificar la acción más adecuada.

Esta aproximación del diálogo se ha desarrollado mediante una arquitectura llamada Quartet (Paek y Horvitz, 2000). Se trata de una plataforma para desarrollar sistemas de diálogo multimodales. Los aspectos claves con los que se representa la incertidumbre se modelan mediante redes bayesianas. Se han definido cuatro niveles de representación que soportan la inferencia y toma de decisiones durante el diálogo: el canal de comunicación, los tipos de señal, la intención (o contenido semántico de la señal) y la conversación.

Utilizando la arquitectura Quartet se han desarrollado diferentes sistemas (Paek y Horvitz, 2000) (Paek y Horvitz, 1999), como *Presenter* (sistema para navegar por presentaciones Microsoft PowerPoint), *Deep Listener* (proyecto en el que se han desarrollado diferentes prototipos, entre ellos un sistema para interactuar con Microsoft Outlook) y *Bayesian Receptionist* (sistema que simula un recepcionista de escritorio en el entorno Microsoft Campus. Utiliza el reconocedor Microsoft Whisper y el módulo de comprensión del habla Microsoft NLPWin).

- **AT&T Labs Research**, (www.research.att.com)

- *AT&T-DARPA Communicator*: El sistema Communicator desarrollado por AT&T (Levin et al., 2000) sigue la arquitectura Galaxy. Utiliza el reconocedor de habla continua AT&T Watson (Goffin et al., 2005), que soporta gramáticas basadas en modelos estocásticos y basadas en reglas, y el sistema de comprensión CHRONUS (Levin y Pieraccini, 1995). Para almacenar la información del sistema, se utiliza el servidor de vuelos desarrollado por la Universidad de Colorado. Este servidor obtiene la información en tiempo real, accediendo a un portal web comercial, y la almacena en una base de datos local (SYBASE). Para la síntesis de voz se utilizó el sistema AT&T Next Generation. Además, existe un módulo temporizador encargado de remitir frames al hub central en intervalos de tiempo especificados. Este servidor se utiliza cuando el gestor de diálogo requiere al hub realizar una consulta a la base de datos. La información relativa al gestor de diálogo se expone en el siguiente apartado de la tesis.
- *How may I help you? (HMIHY)* (Gorin, Riccardi, y Wright, 1997): La arquitectura del sistema la conforman un reconocedor automático del habla, el módulo de comprensión semántica, el gestor de diálogo y una plataforma telefónica. La tarea del sistema es redireccionar las llamadas recibidas basándose en la respuesta del usuario a la pregunta inicial del sistema *¿En qué puedo ayudarle?*

De esta forma, el objetivo final del sistema es clasificar las llamadas del usuario determinando la funcionalidad que persiguen.

- *CoBotDS*: En el proyecto CoBotDS (Kearns et al., 2002), extensión del proyecto Cobot, se desarrolló un sistema de diálogo para el acceso a un servidor de chat en Internet llamado LambdaMOO. El sistema provee comunicación en tiempo real entre un usuario y múltiples usuarios del chat. Utiliza el canal telefónico y lenguaje natural.
- *DiSCoH*: DiSCoH (*Spoken Dialogue System for Conference Help*) (Andeani et al., 2006) es un sistema de diálogo con iniciativa mixta desarrollado para proporcionar información sobre conferencias. Se ha utilizado por primera vez en la conferencia IEEE/ACL SLT en diciembre de 2006. En el proyecto participan investigadores de AT&T, ICSI y la Universidad de Edimburgo. Uno de los objetivos principales del proyecto es la adquisición y etiquetado de un gran corpus de diálogos en lenguaje natural que se pondrá a disposición de la comunidad científica. Para realizar este etiquetado se propone la anotación de actos de diálogo, intervenciones del usuario, intenciones del usuario, éxito en la tarea, etc.

La estructura y dominio del sistema es muy similar a los de sistemas previos desarrollados por AT&T: W99 (Rahim et al., 2000), desarrollado para el congreso ASRU de 1999, y VoiceIF (Rahim et al., 2000) (AT&T Innovation Forum Workshop del año 2000). Para su implementación se han utilizado las herramientas AT&T VoiceTone (Gilbert et al., 2005) para el desarrollo de sistemas de diálogo hablado. Estas herramientas proporcionan un reconocedor, cuyos modelos acústicos y de lenguaje iniciales se entrenaron a partir del corpus W99, y un módulo de comprensión semántica, basado en la detección de entidades nombradas, *named entities*, y que utiliza medidas de confianza.

- *ELVIS*: La tarea del sistema ELVIS (Email Voice Interactive System) (Walker et al., 1997) es gestionar telefónicamente las consultas a un servidor de correo electrónico.

Se desarrollaron dos versiones del gestor del diálogo: una con iniciativa mixta y otra en la que la iniciativa recae en el sistema. Ambos gestores utilizan una máquina de estados para definir la estrategia. Cada estado especifica una respuesta inicial (que el sistema reproduce una vez se ha entrado en el estado correspondiente), una variable que especifica si es posible interrumpir o no al sistema (*barge-in*) y un mensaje de ayuda (a reproducir cuando el usuario así lo mencione).

- *NJFUN*: NJFUN (Litman et al., 2000) es un sistema de diálogo en tiempo real que proporciona información sobre actividades que pueden realizarse en New Jersey. Se desarrolló utilizando una plataforma propia de AT&T y utiliza una base de datos de la web para obtener la información del dominio del sistema. Esta base de datos está indexada utilizando tres atributos: tipo de actividad, localización y periodo del día.

El funcionamiento del gestor de diálogo se basa en requerir secuencialmente al usuario la actividad, localización y periodo de tiempo. Para ello se definió

una estrategia inicial: el sistema pregunta por un determinado atributo, si no lo obtiene vuelve a realizar la pregunta. Si por segunda vez el sistema no detecta en la intervención del usuario el atributo solicitado, solicita los siguientes atributos siguiendo un orden establecido. Cada vez que el sistema obtiene un atributo solicitado, puede requerir su confirmación o solicitar los siguientes. Una vez dispone de toda la información necesaria, se realiza la consulta a la base de datos.

Para tratar de depurar esta estrategia inicial, se utilizaron técnicas de aprendizaje reforzado (ver Sección 2.2.2) con el objetivo de seleccionar la estrategia del diálogo óptima en un conjunto de situaciones del diálogo, que se seleccionaron teniendo en cuanta las dificultades que encontraron los usuarios.

- *TOOT*: El dominio del sistema TOOT (Litman y Pan, 2002) es la búsqueda de información en la web sobre horarios de trenes utilizando el canal telefónico. Se desarrolló utilizando la plataforma mencionada en el sistema anterior, disponiendo de un reconocedor automático del habla (independiente del usuario, basado en gramáticas y con la capacidad de interrumpir al sistema), un interfaz telefónico, un sintetizador de texto a voz y módulos para especificar el gestor de diálogo y las funcionalidades ligadas con el dominio de la aplicación. El gestor de diálogo utiliza una máquina de estados finitos.
- *WebTalk*: WebTalk (Feng et al., 2006) es una herramienta para automatizar la creación de sistemas de diálogo hablado que utilicen información de la web. El principal objetivo es facilitar el acceso a la información de websites corporativos mediante un interfaz vocal.

La arquitectura del sistema sigue el paradigma general de los sistemas de diálogo hablado comentado en la introducción de la tesis. A esta estructura se le ha incorporado un módulo denominado extractor del conocimiento de la tarea (*Task Knowledge Extraction*), encargado de aprender automáticamente el conocimiento relacionado con el dominio de la tarea utilizando los contenidos y estructura de un website dado. Para realizar esta función se utilizan técnicas de análisis de la estructura del website (obtención de la relación de enlaces entre las diferentes páginas), análisis de la estructura de la página web (se utilizan máquinas de soporte vectorial, SVM, para segmentar la página y clasificar los segmentos en 8 categorías predefinidas) y extracción de la información (se utilizan múltiples técnicas en paralelo).

El sistema utiliza el reconocedor AT&T Watson. El modelo de lenguaje combina términos clave dependientes de la tarea extraídos de la web y un modelo de lenguaje general desarrollado para tratar aspectos independientes de la tarea. El diseño del gestor de diálogo, basado en reglas, sigue una estrategia que puede resumirse en dos puntos:

1. Descomponer el flujo del diálogo en subdiálogos que sean independientes de la tarea y se hayan implementado a mano previamente.
2. Durante la interacción con el usuario, el gestor determina qué subdiálogo debe activarse y le suministra los parámetros que requiera como entrada. La decisión sobre qué subdiálogo invocar se toma a partir del contexto del diálogo y la representación semántica del último turno de usuario.

Las funcionalidades que puede desempeñar el gestor son: responder a peticiones del usuario independientes de la tarea (saludos y comandos); utilización del conocimiento de la tarea para responder a preguntas sobre la misma de forma precisa y concisa; aplicación de técnicas de respuesta a preguntas (*question answering*) para seleccionar fragmentos de la página web o de la estructura de subdirectorios; y proveer mensajes de ayuda cuando el sistema es incapaz de confeccionar una respuesta siguiendo los mecanismos anteriores.

El generador de respuestas está basado en la utilización de plantillas. Por último, el sistema utiliza el sintetizador AT&T TTS.

▪ **Center for Spoken Language Research de la Universidad de Colorado,** Boulder, Colorado (E.E.U.U.)

- CU Communicator: En (Ward y Pellom, 1999) se resume el sistema Communicator desarrollado por la Universidad de Colorado. El sistema sigue la arquitectura propuesta por DARPA, existiendo un servidor central que centraliza las comunicaciones y comportándose el resto de módulos del sistema como servidores. El ámbito del sistema es el acceso telefónico a información sobre vuelos, hoteles y coches de alquiler.

El sistema utiliza el reconocedor Sphinx-II con tres tipos de modelos acústicos (modelo de teléfono fijo independiente del usuario, modelo de teléfono fijo adaptado a mujeres, modelo adaptado a la telefonía móvil), transmitiéndose al módulo de comprensión la hipótesis más probable de las tres. Para construir el modelo de lenguaje de trigramas se utilizó el toolkit CMU-SLM (Statistical Language Modeling Toolkit) (Rosenfeld, 1995).

Como módulo de comprensión semántica se utilizó Phoenix. Recibe la secuencia de palabras del reconocedor y genera uno o más frames. El sistema genera medidas de confianza para las unidades semánticas, teniendo en cuenta tanto características de los modelos acústicos como del modelo de lenguaje.

La información del dominio del sistema se almacena en una base de datos SQL. Se desarrollaron scripts en Perl para la consulta de información en Internet y posterior actualización de la base de datos local. La generación de respuestas se basa en la utilización de plantillas. Para la síntesis de texto a voz se desarrolló un sintetizador propio dependiente del dominio y basado en la concatenación de unidades (fonemas, palabras y frases).

▪ **Spoken Language Processing Group (TLP) del LIMSI,** Orsay (Francia).

- *LIMSI RAITEL* (Lamel et al., 1997): El dominio del sistema desarrollado por el LIMSI dentro del proyecto RAILTEL (Railway Telephone Information Service) es proporcionar información de la red ferroviaria francesa (horarios, precios y servicios de trenes) por vía telefónica. Los principales componentes del sistema son:
 - Reconocedor automático del habla: Se trata de un reconocedor de habla continua independiente del locutor. El vocabulario de la tarea es de 1500 palabras, incluyendo 600 nombres de estaciones y de ciudades. Se utilizan modelos de lenguaje de n-gramas con backoff.

- La comprensión semántica se lleva a cabo en dos fases. En una primera fase se lleva a cabo una comprensión literal de la frase suministrada, en la que el analizador semántico utiliza una gramática *caseframe* para determinar el significado de la intervención del usuario representada por frames. La segunda fase consiste en una reinterpretación de la información semántica teniendo en cuenta la semántica de la tarea. Para ello, se utilizan una serie de reglas para convertir los valores cualitativos en cuantitativos y reinterpretar los frames semánticos obtenidos tras la primera fase. Existen dos tipos de reglas: las encargadas de suministrar valores por defecto no aportados por el usuario y las reglas interpretativas, que transforman valores literales en otros más apropiados para la gestión de diálogo.
- Gestión de diálogo. Se utiliza una iniciativa de gestión mixta basada en reglas. El diálogo se divide mediante una estructura jerárquica de subdiálogos, cada uno con una determinada funcionalidad dentro de la tarea o independientes de ella (metadiálogos). El diálogo se modela mediante un conjunto de reglas establecidas en forma de gramáticas, donde los símbolos no terminales se corresponden con subdiálogos y los terminales con actos de diálogo.

Se definieron diferentes tipos de subdiálogos: subdiálogo de apertura (iniciar el diálogo), subdiálogos de precisión (requerir información necesaria para la consulta), subdiálogos de explicación (utilizados para suministrar la información requerida por el usuario), subdiálogos de reformulación (solicitar la repetición del turno de usuario), subdiálogos de confirmación (confirmar las inconsistencias que detecta el sistema, por ejemplo, cuando la ciudad destino coincide con el origen) y subdiálogo de cierre.

La historia del diálogo se gestiona también mediante reglas, a partir de las cuales se decide qué atributos se utilizarán en la consulta a la base de datos y en qué condiciones se realizará su borrado.

- Generación de respuestas. El generador de respuestas se basa en una gramática en la que los símbolos no terminales están condicionados por el contexto del diálogo. Para generar la respuesta se utiliza la historia del diálogo y la información proporcionada por el módulo de consulta a la base de datos del sistema. En este último caso, la respuesta del sistema está influenciada por la cardinalidad de la información proporcionada.
- Sintetizador texto a voz. Se utiliza una combinación basada en la reproducción de audio pregrabado y en la síntesis por concatenación de unidades contenidas en un diccionario (nombres de ciudades, frases cortas, fechas, números, horas, etc.).

La evaluación del sistema se llevó a cabo utilizando 100 llamadas de usuarios no expertos, que debían cumplir un determinado escenario. Se utilizaron estadísticas tanto objetivas como subjetivas. El 72 % de los diálogos finalizaron con éxito.

- *LIMSI ARISE* (Lamel et al., 2000) (Lamel et al., 1999): El sistema desarrollado por el LIMSI dentro del proyecto LE-3 ARISE (Automatic Railway Information Systems for Europe) proporciona información telefónica sobre horarios, precios,

reserva de billetes, descuentos y servicios de trenes en Francia.

El sistema está basado en el trabajo desarrollado para el sistema RAILTEL. Las principales diferencias entre ambos sistemas se centran en la gestión del diálogo: en ARISE se utilizan medidas de confianza, se introduce la posibilidad de deletrear nombres de ciudades o estaciones, así como poder interrumpir las locuciones del sistema.

La arquitectura del sistema la forman seis módulos:

- El reconocedor automático del habla utiliza HMM con modelos acústicos basados en mixturas de gaussianas y modelos de lenguaje de n-gramas con backoff. El vocabulario está formado por 1800 palabras, incluyendo 500 nombres de estaciones.
- La comprensión semántica se lleva a cabo de idéntica forma a la comentada en el sistema RAILTEL.
- El gestor de diálogo desarrollado se basa en reglas, utilizándose la experiencia obtenida tras el análisis de diálogos persona-persona. Los objetivos definidos para la estrategia del diálogo son:
 - ◇ Impedir que el usuario pueda perderse durante el diálogo. Para ello, el sistema informa al usuario tras cada una de sus intervenciones sobre qué ha entendido exactamente.
 - ◇ Responder al usuario de la forma más precisa y concisa posible. Esta tarea se ha llevado a cabo en el contexto de la generación de respuestas.
 - ◇ Dar al usuario la oportunidad de corregir al sistema en cualquier momento del diálogo.
 - ◇ Evitar malas interpretaciones por parte del sistema, minimizando el número de errores del reconocedor.

Se desarrollaron dos iniciativas de control del diálogo: iniciativa mixta e iniciativa por parte del sistema. Ésta última se utiliza en los casos en los que no se obtiene el dato solicitado al usuario tras un número de repeticiones de la pregunta del sistema o cuando la información recibida del usuario es contradictoria con la entendida previamente (por ejemplo, tras solicitar la confirmación explícita de un determinado dato). En estos casos, el reconocedor utiliza un modelo de lenguaje dependiente del estado del diálogo.

- El módulo de acceso a la base de datos utiliza información de trenes real, almacenada estáticamente en local.
- El generador de respuestas tiene en cuenta la información más reciente proporcionada por el usuario para incorporarla en la nueva respuesta del sistema, de forma que el usuario pueda conocer exactamente la información que va entendiendo el sistema durante el diálogo.
- Se desarrolló un sintetizador texto a voz basado en la concatenación de unidades de diferente longitud almacenadas en un diccionario. El usuario puede interrumpir la locución del sistema.

En (Lamel, 1998) se recogen las estadísticas de las evaluaciones comparativas del sistema en sus diferentes fases de desarrollo. Se comparan los porcentajes

de éxito de los diferentes tipos de consultas, porcentaje de interrupción del usuario al sistema, tiempo de las llamadas, medida de errores de reconocimiento; así como la satisfacción del usuario evaluada a partir de un cuestionario de preguntas tras su interacción con el sistema.

- *PARIS-SITI (Système d'Informations Touristiques Interactif)* (Bonneau-Maynard y Devillers, 1998): Se trata de un sistema encargado de proporcionar información turística en francés de la ciudad de París (precios, horarios, direcciones, rutas, descripciones y servicios relativos a hoteles, restaurantes, cines, grandes almacenes, museos y monumentos). La arquitectura del sistema es muy similar a la presentada en los dos sistemas previos. El reconocedor dispone de un vocabulario de 2000 palabras y utiliza un modelo de lenguaje de bigramas. El gestor de diálogo utiliza también una iniciativa de control mixta y se desarrolló siguiendo una representación jerárquica del dominio del sistema utilizando cuatro niveles (objeto, clase, información y restricciones). Seguidamente se muestra un ejemplo de esta representación:

```
nivel-objeto =restaurante
nivel-clase = descripción, localización
nivel-información = especialidad, dirección
nivel - restricción =pescado
```

La estrategia del diálogo se organiza en tres pasos dependiendo de qué niveles contenga el frame de generación: Paso 0 (el frame contiene únicamente el nivel objeto, con lo que el sistema solicita los valores correspondientes al nivel clase), Paso 1 (el frame contiene únicamente el nivel clase, con el objetivo del diálogo es lograr que el usuario suministre sus atributos) y Paso 2 (cuando el usuario ya ha solicitado toda la información de una clase, el sistema conduce al usuario a realizar preguntas sobre las otras clases). En cada uno de estos pasos el usuario tiene la opción de no seguir la sugerencia del sistema y aportar cuánta información desee.

Para la evaluación del sistema se desarrollaron dos prototipos: uno en el que se seguía la estrategia de diálogo descrita y un segundo en el que el sistema respondía de forma directa a la intervención del usuario, eliminándose los prompts de sugerencia. La evaluación se llevó a cabo con 32 usuarios (16 expertos y 16 no familiarizados con el sistema) que adquirieron cuatro escenarios en cada prototipo. El porcentaje de fallo del diálogo fue del 4,6 % para el primer prototipo y del 17,2 % para el segundo.

■ **Laboratorios Bell**, (www.bell-labs.com)

- Bell Labs Communicator (Potamianos, Ammicht, y Fosler-Lussier, 2003): El principal objetivo del sistema Communicator desarrollado por Bell fue el diseño de una plataforma independiente del dominio y la modalidad, que facilitase el desarrollo rápido de sistemas de diálogo. El sistema consta de seis módulos: reconocedor automático del habla, módulo de comprensión semántica, módulo gestor de la pragmática, gestor de diálogo, generador de respuestas y sintetizador TTS.

- **Centre for Speech Technology Research en la Universidad de Edimburgo, Edimburgo (Reino Unido)**

- COMIC (Catizone, Setzer, y Wilks, 2003) fue un proyecto IST (2002-2005) cuyo dominio se centró en los ámbitos del teletrabajo y el comercio electrónico. Como base del proyecto se estableció la necesidad de utilizar servicios multimodales para modelar la conversación máquina-persona de una forma más semejante a la conversación que mantendrían varias personas entre sí. El principal objetivo de COMIC fue la definición de modelos cognoscitivos en los que se basase la interacción multimodal y realizar su evaluación en diversos escenarios.

Una de los sistemas desarrollados tenía como tarea el diseño de cuartos de baño, disponiendo de voz y gestos como modalidades de entrada y salida del sistema (se utiliza un avatar para generar las emociones de las cara).

- DIPPER (Bos et al., 2003) es una plataforma para el desarrollo de sistemas de diálogo hablados, basada en la aproximación de estados de la información (Sección 2.2.1), que utiliza la arquitectura OAA. La plataforma está compuesta por interfaces para realizar el reconocimiento del habla (Nuance), síntesis texto a voz (Festival, rVoice), gestión del diálogo, comprensión del habla y generación de respuestas.

- **Otros sistemas con participación internacional.**

- *SMARTKOM*: SMARTKOM (Alexandersson y Becker, 2001) es un sistema de diálogo multimodal que combina voz y gestos como modalidades de entrada y de salida. La arquitectura del sistema está conformada por:
 - Módulos interfaz: A la entrada existe un módulo de audio y a la salida, un gestor de visualización.
 - Reconocedores y sintetizadores: A la entrada del sistema existen módulos para el reconocimiento de voz, gestos y prosodia. A la salida, existe un sintetizador de voz y el gestor de visualización.
 - Módulos de procesamiento semántico: Este grupo de módulos realiza la comprensión y transformación de las representaciones semánticas (análisis de gestos y voz, reconocimiento de la intención del usuario, modelado del discurso y del dominio, planificador de acciones, transformación de conceptos a voz).
 - Servicios externos: bases de datos y módulos encargados de extraer información de la web.
 - *VERBMOBIL*: VERBMOBIL (Bos et al., 1999) es un proyecto interdisciplinar en el ámbito de las tecnologías del lenguaje en el que participaron 29 instituciones (universidades, empresas y centros tecnológicos) de Alemania, Japón y Estados Unidos. El sistema, independiente del usuario, reconoce la entrada hablada (habla espontánea), la analiza, traduce y sintetiza la traducción. VERBMOBIL se definió para la ayuda en diferentes dominios de diálogos multilingües (por ejemplo: programación de citas, planificación de viajes y reservas de hotel).
-

El proyecto, financiado con fondos públicos y privados, se subdividió en dos fases. El primer prototipo, *VERMOBIL Demonstrator*, se presentó en 1995. El prototipo reconocía un vocabulario de 1292 palabras en alemán en el dominio de la planificación de citas, las analizaba y traducía a inglés hablado. En 1997 se presentó un segundo prototipo que ampliaba el vocabulario a 2500 palabras en alemán y 400 en japonés. La finalidad de este prototipo es permitir el diálogo entre dos interlocutores que empleen lenguas diferentes (cada uno de los locutores se sitúa en un extremo del sistema, realizando éste el reconocimiento y traducción necesarias para posibilitar el diálogo).

En la segunda fase del proyecto, finalizada en el año 2000, se estableció como objetivo la ampliación del sistema para llevar a cabo traducciones bidireccionales alemán/inglés y alemán/japonés con vocabularios cercanos a 10000 palabras y utilizando la telefonía móvil como canal de comunicación.

- SENECA: La arquitectura del sistema de diálogo SENECA (Minker et al., 2004), desarrollado para su utilización en el entorno de automóviles, la constituyen cuatro módulos: el gestor de comandos, un módulo de comunicaciones GSM, un intercambiador de CDs y un módulo DSP, que realiza las operaciones relativas al procesado de la señal y del diálogo.

El sistema utiliza la tecnología Linguatronic (Heisterkamp, 2001), introducida por Mercedes-Benz en 1996 y desarrollada por Temic. Se trata de un sistema de diálogo hablado manos libres para el control del teléfono en el automóvil (indicar el número al que se desea llamar, almacenar números, editar agenda, etc.). El sistema incluye un reconocedor con modelos HMM independiente del usuario y un módulo de control (*Command & Control system*), encargado de la gestión de diálogo y almacenamiento de las respuestas vocales del sistema. Además, permite otras capacidades como el control de una autorradio, un intercambiador de CDs, un reproductor de cintas de cassette o el acceso a la información de un sistema de guiado GPS.

La finalidad de SENECA es proveer la interacción mediante la voz con el sistema de navegación COMAND. El procesado del diálogo utiliza una estrategia basada en menús con comandos accesibles en diferentes estados del diálogo (abortar, almacenar, borrar, ayuda...)

- AMITIÉS: El proyecto AMITIÉS (Automated Multilingual Interaction with Information and Services) (Hardy et al., 2006) está financiado por la Unión Europea y el programa estadounidense DARPA. Uno de los principales objetivos del proyecto es desarrollar un sistema de diálogo hablado que permita llevar a cabo mediante el habla las operaciones más corrientes en los portales de voz.

El dominio del sistema presentado en (Hardy et al., 2006) es funcionar como centro de atención de llamadas (*call center*) para operaciones bancarias. El prototipo desarrollado permite dos operaciones básicas: la identificación del usuario y la determinación de la razón por la que realiza la llamada. El sistema soporta iniciativa mixta y habla espontánea. Para la definición del sistema se ha analizado un corpus de 1000 diálogos grabados de este modo dentro del dominio financiero con el propósito de incorporar funcionalidades de las conversaciones

persona-persona.

La arquitectura del sistema coincide con la del Galaxy Communicator. A partir de los diálogos disponibles se han entrenado dos reconocedores (inglés de Reino Unido e inglés americano) utilizando herramientas de Nuance. El sintetizador utilizado es Nuance Vocalizer 3.0.

Un problema importante en el dominio del sistema consiste en realizar la correcta identificación del usuario. Para ello, se utiliza un estimador bayesiano que calcula la probabilidad de que el locutor coincida con alguno de los usuarios registrados en la base de datos dada la información que ha aportado. A partir de este estimador, se selecciona si es necesario preguntar de nuevo la información al usuario o si la identificación se asume como correcta.

Para la generación de respuestas se utiliza el software disponible en el Galaxy Communicator. La generación se basa en un esquema extensible de plantillas que disponen de cinco claves asociadas: tipo (indica de qué tipo de acto de diálogo se trata), repetición (indica el número de veces que este tipo de respuesta se ha generado durante el diálogo), atributos (lista de ítems necesarios para realizar la consulta solicitada), modificadores (descriptores opcionales que pueden acompañar/modificar a los atributos obligatorios), persona (campo opcional que indica el nombre del usuario que está interactuando con el sistema). Se desea que el sistema sea multilingüe (inglés, francés y alemán).

- *Companions* (www.companions-project.org): Companions (Intelligent, Persistent, Personalized Multimodal Interfaces to the Internet) es un proyecto IST del sexto programa. Se centra en el desarrollo de agentes conversacionales personificados (*Embodied Conversational Agents, ECA*) para el acceso de información en la web. Los desarrollos tecnológicos del proyecto se centran en la investigación de interfaces multimodales, tratando el proceso de comprensión semántica, el modelado del diálogo y la síntesis de voz. Entre los objetivos del proyecto se encuentra el desarrollo de un nuevo paradigma para las arquitecturas multimodales que esté basado en técnicas de aprendizaje automático.
- En (Juhar et al., 2006) se presenta un sistema de diálogo, basado en la arquitectura definida en DARPA Communicator, que proporciona información meteorológica y de horarios de tren en Eslovaquia. El sistema consta de seis módulos: módulo gestor de telefonía, reconocedor automático del habla (Sphinx-IV con modelos acústicos aprendidos a partir de las bases de datos SpeechDat-Sk y MobilDat-Sk), sintetizador texto a voz (se han desarrollado dos sintetizadores basándose en la concatenación de difonemas o de unidades de longitud variable), módulo de transporte (posibilita la conexión de los diferentes módulos cuando se utilicen simultáneamente los sistemas operativos Linux y Windows), gestor de diálogo (desarrollado siguiendo las especificaciones del estándar VoiceXML) y módulo de consulta de la información (obtiene la información mediante el acceso a websites).

En cuanto a los sistemas españoles, merece destacarse el sistema ATOS de Telefónica Investigación y Desarrollo (Álvarez et al., 1997), el sistema SAPLEN desarrollado por el grupo de Tratamiento del Habla del Departamento de Electrónica y Tecnología de

Computadores de la Universidad de Granada (López-Cózar et al., 1997) y los sistemas desarrollados en los proyectos de investigación BASURDE (Bonafonte et al., 2000) y DIHANA (Benedí, Varona, y Lleida, 2004). A continuación se resume un compendio de proyectos y sistemas de diálogo a nivel nacional:

- El sistema *ATTEMPS* (Hernando, 2002), desarrollado por la Universitat Politècnica de Catalunya, suministra información meteorológica en el territorio catalán mediante el uso del canal telefónico.

La información del sistema se obtiene en tiempo real, utilizando para ello un conjunto de estaciones meteorológicas con sensores que captan los diferentes parámetros (temperatura, lluvias, dirección y fuerza del viento, estado de la mar, etc.). La arquitectura está compuesta por cinco módulos: reconocimiento y comprensión del habla, control de diálogo, generación de habla, módulo de avisos y alarmas, y bases de datos meteorológicos y de usuario.

- El sistema *SAPLEN* (López-Cózar y Rubio, 1997) (Sistema Automático de Pedidos en Lenguaje Natural) tiene como dominio atender las peticiones y consultas formuladas por los clientes de un restaurante de comida rápida. Se define como un sistema experto basado en reglas y guiado por objetivos. La arquitectura del sistema se compone de los siguientes módulos: interfaz de entrada (recibe la secuencia de palabras del usuario en lenguaje natural y genera su interpretación semántica), módulo de control (activa los diferentes módulos teniendo en cuenta cada interacción del usuario), módulo de memoria (contiene la historia del diálogo), base de conocimiento de productos del restaurante, lexicon (almacena un conjunto de palabras clave dentro del dominio de la tarea) e interfaz de salida (genera la respuesta del sistema en lenguaje natural).
 - *Sistema DS-UCAT* (López-Cózar et al., 2006): Uno de los objetivos de investigación del proyecto UCAT (Ubiquitous Collaborative Training), en el que participan investigadores de la Universidad de Granada, es el desarrollo de un sistema de diálogo multimodal y multilingüe denominado DS-UCAT (Dialogue System for Ubiquitous Collaborative Training). La tarea de este sistema es proporcionar asistencia a profesores y estudiantes en algunas de sus actividades cotidianas dentro un centro educativo. Para ello, se han definido tres tipos posibles de entornos: biblioteca, despachos de profesores y aulas. El sistema utiliza una arquitectura modular, compuesta por un servidor de documentos XHTML+Voice al que acceden los dispositivos móviles de los usuarios (Tablet PCs, ordenadores portátiles o PDAs) mediante conexiones inalámbricas. La entrada y salida definida para el sistema es multimodal (sonidos, voz, gráficos y texto).
 - Investigadores de la Universidad de Vigo presentan en (Rodríguez et al., 2002) un sistema de diálogo en lenguaje natural para la consulta del correo electrónico. El objetivo del sistema es facilitar el acceso a un servidor de correo entrante para consultar si se han recibido mensajes y contar los mismos, consultar características de los mensajes (fecha, remitente, asunto), leer mensajes existentes o borrarlos del buzón.
-

La arquitectura del sistema la conforman un Reconocedor del Habla (basado en un modelo de reconocimiento que utiliza 25 modelos HMM preentrenados de fonemas), un Módulo de Síntesis de Texto a Voz (sintetizador Cotovía bilingüe castellano/gallego basado en la concatenación de unidades) y un Módulo de Control del Diálogo. Este último módulo consta de un Submódulo Analizador de Consultas y de un Submódulo de Gestión de Interacciones. El Gestor de Interacciones recibe como entrada la frase reconocida y la remite al Analizador de Consultas. El Analizador de Consultas recibe la frase reconocida y extrae mediante gramáticas los campos que caracterizan la consulta (acción a llevar a cabo, criterio de selección de mensajes, número de mensajes, remitente y fecha). A partir de la información generada por el Analizador de Consultas, el Gestor de Interacciones lleva a cabo la acción con el servidor de correo y obtiene la información requerida en la consulta, que es suministrada al sintetizador.

- En (Pérez-Piñar y García, 2005b) se introduce una variante a la arquitectura tradicional de reconocimiento de los sistemas de diálogo con la finalidad de detectar el tema expresado por el usuario y conocer con una mayor fiabilidad si éste ha cambiado. De este modo, el sistema propuesto incorpora dos módulos previos al gestor del diálogo: un módulo de reconocimiento en paralelo (formado por reconocedores adaptados cada uno de ellos a uno de los temas específicos que conforman el dominio de la aplicación) y un módulo de detección de tema y de cambio de intención. Los reconocedores adaptados generan las medidas de confianza que se utilizarán en el módulo de detección de intención, en el que se utilizan perceptrones multicapa (MLP) para generar un indicador de fiabilidad para cada tema. Finalmente, esta medida se envía al gestor del diálogo, que dispone así de la información sobre el tema correspondiente a la transcripción recibida del módulo de reconocimiento.
- En (Macías-Guarasa et al., 2004), investigadores de la Universidad Politécnica de Madrid describen el proyecto GEMINI, llevado a cabo en el periodo 2002-2004 y en el que se establecieron dos objetivos fundamentales: el desarrollo de aplicaciones de diálogo a partir de una descripción de una base de datos de forma semiautomática y la implementación de una plataforma en tiempo real para ejecutar los guiones obtenidos en el sistema de generación. Se trata de un proyecto que parte de los resultados obtenidos en proyectos anteriores llevados a cabo por la UPM, como IDAS (Lehtinen et al., 2000) (en el que se desarrolló un demostrador capaz de dar servicio de páginas blancas por teléfono en el que se proporcionaba información de números telefónicos de empresas y particulares), Servivox (Montero, 1998) (proyecto para la empresa Hewlett Packard para el desarrollo de un servicio telefónico automático de información para un centro de atención al cliente) y el proyecto SAITE (proyecto europeo para la creación de un servicio telefónico automático de información para una aplicación ferroviaria).

Los módulos que conforman la plataforma de desarrollo de aplicaciones de diálogo, utilizados para la implementación del sistema GEMINI, son:

- El módulo de reconocimiento de habla utiliza modelos HMM continuos entrenados con la base de datos SpeechDat. Se dispone de un módulo de cálculo de

medidas de confianza, que influyen en el tipo de confirmaciones que realizará el gestor de diálogo.

- El módulo de comprensión está basado en reglas dependientes de contexto. Se utiliza un único diccionario para la totalidad de *prompts* del sistema, etiquetado semánticamente en función de la información que se desea extraer.
- El módulo de modelado de usuario incluye información sobre la experiencia del usuario con el sistema, que modifica la información proporcionada por el sistema.
- Módulo de reconocimiento de idioma: Para realizar la detección del idioma se utiliza la técnica PPRLM, basada en modelar información de la secuencia de fonemas que se obtiene para cada uno de los idiomas utilizando un reconocedor de fonemas.

El resultado final es un guión del diálogo escrito en VoiceXML 2.0. Como intérprete de VoiceXML se seleccionó OpenVXI.

- El Grupo de Tecnología del Habla de la U.P.M. presenta en (Fernández et al., 2005) un interfaz conversacional que permite a los usuarios controlar un sistema Hifi mediante frases habladas de manera natural. La arquitectura del sistema la conforman un reconocedor de habla continua (vocabulario de 250 palabras), un módulo de comprensión (se utilizan un total de 70 categorías semánticas clasificadas en acciones, aparatos, parámetros y valores), módulo de gestión del diálogo (se han definido 20 posibles acciones a realizar sobre el sistema. Se utilizan redes bayesianas para identificar los objetivos del diálogo presentes en la frase del usuario. Para cada objetivo inferido, es posible detectar qué conceptos hace falta que mencione el usuario y cuáles son erróneos u opcionales. Como salida del módulo se generan tres marcos semánticos por cada acción: un aparato sujeto de la acción, un parámetro del aparato a controlar y un valor a adoptar por el parámetro indicado), un módulo de ejecución (recibe los marcos semánticos generados por el gestor del diálogo, interpreta las diferentes acciones y genera el conjunto de comandos que deben enviarse al sistema Hifi), módulo de generación de respuesta (informa al usuario sobre las acciones que va a recibir el sistema y le remite las preguntas que son necesarias para llevarlas a cabo) y módulo de conversión texto-voz (sintetiza los mensajes generados por el módulo anterior).
- *MIMUS*: MIMUS (Pérez, Amores, y Manchón, 2006) es un sistema multimodal y multilingüe, desarrollado por la Universidad de Sevilla, que permite controlar dispositivos del hogar mediante la voz y clicks del ratón. El sistema está preparado para la interacción con usuarios con discapacidades. La arquitectura la conforman los siguientes módulos, conectados utilizando la tecnología OAA:
 - Reconocedor automático del habla: Se utilizan los reconocedores desarrollados por Nuance y Verbio.
 - Sintetizador TTS: Se utiliza el sintetizador Festival y los agentes animados de Microsoft.

- Agentes GUI generales y de aplicación: Agentes, dependientes o no del dominio, que proveen diferentes tipos de salidas gráficas.
 - Gestor del Hogar: Proporciona un interfaz visual de los diferentes dispositivos que pueden controlarse.
 - Gestor de dispositivos: Realizan el control de los dispositivos físicos.
 - Gestor de conocimiento: Gestiona el acceso de los diferentes agentes a la información relativa a la tarea.
 - Gestor de diálogo: Es el núcleo del sistema. Está basado en reglas e interconectado al módulo de comprensión de lenguaje natural.
- En (González y Gatiús, 2005) se presenta un sistema de diálogo multimodal desarrollado para el proyecto europeo HOPS. El gestor del diálogo del sistema utiliza las ontologías para decidir la siguiente acción del sistema, así como para generar en las diferentes lenguas las gramáticas, léxicos y mensajes que intervendrán en cada interacción. La tarea del sistema es acceder al servicio de recogida de muebles del Ayuntamiento de Barcelona. La arquitectura del sistema está conformada por los siguientes módulos:
- Módulo de voz: Controla la interacción a través del teléfono. Sus componentes son un reconocedor, un intérprete VoiceXML y un sintetizador.
 - Módulo de texto: Controla la interacción a través de la web. Consiste en un servidor de texto y un analizador sintáctico-semántico.
 - Gestor de diálogo: Se trata de un gestor de propósito general, independiente del servicio concreto. Utiliza la información semántica generada por los módulos de texto y voz.
 - Gestor de ontologías: Envía al gestor de diálogo una representación del servicio basada en precondiciones, que son evaluadas por el gestor de diálogo para seleccionar el estado activado que posea una precondición más estricta.
 - Gestor de acciones y consultas: Accede a las ontologías y a las bases de datos de las aplicaciones de forma transparente para el gestor del diálogo.
 - Gestor de aplicaciones: Controla los recursos que requiere cada servicio implementado en el sistema.
- El proyecto *AVIVAVOZ* (Mariño, 2006) está dirigido a la investigación avanzada en todas las tecnologías clave que intervienen en un sistema de traducción de voz (reconocimiento, traducción y síntesis de voz). El objetivo del proyecto es posibilitar el desarrollo de sistemas que permita la intermediación oral entre personas en las lenguas oficiales del estado español (castellano, catalán, euskera y gallego) entre sí y entre el castellano y el inglés. Participan investigadores de la Universitat Politècnica de Catalunya, la Universidad de Vigo y la Universidad del País Vasco.
- *VOCALIZA* (Vaquero et al., 2006): Se trata de un sistema desarrollado por la Universidad de Zaragoza para su utilización por terapeutas en el tratamiento de diferentes patologías del habla y del lenguaje.
-

La aplicación dispone de diferentes funcionalidades para tratar tres niveles del habla (fonológico, semántico y sintáctico). Para ello consta de un reconocedor automático del habla, un sistema de síntesis de voz, un módulo de adaptación al usuario y un módulo de verificación del habla.

Uno de los objetivos del proyecto es la adquisición de un corpus de palabras y frases simples entre usuarios de edades comprendidas entre los 10 y 21 años con diferentes patologías.

- En (Uría et al., 2006) investigadores de las universidades de Zaragoza y del País Vasco presentan un servicio de mayordomo virtual para utilizar en el hogar con la finalidad de controlar electrodomésticos (horno, lavadora y lavavajillas) y conocer su estado a través de la voz. Está desarrollado para la empresa FAGOR. Cuenta con un sistema de diálogo que permite el habla espontánea y es independiente del usuario. El gestor del diálogo, basado en reglas, decide la respuesta más apropiada al resultado generado por los módulos de reconocimiento automático del habla y comprensión del lenguaje (también basado en reglas), y controla también la activación del dispositivo cuando ya se dispone de la información requerida. Este sistema se ha adaptado con el objetivo de proporcionar información meteorológica e información sobre noticias de prensa (sistema INMA).
- El objetivo principal del proyecto *BASURDE* (Bonafonte et al., 2000) fue el desarrollo de un sistema de diálogo encargado de atender consultas telefónicas sobre horarios, precios y servicios de trenes españoles de largo recorrido. En el proyecto participaron la Universidad del País Vasco, la Universitat Jaume I de Castelló, la Universitat Politècnica de Catalunya, la Universidad de Zaragoza y la Universitat Politècnica de València.

En el proyecto se adquirió un corpus de 227 diálogos usando la técnica del Mago de Oz. En esta adquisición participaron 75 informantes y cada uno completó 3 escenarios del conjunto definido para la adquisición. El etiquetado del corpus se llevó a cabo utilizando el concepto de acto de diálogo. Los actos de diálogo de usuario se corresponden con la interpretación semántica de la intervención del usuario en base a frames (17 conceptos y 30 atributos). Los turnos de sistema vienen etiquetados en base a tres niveles: un primer nivel independiente de la tarea que describe la función genérica del turno, un segundo nivel específico de la tarea en el que se referencian los conceptos existentes y un tercer nivel en el que se incluyen los valores de los atributos que reciben valores en el turno de usuario.

- El objetivo principal marcado en el proyecto *DIHANA* (Benedí, Varona, y Lleida, 2004) (proyecto continuación del proyecto *BASURDE*), en el cual se engloba la tesis doctoral que se presenta, fue el estudio y desarrollo de un sistema robusto de diálogo modular y distribuido para el acceso a sistemas de información. En concreto, se profundizó en aspectos metodológicos fundamentales en los campos del modelado acústico en diferentes entornos, tratamiento del habla espontánea, modelado del lenguaje, comprensión y diálogo. En el proyecto participaron la Universidad del País Vasco, la Universidad de Zaragoza y la Universitat Politècnica de València.
-

Para el desarrollo de un prototipo de sistema de diálogo en DIHANA se pensó en una arquitectura distribuida formada por siete componentes: un servidor de audio, un servidor de reconocimiento automático del habla, un servidor de comprensión del habla, un servidor de gestión del diálogo, un servidor de generación de respuesta oral, un servidor de conversión texto-voz y finalmente por un cliente gestor de comunicaciones. El gestor de comunicaciones está encargado de establecer todas las comunicaciones con el resto de módulos.

- Proyecto *EDECÁN* (Lleida et al., 2006): El objetivo del proyecto EDECÁN es aumentar la robustez de un sistema de diálogo de habla espontánea a través del desarrollo de metodologías para la adaptación y personalización a diferentes contextos acústicos o dominios de aplicación. Para alcanzar este objetivo, se propone el desarrollo de estrategias que permitan caracterizar las condiciones de funcionamiento del sistema de diálogo (acústicas, contexto semántico de la tarea, tipo de usuario...) y la definición de técnicas de adaptaciones a dichas condiciones. Otros objetivos del proyecto son la definición de medidas de evaluación adaptadas a las condiciones expuestas en el proyecto y abordar la extensión del sistema de diálogo para permitir al usuario la consecución de múltiples objetivos en el transcurso del diálogo. Se proponen diferentes tareas (domótica, información y reserva de pistas deportivas...) sobre diferentes contextos acústicos (hogar, coche, kiosco de información).

2.2. Gestión del diálogo

Teniendo en cuenta el método utilizado para representar e implementar el flujo del diálogo, pueden distinguirse diferentes tipos de gestión del diálogo, que permiten clasificar a los sistemas de diálogo que los utilizan (Allen et al., 2001) (McTear, 2004). De menor a mayor complejidad se distinguen:

- **Sistemas de estados finitos.**

En los sistemas de estados finitos el flujo del diálogo puede determinarse con anterioridad y representarse por medio de una red o gramática. Utilizando la representación del diálogo mediante una red de transiciones entre estados, los nodos representan las preguntas del sistema y las transiciones determinan el conjunto de caminos que pueden establecerse en la red, de modo que la interacción está completamente estructurada: el gestor de diálogo se desplaza a través de la red, en la que se indica que información debe ser intercambiada en cada estado del diálogo, y obtiene la información del usuario necesaria para llevar a cabo una determinada tarea.

La principal ventaja de esta aproximación es su simplicidad, facilitando el desarrollo de gestores cuando la tarea sea muy sencilla que estén claramente estructuradas y en las que existan un número pequeño de tipos de respuestas. Los principales inconvenientes residen en el hecho de no ser adecuados para gestionar diálogos complejos y su falta de flexibilidad, dado que los usuarios no pueden desviarse de los caminos establecidos para cada estado. El diseño del gestor de diálogo requiere una labor intensiva, específica para cada dominio, y basada en la depuración de errores conforme se van detectando, ya que el control del flujo del diálogo se determina de forma manual.

El CLSU (*Center for Spoken Language Understanding*) desarrolló una herramienta gráfica, CSLU Toolkit (McTear, 1998), para el diseño de sistemas de diálogo siguiendo esta aproximación. Otra herramienta para la construcción y minimización de máquinas de estados finitos es la librería AT&T FSM, de funcionamiento en Linux.

Las utilización de gramáticas se basa en realizar un análisis sintáctico del diálogo. Para su implementación suelen utilizarse reglas gramaticales y diversos tipos de máquinas de estados. Diferentes trabajos relativos al uso de gramáticas para representar el diálogo son el sistema SUNDIAL (Müller y Runge, 1993) y SUNSTAR (Nielsen y Baekgaard, 1992).

El gestor de diálogo del sistema TOOT utiliza una máquina de estados finitos para controlar la interacción, basada en el estado actual del sistema y los resultados suministrados por el reconocedor. El gestor consta de 168 estados, cada uno de los cuales está asociado con una de las 12 gramáticas definidas, que especifican el modelo de lenguaje del reconocedor en ese punto del diálogo.

■ Sistemas basados en frames.

El objetivo de esta aproximación es solucionar la falta de flexibilidad de los modelos de estados finitos. Se asemejan a esta aproximación en que ambas son capaces de gestionar tareas basadas en completar un formulario de datos solicitando información al usuario para posteriormente realizar una consulta a una fuente de conocimiento externa (*form-filling tasks*). La diferencia estriba en la no necesidad de seguir un orden preestablecido para completar los diferentes campos requeridos, de forma que se le dote al sistema de iniciativa mixta. Para permitir este grado de flexibilidad mayor, es necesario dotar al sistema de tres componentes:

- Un frame que haga referencia a los conceptos y atributos definidos para la tarea.
- Una gramática o modelo de lenguaje para el reconocedor más extensa.
- Un algoritmo de control del diálogo que determine las próximas acciones del sistema basándose en los contenidos del frame.

En la definición del frame pueden considerarse valores adicionales a la simple anotación de si el campo (slot) posee o no un valor, por ejemplo, utilizar las medidas de confianza para hacer referencia a la fiabilidad del dato almacenado. En (Goddeau et al., 1996) se presenta un gestor de diálogo en el dominio de anuncios de compra-venta de coches usados en el que se extiende esta idea. La variante de frame definida, denominada E-form (electronic form), incluye además información sobre las preferencias del usuario en forma de prioridades. Estos formularios se utilizan en la gestión de diálogo desarrollada en el sistema Bell Labs Communicator.

Ejemplos de utilización de la aproximación basada en frames son el sistema JUPITER, ARISE, WITAS, COMIC, etc.

Una alternativa en auge para la elaboración de estrategias de gestión basadas en la definición de reglas, es el lenguaje estándar VoiceXML (McGlashan et al., 2004).

La investigación en sistemas de diálogo ha coincidido en el tiempo con el desarrollo de la red Internet y las aplicaciones Web y, en consecuencia, se ha planteado el acceso a los sistemas de diálogo mediante aplicaciones Web. Se puede entender un documento VoiceXML como una máquina de estados finitos que va recorriendo el usuario durante el diálogo.

El estándar VoiceXML es el resultado del trabajo conjunto de varias importantes compañías (AT&T, IBM, Lucent, Motorola, etc) que componen el denominado VoiceXML Forum. VoiceXML ha sido diseñado para crear diálogos con audio, posibilitando la síntesis, el reconocimiento y la grabación del habla, así como el desarrollo de conversaciones con iniciativa mixta. A todas estas funcionalidades, VoiceXML aporta las ventajas del desarrollo basado en las aplicaciones Web.

El gestor de diálogo Florence (Fabrizio y Lewis, 2004) desarrollado por los laboratorios de AT&T soporta iniciativa mixta, diferentes estrategias para la confirmación y técnicas para la corrección de errores. Basado en el estándar VoiceXML, se trata de una arquitectura para el desarrollo rápido de nuevos servicios de diálogo, con los objetivos de simplificar el desarrollo de las aplicaciones, facilitar la reusabilidad y modularidad del sistema, y cumplir estándares existentes como SALT, estándar de interacción multimodal definido por el W3C y X+V. El formato de comunicación definido se basa en paquetes XML. Existen dos tipos de fichero: ficheros en los que se fija la estrategia del diálogo y ficheros de configuración de la aplicación (almacenan parámetros que caracterizan a la aplicación como el diálogo inicial, módulo procesador de salida a utilizar, etc.).

En VoiceXML existen dos estrategias para llevar a cabo la gestión del diálogo. En la estrategia ATN (*Augmented Transition Networks*), el flujo del diálogo se representa mediante un conjunto de estados, transiciones, condiciones y variables. Se selecciona una transición determinada en un estado específico cuando se cumplen las condiciones así establecidas y se han llevado a cabo las acciones prefijadas. La segunda modelización, denominada *Clarificación FC*, define la estrategia del diálogo en forma de árbol jerárquico. El árbol incluye condiciones que describen categorías y temas, y mensajes (*prompts*) a utilizar para obtener información del usuario

En cuanto a los sistemas que utilizan modelos basados en reglas, destaca también el sistema ARISE (Automatic Railway Information Systems for Europe). En el prototipo desarrollado por el LIMSI, el gestor de diálogo pregunta al usuario por datos, necesarios y aún no facilitados, o bien utiliza el frame dado por comprensión, junto con una serie de reglas que gestionan la historia del diálogo, para generar una consulta SQL a la base de datos. En ambos casos, el gestor de diálogo produce un acto de diálogo en forma de frame.

■ Sistemas basados en planes.

Los sistemas basados en planes consideran que los seres humanos se comunican para conseguir objetivos. Las intervenciones del usuario suelen representarse como actos de diálogos (Searle, 1969) y se utilizan para alcanzar estas metas. Las teorías de modelado del diálogo basadas en planes (Allen y Perault, 1980), (Appelt, 1985), (Cohen y Levesque, 1990) argumentan que los actos de diálogo del usuario forman parte de un plan u objetivo global, que el sistema tiene que identificar y responder

en función de él de la forma más adecuada. Las principales críticas que se realizan a esta aproximación se basan en la dificultad de realizar el reconocimiento del plan del usuario. Un ejemplo de esta aproximación es la librería ATLAS.

La Teoría de los Juegos Conversacionales (Kowtko, Isard, y Doherty, 1991) utiliza técnicas mixtas correspondientes a las gramáticas de discurso y a la aproximación de sistemas basados en planes. Esta teoría se ha utilizado para la representación de modelos de diálogo orientados a la tarea (Williams, 1996), en los que el diálogo consiste en una o más transacciones definidas como subtareas.

Otra variante, denominada aproximación colaborativa, se basa en la visión del diálogo como un proceso en el que el usuario y la máquina deben trabajar conjuntamente para conseguir la comprensión mutua del diálogo. Este tipo de aproximación trata de capturar el conjunto de motivaciones que se producen durante el diálogo, en lugar de centrarse en la estructura de la tarea. Un ejemplo de utilización de esta aproximación es el gestor de diálogo del sistema TRAINS-93.

■ **Sistemas basados en agentes.**

En los sistemas basados en agentes el flujo del diálogo se determina de forma dinámica a través de un proceso en el que el gestor de diálogo lleva a cabo un cierto razonamiento para determinar las próximas acciones. Estas aproximaciones utilizan técnicas de Inteligencia Artificial y orientan el modelado del diálogo como una colaboración entre agentes inteligentes para solucionar un determinado problema o tarea. Son adecuadas para la gestión del diálogo en tareas complejas, como las negociaciones y la resolución de problemas.

La comunicación en este tipo de aproximación puede verse como la interacción entre dos agentes, cada uno de los cuales dispone de su propio conjunto de acciones. Estas aproximaciones incluyen normalmente mecanismos para la detección y corrección de errores, definen estructuras para tener en cuenta la historia del diálogo y se basan usualmente en iniciativas mixtas para el control del diálogo.

Ejemplos de estas aproximaciones son el gestor Agenda desarrollado por la CMU y la arquitectura Ravenclaw, Queen's Communicator, SesaMe y la arquitectura JASPIS. En estas aproximaciones se utilizan aproximaciones orientadas a objetos y se intenta separar la gestión genérica del diálogo de las acciones específicas del dominio.

2.2.1. Resumen de metodologías de gestión relevantes

A continuación se resumen brevemente un compendio de aproximaciones relevantes en el ámbito de la gestión del diálogo, que siguen los diferentes métodos anteriormente citados:

- Los responsables del ARISE-NL discuten varios aspectos relativos a la estrategia de diálogo como resultado de las experiencias con los primeros prototipos del sistema. En cuanto a modelos de diálogo, en (Sturm, den Os, y Boves, 1999a) se presentan distintas técnicas para obtener modelos de lenguaje adecuados para los sistemas de diálogo, teniendo en cuenta aspectos como el tamaño reducido de los corpus de entrenamiento o como las restricciones disponibles en un sistema de diálogo. Se

plantean las siguientes alternativas: modelos de segmentos de conceptos, modelos dependientes de estados de diálogo y modelos iniciales sintéticos.

- Gran parte de las funciones que suele realizar el gestor del diálogo se transfieren en el sistema JUPITER (Zue et al., 2000) al hub programable a través del cual se conectan todos los módulos siguiendo la arquitectura GALAXY II. Para dotarlo de cierta independencia respecto a la tarea, el comportamiento de este módulo se ajusta por medio de una tabla de control de diálogo, externa a su código. En dicha tabla se especifican los conjuntos de condiciones necesarias para llevar a cabo ciertas acciones; y las secuencias de operaciones que se activarán si se cumple un conjunto determinado de condiciones.
- El gestor de diálogo integrado en el Carnegie Mellon Communicator (Rudnicky et al., 1999) se ha diseñado como un módulo independiente de la tarea y cuyo comportamiento se fija mediante un script dependiente de la tarea, además de la acción complementaria de tres agentes de dominio (agente de información de viajes, agente de interpretación de fechas y horas, y agente de modelo de usuario). Se trata de un gestor de diálogo basado en scripts, que determinan una secuencia explícita de conceptos propios de la tarea. Se permiten iniciativas de sistema e iniciativas mixtas. En cuanto a los modelos de diálogo se consideran dos aproximaciones: sistemas basados en grafos (se enumeran los posibles estados de diálogo y las transiciones permitidas) y sistemas basados en formularios (el usuario tiene que ir rellenando campos de información).

La gestión de diálogo se detalla como un problema complejo que no puede reducirse a ejecutar una secuencia de pasos predeterminados, ni formularse como un objetivo de completar un formulario, dado que el viaje se enfoca como una situación negociable durante el diálogo y la estructura del formulario no se conoce a priori.

Uno de los aspectos claves de la gestión son los cambios de consulta que puede realizar el usuario durante el transcurso del diálogo y que marcan la impredecibilidad del diálogo. De este modo, aunque el sistema tiene el objetivo de ir solicitando la información requerida para el dominio de los vuelos y dispone de preguntas específicas para solicitar los diferentes datos, la estructura de la actividad que debe realizar el gestor no está predeterminada. Por ello, en lugar de disponer de un formulario que contiene los slots a rellenar, el gestor del CMU Communicator representa el diálogo como un árbol en el que pueden añadirse o modificarse ramas de forma dinámica, representando el itinerario que va a seguir el diálogo adaptándose a los nuevos requerimientos del usuario

- El gestor de diálogo desarrollado para el CU Communicator (Ward y Pellom, 1999) está dirigido por sucesos que afectan al cambio del contexto del diálogo (*event driven dialog manager*). No se utiliza un guión (script) explícito, como en los gestores de diálogo de estados finitos, sino se define un conjunto jerárquico de frames que representan la información que pueden intercambiar el usuario y el sistema como base para decidir qué acción realizar a continuación en el diálogo. El objetivo del diálogo es completar los campos (slots) asociados al frame. Un ejemplo de los frames se muestra a continuación:

```
Frame:Air_Travel
[Travel_Time]
  [Depart_Time]
    [Start_Time]
    [End_Time]
  [Arrive_Time]
    [Start_Time]
    [End_Time]
```

La ontología del dominio se define en un fichero de tarea que contiene información de los frames, plantillas para respuestas del sistema, verificaciones e información para generar consultas SQL. Los usuarios pueden proveer la información en cualquier orden y el sistema se dedica a ir solicitando los campos obligatorios hasta completar el frame. El procesado de la entrada suministrada por el módulo de compresión se lleva a cabo en las siguientes etapas:

- Obtener los valores canónicos de los atributos (como, por ejemplo, resolver fechas relativas como “mañana”).
- Resolver la elipsis (por ejemplo, determinar si el valor suministrado de una ciudad se corresponde con el origen o el destino).
- Ejecutar funciones contextuales (por ejemplo, para determinar si el valor de una hora se corresponde al intervalo AM o PM).
- Crear la estructura de frames final teniendo en cuenta la información contextual.

Una vez se ha extraído la información semántica del turno de usuario y se ha contextualizado, el sistema decide qué acción realizar basándose en un conjunto de prioridades predefinido:

- Comprobar si existen acciones globales definidas para el sistema (Colgar, Reiniciar, Repetir...).
 - Comprobar si existe algún mensaje del sistema pendiente de reproducción (por ejemplo, para confirmar una información aportada previamente por el usuario).
 - Comprobar si existe información pendiente de notificar al usuario.
 - Comprobar si deben generarse consultas a la base de datos.
 - Comprobar si el frame actual está completo, sino solicitar los campos pendientes. Si existen slots obligatorios para la consulta actual que no poseen valor, el sistema pregunta por el slot marcado con mayor prioridad para el frame actual.
 - Comprobar si existen otros frames pendientes de completar. Si para el frame actual poseen valor todos sus campos, se revisa si existe información del contexto que es necesario solicitar.
 - Decidir que debe realizarse el cierre del diálogo.
 - Finalizar el diálogo si el usuario así lo solicita.
-

Utilizando el fichero de la tarea, los frames definidos y este conjunto de prioridades, el gestor de diálogo puede generar diálogos flexibles y naturales sin necesidad de ser definidos en forma de estados finitos.

En cuanto a la iniciativa del diálogo, tal y como se ha descrito, el sistema se basa en la utilización de una iniciativa mixta: requiere información específica al usuario, pero éste puede aportar toda aquella información adicional que desee o cambiar el frame actual del sistema.

- El modelo de iniciativa mixta para la gestión de diálogo desarrollado para AT&T Communicator se formaliza mediante un proceso de decisión secuencial. Este modelo se basa en la definición del estado del diálogo y de las acciones del diálogo, ya sean con el usuario, las bases de datos o el módulo de temporización. El estado del diálogo representa el conocimiento (los valores de las variables relevantes) que el gestor de diálogo tiene almacenado en determinado instante del diálogo para determinar la próxima acción. El algoritmo que sigue el gestor de diálogo de acuerdo con el proceso de decisión secuencial descrito se resume en los siguientes pasos:

```
Inicialización: Seleccionar el estado inicial
Iterar hasta alcanzar el estado final:
  Próxima acción: Seleccionar y ejecutar la acción marcada como siguiente
  Obtener la nueva entrada
  Siguiente estado: Actualizar el estado en función de la nueva entrada
```

En la estrategia implementada para el gestor consiste en:

- Etapa de identificación: El usuario proporciona su número personal (PIN). A partir de esta identificación se consulta el perfil del usuario: preferencias sobre destinos, compañías de aéreas o de alquiler de coche, hoteles, opciones de comida y asientos preferidos y todas las demás características que el sistema puede instanciar. Además, el usuario puede identificarse como “invitado”.
- Planificación de vuelos: Se lleva a cabo en dos etapas:
 - Solicitud de información: El sistema va solicitando al usuario la información marcada como obligatoria, siguiéndose siempre una iniciativa mixta.
 - Presentación de la información y negociación: Disponiendo de la información necesaria, el sistema realiza consultas a la base de datos e informa al usuario con mensajes de espera para los casos en los que el temporizador así lo indique (demoras en las consultas). El sistema puede realizar cambios en las preferencias indicadas por el usuario para sugerirle alternativas en caso de no encontrar vuelos que cumplan los requerimientos marcados. Asimismo, puede proporcionar información sobre el itinerario del vuelo. La estrategia para tratar consultas sobre reservas de hoteles o de coches es similar a la de reserva de vuelos. La información solicitada al usuario es el nombre del hotel preferido, localización, nombre de la compañía de alquiler de coches y tipo de coche. Si el usuario no expresa preferencias, el sistema toma la iniciativa y le sugiere alternativas. Al final de la interacción se requiere información al usuario sobre si fue capaz o no de resolver satisfactoriamente sus consultas.

De este modo, la estrategia de diálogo consiste en desglosar una tarea compleja en subtareas, cada una con un objetivo específico y llevada a cabo mediante subdiálogos que mantienen un esquema común (recopilación de datos, presentación de resultados y posible negociación).

En cuanto al uso de la historia del diálogo, en la arquitectura AMICA (Pieraccini, Levin, y Eckert, 1997), el curso de diálogo depende de la historia (*dialogue state*), que almacena la información disponible desde el inicio del diálogo en una estructura de pares nombre, valor, y el estado (*control state*), que identifica una situación en el diagrama de flujo que regula el diálogo. En (Levin y Pieraccini, 1997), la representación del estado del sistema se estructura en tres plantillas que reflejan la historia del diálogo:

- Plantilla de usuario (*user template*). Representa el significado de la petición del usuario, interpretada en su contexto. Consta de una lista de pares {nombre, valor} de todos los atributos.
 - Plantilla de datos (*data template*). Describe los datos extraídos de la base de datos, conforme a una consulta basada en la plantilla de usuario. Consta de un cierto número de tuplas de datos.
 - Plantilla del sistema (*system template*). Representa la historia de acciones del sistema durante el diálogo.
- El gestor de diálogo del sistema Queen's Communicator (O'Neill et al., 2003) utiliza técnicas orientadas a objetos para modelar tanto estrategias del diálogo genéricas, como aquellas que son específicas del dominio. El gestor del discurso (*DiscourseManager*) determina el comportamiento genérico del diálogo en respuesta a las intervenciones del usuario. Para ello, existe un conjunto de módulos expertos específicos del dominio que poseen un conjunto de reglas para gestionar el diálogo.

Para decidir la acción a llevar a cabo, el gestor del discurso realiza consultas a un módulo, *DiscourseHistory*, que almacena un registro de la información del diálogo en forma de una pila de frames (*DialogFrames*), cada uno de los cuales con un conjunto de atributos asociado. Estos frames son similares a los utilizados en el CU Communicator, dado que representan los slots que deben completarse para llevar a cabo una determinada transacción. Sin embargo, los slots definidos contienen información adicional:

- El estado de confirmación de los atributos. Se definen un conjunto de estados como "atributo nuevo", "inferido por el sistema", "repetido por el usuario", "modificado por el usuario", etc.
 - Medida de discurso que asegura que cada atributo se ha confirmado adecuadamente antes de utilizarse en la transacción. El usuario debe repetir una vez el valor del atributo o confirmarlo explícitamente para que el sistema considere que está correctamente confirmado, y poder así utilizarlo en las respuestas del sistemas.
-

A partir del estado de confirmación y la medida de discurso, el sistema selecciona la acción más adecuada (confirmación de valores, solicitud de un atributo negado, pregunta del valor, etc.).

- En los sistemas desarrollados en PHILIPS (Aust et al., 1995), el gestor de diálogo sigue una estrategia consistente en completar los datos necesarios para la consulta (*slot-filling strategy*). Al tratarse de sistemas con iniciativa mixta, el usuario puede proporcionar valores para varios slots, con independencia de que éstos figurasen en el precedente prompt del sistema. La tarea del gestor de diálogo es rellenar los slots necesarios para poder satisfacer la petición del usuario, procurando que el diálogo sea lo más breve posible. Para facilitar la adaptación a otras aplicaciones, se ha desarrollado HDDL, un lenguaje de descripción de diálogos que permite especificar las definiciones de los slots, las preguntas y las estrategias de confirmación.
- En (Macherey y Ney, 2003), el gestor de diálogo que se propone es independiente del dominio y se basa en el uso de estructuras de datos arborescentes, que representan el conocimiento asociado a una tarea específica. De este modo, las acciones del gestor de diálogo se pueden definir como operaciones sobre árboles, con cierta independencia del dominio de la tarea. A partir del árbol original se generan árboles específicos, donde se almacenan los pares de conceptos y atributos dados en la entrada del usuario. En caso de que se disponga de varios caminos válidos para continuar un diálogo, la decisión sobre la siguiente acción de diálogo se toma optimizando una función de coste que tiene en cuenta diversos indicadores.
- El gestor de diálogo desarrollado para el proyecto AMITIÉS (Hardy, Strzalkowski, y Wu, 2003) comparte la filosofía del gestor de CU Communicator. Entre las tareas que se incluyen en el sistema, se encuentra la verificación de la identidad del cliente, la identificación de la operación que desea llevar a cabo y la ejecución de la transacción (obtener un resumen de las operaciones de saldos u operaciones bancarias, informar de un robo o pérdida de tarjetas, cambio de datos, etc.). Cada una de estas tareas es tratada por un agente específico (*Frame Agent*, FA).

Para completar una tarea es necesario identificarla y requerir la información que sea necesaria para llevarla a cabo (*frame-filling*). El módulo de identificación de la tarea está modelado por un corpus de 500 diálogos grabados en un centro de atención de llamadas. Analizando estas conversaciones se detectaron manualmente 14 tareas distintas. La identificación de la tarea se trata como un problema de clasificación.

El segundo submódulo del gestor de diálogo es un clasificador de actos de diálogo. La finalidad de este clasificador es identificar el turno de usuario como uno o más actos de diálogo independientes del dominio: Aceptar, Rechazar, No-Entendido, Apertura, Cierre, Backchannel y Expresión.

Para gestionar el diálogo se almacenan dos historiales: un historial en el que se guarda la tarea actual y el conjunto de tareas que han sido ejecutadas previamente (satisfactoriamente o no); y un historial que almacena los atributos que se requieren para la tarea actual y el estado de los mismos.

- Dentro del proyecto BASURDE se propone una modelización estocástica de la gestión del diálogo (Torres, Sanchis, y Segarra, 2003) (Torres, 2006). A partir del cor-

pus BASURDE etiquetado se aprendieron dos modelos estocásticos de diálogo. En el aprendizaje de estos modelos se utiliza únicamente el etiquetado de primer y segundo nivel definido para etiquetar el corpus. La información correspondiente al tercer nivel de etiquetado (valores de atributos) se almacena en el registro histórico del diálogo. Los dos modelos, denominados modelo principal y modelo auxiliar, son modelos de bigramas, diferenciándose en el nivel de etiquetado que identifica sus estados. En el modelo principal, los estados se identifican por una concatenación de una o más cadenas tipo (Turno:Primer-Nivel:SegundoNivel). En el modelo auxiliar, los estados se identifican por una concatenación de una o más cadenas tipo (Turno:PrimerNivel). El modelo principal tiene 155 estados y el modelo auxiliar tiene 33 estados. El modelo auxiliar es independiente de la tarea y, por ello, el gestor de diálogo no lo utiliza para decidir su estrategia mientras el modelo principal le proporcione transiciones compatibles con el diálogo en curso.

Para resolver el problema de las situaciones no vistas durante la gestión del diálogo, se aplica un procesamiento de los frames de usuario antes de explorar las transiciones del modelo. Este preprocesamiento de los frames, al que se denomina generalización semántica, consiste en generar un conjunto de identificadores de actos de diálogo (muy similares al acto de diálogo correspondiente a los frames de usuario) y usar todo el conjunto, en lugar de un solo identificador, en la posterior fase de búsqueda y comparación con los estados del modelo.

2.2.2. Teoría de los Estados de la Información

El proyecto Trindi (www.ling.gu.se/projekt/trindi) (TRINDI Consortium, 2001) propone una arquitectura para el desarrollo de gestores de diálogo basada en el concepto de estado de la información (*information state*) (Traum et al., 1999). La aproximación de estados de información intenta combinar los puntos fuertes de los modelos de diálogo basados en estados y los basados en planes, utilizando aspectos del modelado del diálogo mediante estados e incluyendo representación semántica detallada y nociones de obligaciones, compromisos, creencias y planes. La teoría del modelado del diálogo basada en el concepto de estado de la información se caracteriza por los siguientes conceptos:

- Una descripción de las componentes de la información (*informational components*), que incluyen el contexto y factores de motivación internos (intenciones, creencias, compromisos, etc.).
 - Representaciones formales de las componentes anteriores (como listas, conjuntos, vectores de características, etc.).
 - Un conjunto de acciones o cambios en el diálogo que provocarán la actualización del estado de información.
 - Un conjunto de reglas de actualización que gobiernan las transiciones entre estados de información cuando se cumplan condiciones preestablecidas en el estado de información actual y se hayan llevado a cabo acciones previas durante el diálogo.
 - Una estrategia de actualización para decidir qué regla (o reglas) se selecciona del conjunto de reglas aplicables en un determinado instante del diálogo.
-

El estado de la información hace referencia a toda la información necesaria para distinguir un diálogo de otros diálogos, incluyendo una representación de la historia de las acciones previas llevadas a cabo durante el diálogo, utilizada para decidir la nueva acción que realizará el sistema. Se trata de un concepto muy abstracto, instanciado muchas veces como modelo mental, contexto del diálogo, medida conversacional, etc.

Cabe realizar una distinción entre las aproximaciones basadas en el estado de la información y las aproximaciones basadas en el estado del diálogo. En éstas últimas, el diálogo se comporta de acuerdo a una gramática en la que los estados representan los resultados de las acciones llevadas a cabo en el estado anterior y cada estado dispone de un conjunto de transiciones o acciones permitidas. La información es implícita únicamente del estado y de la relación que pueda tener con los otros estados.

TrindiKit (Larsson et al., 1999) es la primera versión completa de la aproximación del estado de la información. Escrita en Prolog, modela el sistema de diálogo definiendo estados de la información, reglas de actualización y selección, y algoritmos de control que gobiernan las reglas a aplicar en el estado de información. Los estados de información se definen usualmente como una estructura recursiva de la forma *Nombre:Tipo*, donde Nombre es un identificador y Tipo es un tipo específico de datos. Seguidamente se muestra un ejemplo (Larsson et al., 1999), en el que el estado se representa mediante un registro con nombre *is* que contiene tres campos (*grammar*, *queue*, *stack*) de diferentes tipos de datos:

```
is:record([grammar:atomic,
input:queue(atomic),
sem:stack(record([int:atomic,
content:drs]))])])
```

Ejemplos de gestores de diálogo desarrollados utilizando este paradigma son los implementados para los proyectos WITAS y COMIC. El gestor de diálogo desarrollado para WITAS se basa en la utilización de una estructura en forma de árbol de estados de diálogo, donde los nodos representan los cambios en el diálogo y las ramas denotan las acciones llevadas a cabo. Utilizando TrindiKit se han desarrollado sistemas de diálogo como GoDis (Bohlin et al., 1999) o EDIS (Matheson, Poesio, y Traum, 2000). A continuación se resumen unas breves características de un conjunto de sistemas desarrollados siguiendo esta aproximación:

- En el proyecto DIPPER (Bos et al., 2003) se ha desarrollado un conjunto de agentes software, siguiendo la arquitectura OAA (Open Agent Architecture), para la elaboración de sistemas de diálogo. DIPPER adopta la estructura de registros y tipos de datos definida para el Trindikit. La principal diferencia de la aproximación desarrollada es que no existe una separación entre las reglas de actualización y las de selección, definiéndose además una semántica propia en lugar de utilizar Prolog.

El gestor de diálogo de DIPPER está compuesto por dos agentes OAA. El primero de ellos (DME, Dialogue Move Engine) se encarga del tratamiento de la entrada suministrada por los módulos previos del sistema (normalmente el reconocedor) y de realizar llamadas a los agentes de salida del sistema, como el sintetizador. El segundo componente (DME server) gestiona las comunicaciones entre el DME y

el resto de agentes del sistema, pudiendo gestionar múltiples peticiones al mismo tiempo.

Entre los sistemas desarrollados utilizando la metodología diseñada en DIPPER destacan varios prototipos en el ámbito del control de dispositivos domésticos y robots, entre ellos cabe citar el proyecto D'Homme (Bos y Oka, 2002). En (Lemon, Georgila, y Henderson, 2006) se describe un sistema de diálogo multimodal TownInfo, que proporciona información turística. Se han desarrollado dos prototipos utilizando el gestor de diálogo DIPPER. Para el diseño de este sistema se han utilizado técnicas de aprendizaje reforzado conjuntamente con simulación de usuarios.

- En (Weng et al., 2006) se describe el sistema de diálogo CHAT (Conversational Helper for Automotive Tasks), para el que se han definido dos dominios diferenciados en el entorno del automóvil: facilitar el control de un dispositivo MP3 y proporcionar información sobre restaurantes. El gestor de diálogo del sistema se ha implementado utilizando la aproximación de estados de la información. El contexto del diálogo se estructura mediante una representación en forma de árbol, en la que se incluye la historia del diálogo. Los modelos de los dispositivos y sus acciones se enlazan con esta representación de forma externa para determinar la nueva respuesta del sistema. Para incrementar la robustez del sistema se definen múltiples métodos de interpretación para cada uno de los tipos de respuestas del sistema, utilizándose un algoritmo para realizar la selección entre ellos, que tiene en cuenta las medidas de confianza de comprensión y factores relativos a la definición del dominio.
 - En (Purver, Ratiu, y Cavedon, 2006) se presenta un gestor basado en la aproximación de estados de la información. El contexto del diálogo se representa mediante un árbol con reglas que definen el efecto de tomar un conjunto de decisiones en cada estado. Para realizar esta representación del diálogo se han utilizado programas en Java, a partir de los cuales se define la jerarquía de acciones en los estados, las reglas a aplicar en el modelo, la representación de los turnos de sistema y de los parámetros de control de los dispositivos, y las especificaciones para llevar a cabo tareas de desambiguación o petición de información. Además, se han definido múltiples métodos de interpretación, cada uno capaz de propias hipótesis y confianzas asociadas. Para la elección de la siguiente acción del sistema (*dialogue move*) se tienen en cuenta múltiples hipótesis del reconocedor, los métodos de interpretación, los tipos de respuestas posibles y el conjunto de nodos antecedentes para un determinado tipo de respuesta.
 - En (Roque et al., 2006) se describe un sistema de diálogo en el dominio militar para prácticas de artillería simuladas. Siguiendo las especificaciones de un manual militar, el sistema actúa como un radio operador (FDC, *Fire Direction Center*) en un entorno militar simulado y gestiona llamadas de los observadores militares (FO, *Forward Observer*) que realizan las prácticas. Los diálogos pueden dividirse en tres fases. En primer lugar, los observadores se identifican y suministran sus coordenadas, tipo de fuego que solicitan, localización y tipo de blanco. Cada uno de estos datos se confirma por repetición. En una segunda fase, el CFF toma la iniciativa del diálogo y describe la operación solicitada, requiriendo una confirmación del FO previa a la realización de cada una de las operaciones críticas. En la tercera fase, el FO
-

vuelve a tomar la iniciativa del diálogo e informa del fin de la misión o del ajuste de parámetros para realizar una nueva operación.

El gestor de diálogo desarrollado utiliza la aproximación de estados de la información para definir la información relevante que determina el estado del diálogo. Asimismo, se han definido un conjunto de reglas por las que se reconoce la actualización de la información del estado y se determina qué acción llevar a cabo. La información que define el estado es de tres tipos fundamentales: se utilizan siete componentes para determinar si existe suficiente información para realizar el fuego, ocho componentes para almacenar la información sobre el blanco y el entorno y cuatro componentes para realizar tareas de desambiguación y tratamiento de fallos de comprensión.

2.2.3. Metodologías estadísticas. Aprendizaje Reforzado

Las técnicas de aprendizaje reforzado (*reinforcement learning*) para la gestión de diálogo se fundamentan en explorar el conjunto de acciones que puede llevar a cabo el sistema, y determinar la mejor selección de acciones, o estrategia del gestor de diálogo (*policy*), que optimizará el funcionamiento del sistema, representado mediante una función de utilidad (como, por ejemplo, la evaluación del sistema llevada a cabo por los usuarios). Para realizar esta optimización puede utilizarse un corpus de diálogos adquirido mediante la interacción de usuarios reales con el sistema o simular el comportamiento del usuario para obtener los datos de entrenamiento requeridos (ver Sección 2.4).

Para explorar las diferentes elecciones de acciones que puede realizar el sistema para cada estado del diálogo, es necesario definir una representación explícita de todos los estados del diálogo y de las alternativas que puede llevar a cabo el sistema en cada uno de ellos. Cada estado incorpora información que es relevante para realizar la elección de acciones, por ejemplo, las medidas de confianza de la información y la historia del diálogo. Dada la naturaleza de los algoritmos de aprendizaje que se utilizan, la información debe resumirse en función de un conjunto pequeño de características.

El componente final del modelo es la recompensa (refuerzo, *reward*) asociada a cada estado. La recompensa hace referencia a las consecuencias inmediatas de ejecutar una acción en un determinado estado, determinándose la recompensa final acumulada en el diálogo teniendo en cuenta parámetros como el número de correcciones realizadas, el número de accesos a las bases de datos, los errores que hayan podido cometerse en el reconocimiento de la voz, la duración del diálogo, etc. La utilidad de tomar una decisión a en el estado s es la recompensa asignada a dicha acción más la suma de las recompensas acumuladas hasta el instante actual del diálogo, asumiendo que se ha utilizado la mejor estrategia de diálogo.

Puede consultarse numerosa bibliografía referente a la utilización de técnicas de aprendizaje reforzado para obtener la mejor estrategia de gestión de diálogo:

- Los investigadores de AT&T exponen en (Levin, Pieraccini, y Eckert, 2000) que la gestión del diálogo puede analizarse como un problema de optimización de una función objetivo, representativa de las componentes relevantes de la tarea de diálogo considerada. Bajo este planteamiento, el sistema de diálogo se implementa mediante un modelo estocástico, el MDP (Proceso de Decisión de Markov), que posibilita el

aprendizaje automático de las estrategias de diálogo mediante la interacción con un usuario simulado.

Esta técnica se aplica para obtener automáticamente una estrategia óptima para la tarea ATIS. El aprendizaje de los modelos se realiza mediante una combinación de aprendizaje reforzado (el sistema aprenderá la estrategia óptima en su interacción con los usuarios) y aprendizaje supervisado (modelos de simulación de usuarios). El diálogo puede formalizarse como un proceso de decisión secuencial por medio de un conjunto de acciones, un espacio de estados y una estrategia.

Los MDP sirven como representación formal del diálogo persona-máquina y sirven como base para formular los problemas del aprendizaje de una estrategia para gestionar el diálogo (Biermann y Long, 1996), (Levin y Pieraccini, 1997), (Singh et al., 1999), (Levin, Pieraccini, y Eckert, 2000). Un MDP se describe formalmente mediante un espacio de estados finitos S , un conjunto finito de acciones A , un conjunto de probabilidades de transición T y una función de recompensa R . En un instante de tiempo t , el gestor del diálogo se encuentra en un determinado estado $s_t \in S$, ejecuta una acción discreta $a_t \in A$, transita a un nuevo estado s_{t+1} de acuerdo con la probabilidad $p(s_{t+1}|s_t, a_t)$ y recibe una recompensa r_{t+1} . La propiedad de Markov asegura que el estado y la recompensa en el instante $t + 1$ dependen únicamente del estado y acción del instante t .

Modelando el diálogo utilizando MDPs, el gestor del diálogo puede entenderse como un agente que se desplaza a través de una red de estados de diálogos interconectados. Comenzando en un determinado estado inicial, el gestor va transitando entre estados tomando decisiones y recibiendo recompensas tras cada una de ellas. Dado que la respuesta del usuario a una determinada acción del sistema se desconoce, las transiciones son no-deterministas, es decir, la elección de una acción a en un estado s en el instante t no conduce siempre al mismo estado y recompensa en el instante $t + 1$.

Mediante los MDP, la estrategia del diálogo puede entenderse como el establecimiento de una correspondencia entre estados y acciones: para cada estado s , la estrategia selecciona la próxima acción del sistema a . De esta forma, la gestión del diálogo puede formalizarse como un problema matemático de optimización. La estrategia óptima es aquella que maximiza la recompensa acumulada en el tiempo, es decir, aquella en la que durante la evolución del diálogo se seleccionan las acciones con mayor recompensa asociada. Aunque puede conseguirse un comportamiento estocástico del modelo utilizando para ello una correspondencia probabilística para las acciones de cada uno de los estados, se suelen utilizar estrategias deterministas.

La definición correcta del espacio de estados del diálogo S y del conjunto de acciones del sistema A es fundamental para el funcionamiento del modelo MDP. En tareas en las que el usuario debe aportar información para que el sistema le responda a sus consultas (tareas *slot-filling*) la aproximación más utilizada consiste en definir un número de variables de estado en función del número de campos (atributos o *slots*) que debe completar el usuario para realizar las consultas permitidas. En este caso, el número posible de estados viene determinado por la suma del número de estados en los que pueden encontrarse cada uno de los *slots*. Para que el número de

estados no se haga inabordable, es habitual utilizar categorías para definir el estado del slot (como “desconocido”, “conocido” y “confirmado”). Además debe limitarse el conjunto de acciones del sistema. Una práctica común suele consistir en representar dichas acciones como tuplas consistentes en un acto de diálogo, el nombre del slot y un valor asociado al mismo (por ejemplo: $\langle \text{confirmación implícita, ciudad destino, Londres} \rangle$).

Las ecuaciones que definen el comportamiento del MDP son:

- La función de transición.

$$T(s', a, s) = P(s_{t+1} = s' | a_t = a, s_t = s) \quad (2.1)$$

- La matriz de la estrategia (*policy*) del diálogo.

$$\pi(s, a) = P(a_t = a | s_t = s) \quad (2.2)$$

- La recompensa esperada.

$$R(s', a, s) = \varepsilon(r_{t+1} | s_{t+1} = s', a_t = a, s_t = s) \quad (2.3)$$

La recompensa hace referencia al cumplimiento del objetivo del diálogo. Representando el diálogo como el recorrido por una secuencia de estados s_0, s_1, \dots, s_T , la recompensa global del diálogo puede calcularse mediante la expresión:

$$R = \sum_{t=1}^T R(s_{t+1}, a_t, s_t) \quad (2.4)$$

y el objetivo del diálogo es obtener una función que maximice dicha expresión.

Sea V una función que representa la recompensa esperada por transitar del estado s al estado terminal s_T dada la estrategia π , el cálculo de la estrategia óptima del diálogo puede realizarse de forma recursiva mediante la expresión

$$V^\pi(s) = \sum_a \pi(s, a) \sum_{s'} T(s', a, s) [R(s', a, s) + V^\pi(s')] = \sum_a \pi(s, a) Q^\pi(s, a) \quad (2.5)$$

donde $Q(s, a)$ proporciona la recompensa asociada a seleccionar la acción a en el estado s .

La estrategia del diálogo óptima es aquella que maximiza la función V para todos los estados del espacio S .

$$V^*(s) = \max_{\pi} V^\pi(s) \quad \forall s \in S \quad (2.6)$$

La función V óptima puede obtenerse mediante:

$$V^*(s) = \max_a \sum_{s'} T(s', a, s) [R(s', a, s) + V^*(s')] \quad (2.7)$$

De forma similar, la función Q óptima puede hallarse resolviendo:

$$Q^*(s, a) = \sum_{s'} T(s', a, s) \left[R(s', a, s) + \max_{a'} Q^*(s', a') \right] \quad (2.8)$$

y, finalmente, la estrategia óptima del diálogo viene dada por:

$$\pi^*(s) = \arg \max_a Q^*(s, a) \quad (2.9)$$

Si las probabilidades de transición y la función de recompensa se conocen a priori, la estrategia óptima (π^*) puede obtenerse mediante técnicas de programación dinámica (Iteración de Valores o Iteración de Estrategias).

El algoritmo Q-Learning (Watkins, 1989) es una de las soluciones más sencillas cuando estas funciones se desconocen. Se basa en utilizar valores de calidad (Q -values) para cada par (s, a) . Estos valores estiman la respuesta esperada tras seleccionar la acción a en el estado s siguiendo la estrategia π .

El proceso comienza inicializando arbitrariamente los valores de calidad para todos los pares estado-acción, que suelen disponerse en forma de matriz. Durante la interacción del sistema con el usuario (simulado), los valores de la matriz van actualizándose iterativamente para obtener una mejor estimación de los mismos.

Después de llevar a cabo una acción a en el estado s , la respuesta del usuario origina una transición a un estado s' y una recompensa r , recalculándose el valor de calidad utilizando:

$$Q(s, a) := (1 - \alpha)Q(s, a) + \alpha(r + \gamma \max_{a'} Q(s', a')) \quad (2.10)$$

donde α representa una tasa de aprendizaje entre 0 y 1; y γ es un factor de descuento entre 0 y 1 que pondera las diferentes recompensas. Una vez se ha estimado el valor de calidad óptimo del conjunto de pares estado-acción (teóricamente tras visitar un número infinito de veces todas las posibles combinaciones de pares), la estrategia óptima viene dada por la Ecuación 2.9. Una forma típica de actuar para resolver este problema consiste en utilizar una estrategia ϵ -greedy: en cada iteración se selecciona un número aleatorio $0 < \beta < 1$, si $\beta < \epsilon$ la siguiente acción se selecciona aleatoriamente; en caso contrario, se selecciona la mejor acción. El valor de ϵ va disminuyéndose conforme aumentan el número de ciclos.

- En (Krsmanovic et al., 2006) se modela la gestión del diálogo mediante un MDP y se aplica el aprendizaje reforzado para intentar aprender las acciones óptimas del diálogo. El sistema de diálogo controla un robot encargado de realizar tareas de identificación del usuario mediante el reconocimiento de la voz. Los estados del MDP están formados por la concatenación del turno actual del usuario y de dos

parámetros que caracterizan la similitud entre locutores. Se definieron un total de 80 estados.

- En (Cuayáhuitl et al., 2006a) se aplica el MDP y el aprendizaje reforzado a un dominio de planificación de viajes. El principal objetivo del trabajo es describir una técnica para reducir el espacio de estados-acciones, con la finalidad de realizar un aprendizaje más rápido y reducir la carga computacional. La metodología propuesta se aplica para el aprendizaje de estrategias de diálogo con múltiples objetivos que optimicen las confirmaciones de uno o más datos.

El algoritmo propuesto (*sapReduction*) genera espacios reducidos con restricciones utilizando para ello tres niveles de reducción:

- Reducción de estados: El espacio de estados completo S está conformado por todas las posibles combinaciones de slots Q y variables de estado V , en las que se incluyen combinaciones válidas y no permitidas, que son las que este paso del algoritmo trata de eliminar. Para llevar a cabo esta reducción se tiene en cuenta que un slot terminal (donde típicamente se lleva a cabo la transacción) requiere que los slots no terminales hayan sido confirmados. Además se tiene en cuenta que slots no son necesarios para el objetivo del diálogo.
- Reducción de acciones: El espacio de acciones completo A lo componen todas las combinaciones de slots Q y de acciones simples A^s . Este paso del algoritmo tiene como objetivo eliminar las acciones que no son válidas para cada estado. Para ello se han definido categorías que representan las acciones válidas del sistema, agrupando las combinaciones de slots Q y acciones A^s permitidas.
- Reducción del espacio de búsqueda para cada partición SA_i : Para ello, se tratan de fijar fronteras entre los diversos objetivos del diálogo. En este paso se propone la generación de múltiples espacios de búsqueda de acuerdo con las particiones especificadas en el diseño del diálogo, equivalentes a los objetivos definidos. La finalidad es agrupar múltiples espacios de estados en un único espacio.

Los resultados presentados en el trabajo muestran que la reducción de espacios posibilita menores requerimientos de memoria (94 % de reducción), un aprendizaje más rápido (la convergencia se alcanza un 93 % más rápidamente) y mejor funcionamiento (8,4 % de reducción y un 7,7 % de mayor recompensa final en el modelo) frente a la utilización del espacio de estados sin aplicar el algoritmo de reducción. En la experimentación se han utilizado técnicas de simulación de usuarios.

- En (Young, 2000) (Young, 2002b), investigadores de la Universidad de Cambridge consideran también la aproximación markoviana, el MDP aplicado al sistema de diálogo, y el uso del aprendizaje reforzado para que el gestor de diálogo aprenda la estrategia más adecuada. En particular, la cuestión objeto de su estudio es cómo crear el sistema inicial (*bootstrapping*). El método se ha probado usando el corpus Autoroute, que consta de 166 diálogos, en los que el mago de Oz atiende consultas sobre direcciones y rutas en el Reino Unido. Más recientemente, en (Williams y Young, 2007a) (Young et al., 2005) (Williams y Young, 2006a) (Schatzmann et

al., 2006), los mismos investigadores mejoran sus resultados mediante el uso de un MDP parcialmente observable (POMDP). En los resultados experimentales, el modelo POMDP supera a los MDP tradicionales. Además, el método descrito permite mejorar automáticamente el funcionamiento de gestores de diálogo sencillos (diseñados a mano).

Formalmente un POMDP puede definirse como una tupla $\{S, A, T, R, O, Z, \lambda, b_0\}$ donde:

- S es el conjunto de estados que definen el comportamiento del agente.
- A es el conjunto de acciones que puede tomar el agente.
- T define una probabilidad de transición $P(s'|s, a)$.
- R define la recompensa esperada $r(s, a)$.
- O es el conjunto de observaciones que el agente puede recibir sobre el mundo.
- Z define la probabilidad de observación $P(o'|s', a)$.
- λ es un factor de descuento geométrico $0 \leq \lambda \leq 1$.
- b_0 es el estado inicial $b_0(s)$.

Un POMDP funciona del siguiente modo. En cada instante, el sistema se encuentra en un estado no observado s . El sistema selecciona una acción a_m , recibe una recompensa r y transita a un estado (no observado) s' , donde s' depende únicamente de s y a_m . El sistema recibe una observación o' que depende de s y a_m . Aunque la observación permite que el sistema disponga de alguna evidencia sobre el estado s en el que se encuentra, s no se conoce con exactitud, definiéndose una probabilidad $b(s)$ (*belief state*) que indica la probabilidad de que el sistema se encuentre en un determinado estado s . En cada instante, esta probabilidad se actualiza basándose en o' y a_m :

$$b'(s') = P(s'|o', a_m, b) = k \cdot P(o'|s', a_m) \sum_{s \in S} P(s'|a_m, s) b(s) \quad (2.11)$$

siendo $k = P(o'|a_m, b)$ un factor de normalización.

En cada instante t el sistema recibe una recompensa $r(b_t, a_{m,t})$ dependiente del estado b_t y la acción seleccionada $a_{m,t}$. La recompensa acumulada durante el diálogo (*return*) puede calcularse mediante:

$$R = \sum_{t=0}^{\infty} \lambda^t R(b_t, a_{m,t}) = \sum_{t=0}^{\infty} \lambda^t \sum_{s \in S} b_t(s) r(s, a_{m,t}) \quad (2.12)$$

Cada acción $a_{m,t}$ viene determinada por la estrategia $\pi(b_t)$ y la construcción del modelo POMDP implica encontrar la estrategia π^* que maximiza el *return*. Una resolución exacta de este problema es intratable (Kaelbling, Littman, y Cassandra, 1998) (Littman, 1994), aunque pueden obtenerse soluciones aproximadas que proporcionan estrategias de diálogo útiles. La solución más simple consiste en discretizar el espacio de estados y utilizar los métodos de optimización descritos para el MDP (Sutton y Barto, 1998).

- En (Williams, Poupart, y Young, 2006) se describe como modelar la gestión del diálogo mediante la utilización de POMDP. A esta aproximación se le conoce como SDS-POMDP. En ella, la variable de estado $s \in S$ se descompone en tres términos:

$$s = (s_u, a_u, s_d) \quad (2.13)$$

- el objetivo del usuario, $s_u \in S_u$;
- la acción del usuario, $a_u \in A_u$;
- la historia del diálogo, $s_d \in S_d$.

Para factorizar el modelo, la función de transición del modelo se descompone de la siguiente forma:

$$P(s' | s, a_m) = P(s'_u | s_u, s_d, a_u, a_m) P(a'_u | s'_u, s_u, s_d, a_u, a_m) P(s'_d | a'_u, s'_u, s_u, s_d, a_u, a_m) \quad (2.14)$$

Al primer término se le denomina modelo de objetivo del usuario (*user goal model*) e indica cómo varía el objetivo del usuario en cada instante del diálogo. Se asume que el objetivo del usuario en un determinado instante depende únicamente del objetivo actual y de las acciones llevadas a cabo por el sistema hasta dicho instante:

$$P(s'_u | s_u, s_d, a_u, a_m) = P(s'_u | s_u, a_m) \quad (2.15)$$

El segundo término, denominado modelo de acción del usuario (*user action model*) indica qué acciones del usuario son más probables para cada instante. Se realiza la misma suposición que para el primer término:

$$P(a'_u | s'_u, s_u, s_d, a_u, a_m) = P(a'_u | s'_u, a_m) \quad (2.16)$$

El tercer término, modelo de historia del diálogo (*dialogue history model*) indica cómo han afectado las acciones del usuario y el sistema a la historia del diálogo:

$$P(s'_d | a'_u, s'_u, s_u, s_d, a_u, a_m) = P(s'_d | a'_u, s_d, a_m) \quad (2.17)$$

Por tanto, la función de transición puede reescribirse de la siguiente forma

$$P(s' | s, a_m) = P(s'_u | s_u, a_m) P(a'_u | s'_u, a_m) P(s'_d | a'_u, s_d, a_m) \quad (2.18)$$

Con esta representación se reduce el número de parámetros requeridos para la función de transición y permite la estimación de los diferentes parámetros por separado.

La observación o del POMDP se puede descomponer en dos términos: las hipótesis del reconocedor $\tilde{a}_u \in A_u$ y las medidas de confianza $c \in R$. La función de observación viene dada por:

$$P(o' | s', a_m) = P(\tilde{a}_u, c' | s'_u, s'_d, a'_u, a_m) \quad (2.19)$$

Asumiendo que la observación depende únicamente de la acción llevada a cabo por el usuario y por la gramática g seleccionada por el gestor de diálogo:

$$P(\tilde{a}_u, c' | s'_u, s'_d, a'_u, a_m) = P(\tilde{a}_u, c' | a'_u, g) \quad (2.20)$$

que representa la distribución de probabilidad de “observar” las hipótesis del reconocedor \tilde{a}_u con medida de confianza c cuando el usuario realmente seleccionó la acción a_u y se activó la gramática g .

Por último, utilizando las expresiones anteriores, la probabilidad $b(s)$ puede escribirse de la forma siguiente:

$$b'(s'_u, s'_d, a'_u) = k \cdot P(\tilde{a}_u, c' | a'_u, g) \cdot P(a'_u | s'_u, a_m) \cdot \sum_{S_u} P(s'_u | s_u, a_m) \cdot \sum_{S_d} P(s'_d | a'_u, s'_u, s_d, a_m) \cdot \sum_{a_u} b(s_u, s_d, a_u) \quad (2.21)$$

- En (Williams y Young, 2007b), (Williams y Young, 2006b) y (Williams, Poupart, y Young, 2006) se resume una solución al problema del gran número de estados que pueden presentarse en los sistemas de diálogo reales cuando se modelan mediante los POMDPs, que hace que la estimación de la estrategia óptima sea intratable.

La metodología consiste en centrarse durante el proceso de estimación de la estrategia únicamente en los estados más probables para la acción actual del sistema. De este modo, se definen dos espacios de estados acoplados: el espacio total de estados (*master state space*) y un espacio de estados resumido (*summary state space*) mucho más simple. El espacio de estados resumido contiene los N mejores estados de objetivos (s_u) del espacio total (N es normalmente 1 ó 2) y una codificación simplificada de la acción del usuario a_u y de la historia del diálogo s_d .

La optimización de la estrategia se lleva a cabo utilizando estos dos espacios de estados acoplados, utilizando para ello técnicas basadas en el modelo (por ejemplo PBVI, Point-based Value Iteration (Pineau, Gordon, y Thrun, 2003)) o aproximaciones como *Q-learning* conjuntamente con simuladores de usuarios.

- En (Young et al., 2005) (Young, 2006) (Young et al., 2007) se introduce el modelo de diálogo HIS (*Hidden Information State*), una simplificación específica del modelado del diálogo mediante POMDPs para facilitar su uso en aplicaciones reales, dado que el tamaño del espacio de estados necesario para representar los sistemas de diálogo del mundo real hace que la implementación del modelo SDS-POMDP sea intratable.

La idea principal del modelo HIS consiste en particionar el espacio de estados asumiendo que en un instante t , el espacio de todos los objetivos del usuario S_u puede agruparse en clases de equivalencia (particiones) donde todos los miembros de la clase están enlazados y no son distinguibles. De este modo, las probabilidades se

calculan para cada una de las particiones y no para cada uno de los estados que las componen.

La forma de actuar es la siguiente. Inicialmente todos los estados $s_u \in S_u$ se sitúan en una única partición p_0 . A medida que los diálogos progresan, esta partición raíz se va dividiendo repetidamente en particiones más pequeñas de modo binario. Cada vez que se genera una nueva partición

$$p \rightarrow \{p', p - p'\} \quad (2.22)$$

la probabilidad del modelo puede recalcularse de la siguiente forma:

$$b(p') = P(p' | p) b(p) \quad y \quad b(p - p') = (1 - P(p' | p)) b(p) \quad (2.23)$$

Considerando que la información proporcionada por el usuario viene representada implícitamente por las particiones realizadas, puede asumirse:

$$P(a'_u | s'_u, a_m) = P(a'_u | p', a_m) \quad (2.24)$$

$$P(s'_d | s'_u, a'_u, s_d, a_m) = P(s'_d | p', a'_u, s_d, a_m) \quad (2.25)$$

y, de este modo, la probabilidad final del modelo HIS se expresa mediante la siguiente ecuación:

$$b'(p', a'_u, s'_d) = k \cdot P(o' | a'_u) P(a'_u | p', a_m) \sum_{s_d} P(s'_d | p', a'_u, s_d, a_m) P(p' | p) b(p, s_d) \quad (2.26)$$

La ecuación consta de cuatro distribuciones de probabilidad:

- Modelo de observación: Se aproxima por la probabilidad de la mejor opción del módulo de comprensión:

$$P(o' | a'_u) \approx k' \cdot P(a'_u | o) \quad (2.27)$$

- Modelo de acto de diálogo de usuario. Se compone de dos partes, la probabilidad del bigrama del tipo de acto de diálogo de usuario actual dado el tipo de acto de diálogo de sistema que le ha precedido, y una probabilidad que denota el grado de consistencia del acto de diálogo de usuario con la partición (p').

$$P(a'_u | p', a_m) \approx P(\tau(a'_u) | \tau(a'_m)) P(\mathcal{M}(a'_u) | p') \quad (2.28)$$

donde $\tau(a)$ hace referencia al tipo de acto de diálogo y $\mathcal{M}(a)$ denota la correspondencia o no del acto de diálogo a con la partición actual p' .

- Modelo de diálogo: Es completamente heurístico.

$$\begin{aligned} P(s'_d | p', a'_u, s_d, a_m) &= 1 \quad \text{si } s'_d \text{ es consistente con } p', a'_u, s_d, a_m \\ &= 0 \quad \text{resto de los casos} \end{aligned} \quad (2.29)$$

- Refinamiento del espacio: Depende de las reglas de ontología que definen el dominio de la aplicación. Los objetivos del usuario se construyen utilizando reglas con probabilidades establecidas a priori. La probabilidad correspondiente a utilizar la secuencia de reglas r_1, r_2, \dots, r_k para dividir la partición p en la sub-partición p' se define de la siguiente forma:

$$P(p' | p) = \prod_{i=1}^k P(r_i) \quad (2.30)$$

Para implementar en la práctica el modelo HIS, el espacio de los objetivos del usuario se representa mediante una estructura de árbol jerárquico, donde un conjunto de reglas de ontología describen la estructura jerárquica de la información y los valores específicos que pueden asignarse a los nodos terminales. Cada uno de los nodos no terminales dispone de probabilidades asociadas a establecer particiones en cada uno de ellos $P(p'|p)$.

Las particiones en el espacio de objetivos de usuario se representan mediante un bosque de árboles, representando cada uno de ellos una única partición. Al comienzo del diálogo existe una única partición con un único nodo con probabilidad uno asociada. Cada uno de los actos de diálogo de usuario entrante se intenta emparejar con alguna de las particiones disponible en el turno actual. En caso de no encontrarse el emparejamiento, se consultan las reglas de la ontología y el sistema intenta buscar el emparejamiento expandiendo el árbol actual.

La técnica de optimización para realizar la búsqueda de la estrategia óptima se basa actualmente en utilizar Q-learning conjuntamente con un simulador de usuario externo (Young, 2006). Esta metodología se ha aplicado para la optimización de la estrategia del diálogo en una tarea de información turística. El aprendizaje se lleva a cabo iterando mediante el algoritmo Monte Carlo, manteniendo constante la estrategia actual durante 5000 diálogos, actualizándose de forma continua los valores de confianza asociados a cada uno de los espacios de estados resumidos.

- En (Cuayáhuitl et al., 2006b) se describe una aproximación, conocida como *Hierarchical Abstract Machines* (HAMs), para la obtención de la estrategia óptima del diálogo basada en la combinación de máquinas de estados finitos y MDPs. Las ventajas mencionadas para esta aproximación son:
 - Las estrategias del diálogo se especifican de forma que el desarrollador del sistema decide qué aspectos fijar a mano y qué optimizar.
 - Posibilita un aprendizaje más rápido, dado que se incorpora conocimiento del dominio para reducir el número de estados del espacio.

- Transferencia de conocimiento, dado que las HAMs pueden reutilizarse.

Una HAM puede entenderse como un MDP en el que se restringe las acciones que pueden tomarse en cada estado durante el aprendizaje reforzado. Son similares a las máquinas de estados finitos no deterministas: se parte de un estado inicial y se recorren los estados hasta alcanzar un estado de parada, en el que se devuelve el control al usuario.

Formalmente, una HAM es una colección de tres tuplas $H_i = (\mu, I, \delta)$, donde μ es un conjunto finito de máquinas de estados, I es el estado inicial y δ es la función de transición que determina el siguiente estado mediante transiciones deterministas o estocásticas.

Los principales tipos de máquinas de estados son: *start* (ejecuta la máquina actual), *action* (ejecuta una acción), *call* (ejecuta otra máquina), *choice* (selecciona el siguiente estado de la maquina) y *stop* (detiene la ejecución y devuelve el control). El estado H_i se considera abstracto si incluye estados de elección no deterministas.

Dado un MDP M y una HAM H , puede inducirse un MDP $M' = H\Delta M$, cuya solución define una estrategia óptima que maximiza la recompensa total esperada ejecutando H en M . Los parámetros del MDP $M' = \langle S', A', T', R' \rangle$ son:

- S' es el conjunto de estados obtenido del producto cruzado entre los estados de H y los estados de M ,
- las acciones A' de un determinado estado son sólo aquellas relativas a la máquina,
- la función de transición T' se corresponde a ejecutar en paralelo las funciones de transición T y δ ,
- la función de recompensa R' coincide con la de R para las acciones de un único paso y es cero en el resto de casos.

El objetivo de definir M' , que es un SMDP (*Semi-Markov Decision Process*), es poder trabajar con un espacio más reducido de acciones. El algoritmo de aprendizaje, que se denomina SMDP Q-learning, se aplica extendiendo la tabla de valores de calidad al formato $Q([s, m], a)$, donde s es el estado, m es la máquina y a es la acción seleccionada.

2.2.3.1. Modelado del diálogo mediante redes bayesianas

Otra aproximación estadística alternativa al modelado del diálogo mediante Procesos de Decisión de Markov y aprendizaje reforzado, consiste en la utilización de redes bayesianas (*Belief Networks*, BN) para representar las interacciones del diálogo. En (Meng, Wai, y Pieraccini, 2003) se citan las principales ventajas de la utilización de las BN para el modelado de la iniciativa mixta del diálogo:

- Las probabilidades de la BN pueden estimarse automáticamente a partir de un conjunto de datos disponible, lo que facilita la portabilidad y escalabilidad a otros dominios. El objetivo del usuario puede identificarse mediante inferencia probabilística.

- La topología de la BN puede aprenderse también a partir de un corpus de datos de entrenamiento. La topología puede capturar las dependencias entre los nodos de la BN, representando cada uno de estos nodos un concepto semántico del dominio del sistema
- La probabilidad de propagación en una red BN se corresponde con el cómputo de probabilidades de los eventos que pueden producirse. Mediante este procedimiento pueden detectarse qué conceptos restan por solicitar al usuario o cuáles de ellos deben confirmarse previamente a la consulta.
- La topología y las probabilidades de la BN puede aprenderse de forma automática, definirse a mano o utilizar ambas aproximaciones.

Las redes bayesianas se han utilizado tradicionalmente en el marco de la comprensión del lenguaje natural. En este contexto, existe un conjunto finito de conceptos semánticos (M) y de objetivos de usuario (N). Los objetivos G_i y los conceptos C_j son binarios, siendo ciertos si aparecen en el turno. Por tanto, el problema de la comprensión puede verse como llevar a cabo N decisiones binarias con N BNs (cada una para uno de los objetivos del usuario). La BN para el objetivo G_i tiene como entrada un conjunto de conceptos semánticos C extraídos de la intervención del usuario. La red proporciona la probabilidad a posteriori $P(G|C)$, a partir de la que se toma una decisión binaria mediante su comparación con un umbral. La expresión de esta probabilidad coincide con la formulación de Naive Bayes:

$$P(G_i = 1|\vec{C}) = \frac{P(\vec{C}|G_i = 1)P(G_i = 1)}{P(\vec{C})} \quad (2.31)$$

Se asume que el objetivo G_i está presente si la probabilidad $P(G_i|C)$ es mayor que un umbral θ . Utilizando esta formulación, las consultas del usuario que no superen el umbral de ninguna de las redes se suponen fuera del dominio del sistema.

La idea principal es utilizar las redes bayesianas para detectar conceptos automáticamente de acuerdo a las restricciones del dominio capturadas por sus probabilidades. La detección automática de conceptos se lleva a cabo mediante la técnica de *backward inference*. Una vez inferido el objetivo del diálogo (G_i) en una consulta determinada del usuario, el nodo objetivo de la BN correspondiente se instancia (a 0 ó 1) para examinar la confianza de la red para cada uno de los conceptos de entrada. Si la topología de la red BN asume la independencia condicional entre los conceptos, su probabilidad actualizada es simplemente $P(C_j|G)$. Si existe dependencia entre los diferentes conceptos, la probabilidad actualizada del objetivo ($P^*(C_i)$) se calcula mediante:

$$P^*(\vec{C}|G_i) = P(\vec{C}|G_i)P^*(G_i) = P(\vec{C}, G_i) \frac{P^*(G_i)}{P(G_i)} \quad (2.32)$$

donde $P^*(G_i)$ se actualiza instanciando el nodo objetivo, $P(\vec{C}, G_i)$ se obtiene a partir del corpus de entrenamiento.

Basándose en el valor de $P^*(C_j)$, se decide mediante la utilización del umbral si C_j está presente o no. Esta decisión se compara con la ocurrencia real de C_j en la intervención del usuario. Si la decisión indica que C_j no debería estar presente pero está realmente en

dicha intervención, el concepto se marca como espurio y el modelo de diálogo solicitará su confirmación. Si la decisión binaria indica que C_j debería estar presente pero no es así en la consulta del usuario, el concepto se etiqueta como requerido y el modelo de diálogo lo solicitará al usuario.

- En (Meng, Wai, y Pieraccini, 2003) se aplican las redes bayesianas para el desarrollo de un modelo de diálogo para el sistema CU FOREX (Meng, Lee, y Wai, 2000). El dominio de este sistema es proporcionar información referente a dos tipos de consultas bancarias: cambios de moneda y tipos de intereses. Para este dominio se han definido cinco conceptos. Cada una de las dos BN definidas (una para cada consulta del sistema), recibe como entrada estos cinco conceptos. La topología de las redes se entrenaron automáticamente a partir de un corpus. La evaluación del modelo se llevó a cabo a partir de 550 diálogos, obteniéndose porcentajes de éxito de la tarea del 96 %.

En este trabajo se realiza también la aplicación de las BN para el dominio ATIS. A partir de un corpus de datos extraído de ATIS-3, se implementaron 11 BN (una para cada una de los objetivos detectados) y se definieron un total de 60 etiquetas semánticas (conceptos y atributos necesarios para realizar las consultas en la base de datos). Para solucionar la redundancia del modelo de diálogo (debida a la no detección de las equivalencias entre atributos de la tarea) se propuso la definieron dos umbrales y el refinamiento a mano de las probabilidades.

- Quartet es una plataforma desarrollada por Microsoft Research para implementar sistemas de diálogo multimodales. La incertidumbre del dialogo se representa mediante redes bayesianas. Se han definido cuatro niveles de representación que soportan inferencia y toma de decisiones (canal, señal, intención y conversación).

2.3. Evaluación de sistemas de diálogo

A medida que se producen avances en el estudio y desarrollo de sistemas de diálogo, se hace necesario desarrollar nuevas medidas de evaluación para comprobar si estos sistemas son efectivos o no. La tarea de fijar nuevas medidas, es decir, de plantear nuevos procedimientos y medidas que sean aceptadas unívocamente por la comunidad científica para la evaluación de este tipo de sistemas presenta muchas dificultades. Puede considerarse que el campo de las técnicas y medidas de evaluación de este tipo de sistemas se encuentra en una fase inicial de desarrollo.

En los últimos años se han desarrollado diferentes iniciativas para definir marcos generales en los que englobar el diseño y evaluación de sistemas de diálogo. Algunos ejemplos en Europa son EAGLES (Expert Advisory Group on Language Engineering Standards) 96, ELSE99 y DISC99.

Otras instituciones europeas centradas en el estudio y definición de técnicas de evaluación son: COSCODA (Coordinating Committee on Speech Databases and Speech I/O Systems), dedicada a aspectos relacionados con la creación de bases de datos multilingüe, ELRA (European Language Resources Association), centrada en la colección y distribución de recursos lingüísticos, SQUALE (Speech Recognition Quality Assessment for

Linguistic Engineering) centrada en la adaptación a contextos multilingües del paradigma LVCSR (Large Vocabulary Continuous Speech Recognition) de ARPA.

En Estados Unidos el principal proyecto ha sido DARPA Communicator 02. Teniendo en cuenta estas iniciativas, pueden considerarse dos tendencias fundamentales a la hora de realizar la evaluación de los sistemas de diálogo:

- Definición de medidas cuantitativas que permitan medir la calidad de funcionamiento del sistema (EAGLES 96, DARPA Communicator).
- Proponer marcos de estudio que equiparen la relevancia de las medidas cualitativas (utilizadas normalmente para medir aspectos subjetivos) y cuantitativas (ELSE 99 y DISC 99).

En la bibliografía sobre evaluación de sistemas puede encontrarse un gran número de tipos de evaluación. A continuación se resumen diferentes criterios reseñados en (San-Segundo, 2004):

- Teniendo el tipo de medidas utilizadas para la evaluación, ésta puede clasificarse como objetiva o subjetiva, dependiendo del número de medidas que predominen:
 - Medidas objetivas: Son aquellas que se obtienen directamente del funcionamiento del sistema, no incluyendo ningún tipo de valoración subjetiva por parte de desarrolladores o usuarios del sistema
 - Medidas subjetivas: Denotan un proceso de valoración subjetiva, normalmente llevado a cabo por los usuarios finales del sistema. Un ejemplo de evaluación utilizando este tipo de medidas es la definida en el proyecto europeo Trindi.

Estas medidas también pueden clasificarse teniendo en cuenta la manera de realizar su cómputo (medidas automáticas o manuales) o su influencia en la calidad global del sistema (mediadas positivas o negativas)

- Teniendo en cuenta el objetivo de la evaluación, puede distinguirse entre:
 - Evaluación general: Se analiza el funcionamiento global del sistema teniendo en cuenta entradas y salidas a nivel general
 - Evaluación por componentes: Se analiza de forma separada el funcionamiento de cada uno de los módulos del sistema, teniendo en cuenta las entradas y salidas parciales del mismo.
 - Según el patrón de referencia tomado una vez realizada la evaluación del sistema, puede distinguirse entre:
 - Evaluación comparativa. Se realizan diferentes sistemas en paralelo con las mismas especificaciones pero desarrollados por centros diferentes Esta evaluación se ha utilizado usualmente en proyectos financiados por DARPA, como DARPA Communicator.
-

- Evaluación temporal. El patrón de referencia es el propio sistema desarrollado, realizando comparaciones de su funcionamiento en las diversas fases temporales de su desarrollo.
- Evaluación sustitutiva. Se evalúa el sistema con respecto a otro con las mismas funcionalidades desarrollado previamente, normalmente con diferente tecnología.
- Evaluación inicial. Una alternativa cuando no hay disponible un sistema de referencia con el que comparar consiste en la estimación a priori de su funcionamiento durante la fase de especificaciones y la posterior evaluación de la desviación con respecto al comportamiento previsto.

A la hora de evaluar el sistema de diálogo de manera global, la propuesta con mayor repercusión a nivel internacional es PARADISE (Walker et al., 1998) (Dybkjaer, Bernsen, y Minker, 2004). El modelo PARADISE (PARAdigm for DIalogue System Evaluation) combina diferentes medidas en una única función que mide el rendimiento del sistema, en correlación directa con la satisfacción de los usuarios del sistema. Los supuestos principales del modelo PARADISE son:

- el objetivo a maximizar es la satisfacción del usuario;
- el éxito en la tarea y varios costes asociados a la interacción (medidas objetivas) pueden usarse para predecir la satisfacción del usuario.

Estos dos supuestos pueden formalizarse en la siguiente ecuación:

$$Satisfaccion\ del\ usuario = (\alpha N(Exito\ de\ la\ tarea)) - \sum_{i=1}^N w_i N(Costes\ del\ dialogo)$$

donde las medidas del éxito de la tarea y los costes del diálogo se utilizan normalizando su distribución $N()$ a una distribución normal de media cero y varianza unidad.

Esta formulación se basa en el modelo mostrado en la Figura 2.6. La maximización de la satisfacción del usuario se lleva a cabo minimizando los costes del diálogo y maximizando el éxito de la tarea. Los costes del diálogo se cuantifican mediante medidas de eficiencia y de calidad.

La utilización del modelo de PARADISE requiere de un corpus de diálogos obtenidos en experimentos controlados en los que los diferentes usuarios evalúan subjetivamente su satisfacción sobre una escala.

Las medidas de éxito de la tarea más comúnmente utilizadas son:

- El Factor Kappa (K) se propuso en la formulación inicial del modelo PARADISE. Se calcula a partir de una matriz de confusión de los valores de atributos intercambiados entre el usuario y el sistema, de forma que la diagonal principal de la matriz indica los casos en el que el sistema reconoce y comprende correctamente la información del turno(s) de usuario. Se utiliza la siguiente expresión:

$$K = \frac{P(A) - P(E)}{1 - P(E)}$$

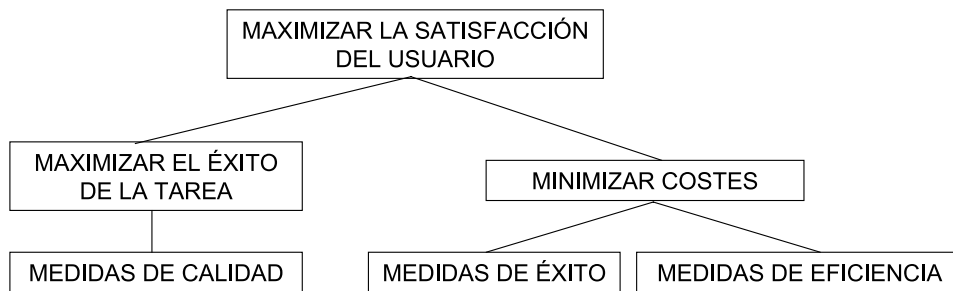


Figura 2.6: Modelo de desarrollo del paradigma PARADISE

donde $P(A)$ es la probabilidad de que el sistema reconozca y comprenda correctamente la información aportada por el usuario; y $P(E)$ es una estimación de las veces que el sistema podría acertar por casualidad, calculada mediante:

$$P(E) = \sum_{i=1}^n \left(\frac{t_i}{T} \right)^2$$

donde t_i es la suma de intentos en la columna i y T es la suma total de intentos.

- Tasa de tarea completada: Es el porcentaje de veces que el sistema satisface correctamente la solicitud realizada por el usuario. Para obtener esta tasa, es necesario revisar manualmente los diálogos transcritos y observar los casos en los que el sistema responde correctamente

Las medidas de eficiencia más importantes son:

- Tiempo medio necesario para completar una tarea (los diferentes tiempos para cada tarea pueden promediarse mediante diferentes criterios: considerar todos los diálogos o sólo aquellos que han terminado con éxito, etc.).
- Tiempo medio por turno (se pueden aplicar los mismos criterios que en el caso anterior para realizar los promedios).
- Número medio de turnos por tarea (mismos promedios que en los dos casos anteriores).
- Mínimo número de turnos o tiempo mínimo para completar una tarea (sirven para establecer una medida de la complejidad del sistema de diálogo independientemente de las pruebas realizadas por los usuarios)
- Tipos de confirmaciones de datos utilizadas (un estudio del número de confirmaciones y de su tipo, explícitas o implícitas, revela el funcionamiento del sistema).
- Número de palabras reconocidas correctamente por turno (a mayor cantidad de palabras correctamente reconocidas, lógicamente, mayor será la calidad del sistema y menor número de turnos se necesitaran para completar la tarea).

Entre las medidas de calidad del sistema, cabe referenciar:

- Tasa de reconocimiento (porcentaje de palabras reconocidas correctamente).
- Tasa de conceptos semánticos correctos (porcentaje de conceptos semánticos correctamente generados por el módulo de comprensión).
- Porcentaje de errores corregidos con éxito (medida de la eficiencia de las técnicas utilizadas para la detección y corrección de errores).
- Tiempo de respuesta del sistema (tiempo que tarda el sistema en reconocer y comprender la frase pronunciada por el usuario).
- Tiempo de respuesta del usuario (pueden detectarse las situaciones del diálogo en los que el usuario tarda en proporcionar una respuesta).
- Número de veces que el usuario no contesta (medida del número de veces que el usuario supera el tiempo máximo establecido para realizar una intervención).
- Número de veces que el usuario solicita repetición (indica el número de veces que el usuario no entiende la información formulada por el sistema).
- Número de veces que el usuario solicita ayuda (indica el entendimiento que tiene el usuario sobre el funcionamiento del sistema).
- Número de veces que el usuario interrumpe al sistema (puede revelar el grado de destreza o entrenamiento que posee el usuario).

El modelo PARADISE se presenta como un marco general para la evaluación de los agentes de diálogo, dado que permite desacoplar los requisitos de la tarea y los comportamientos de los agentes, comparar estrategias de diálogo, calcular el rendimiento de diálogos completos y de subdiálogos, especificar las contribuciones relativas al rendimiento, y comparar agentes que aborden diferentes tareas, mediante una normalización de la complejidad de la tarea. En cuanto a sus inconvenientes, cabe destacar:

- excesivo acoplamiento entre satisfacción de usuario y usabilidad;
- dudas acerca de la posibilidad de predecir la satisfacción de usuario a partir de la información registrada en los ficheros log del sistema;
- dificultad en la interpretación de los cuestionarios, y ausencia de fundamentos teóricos acerca de las cuestiones a incluir en los mismos;
- uso del modelo, por el momento, limitado a experimentos controlados (y no con usuarios reales).

El método de evaluación utilizado por diversos grupos de investigación (por ejemplo, los adscritos al programa ARPA Spoken Language Systems en el dominio de ATIS) consiste en utilizar el protocolo CAS (*Common Answer Specification*). Este protocolo compara el funcionamiento del sistema con una respuesta canónica de la base de datos,

presentando la ventaja de poder realizar la evaluación de forma automática, una vez que los principios para generar las respuestas de referencia se han establecido y un corpus de datos ha sido convenientemente etiquetado. Además, permite la comparación directa entre sistemas de forma sencilla. Sin embargo, el procedimiento de evaluación CAS está bastante limitado. La evaluación se realiza sólo en el nivel de frase, es decir, se compara frase a frase con su respuesta canónica. Tampoco se realiza ninguna distinción entre una respuesta parcialmente correcta y una completamente incorrecta. Por tanto, este método no es completamente eficaz en la evaluación de este tipo de sistemas interactivos, no permite detectar ni corregir errores, no permite evaluar la calidad de las respuestas, etc.

Una posible alternativa consiste en plantear una serie de escenarios donde se le indica al usuario qué información deberá solicitar al sistema. Estos escenarios tienen una respuesta bien definida. El usuario realiza una o varias preguntas al sistema, manteniendo un cierto diálogo con el mismo, y el sistema le envía una respuesta para cada una de ellas, o una nueva pregunta/mensaje de error. Toda esta información queda registrada en un fichero, es decir, cada par pregunta-respuesta. Además, se mide el tiempo consumido por el usuario y el sistema en llevar a cabo los distintos procesos (reconocimiento/transcripción del Mago de Oz de la pregunta, envío de la respuesta del sistema al usuario, tiempo de almacenamiento de toda la información que registra el sistema de medida, tiempo que tarda el usuario en pensar y realizar cada pregunta, etc.). Se suele incluir algún tipo de cuestionario para el usuario, preguntando por su aceptación del sistema, qué le gusta y qué no le gusta, si considera que el sistema le ha “entendido” y con qué frecuencia, si le ha parecido ágil el diálogo, si ha entendido las respuestas enviadas por el sistema, etc. A partir del fichero generado y del cuestionario se realizan una serie de medidas que permiten evaluar cuantitativamente el sistema interactivo. Entre ellas:

- Tiempo en finalizar el escenario (el usuario no sabe que le están midiendo).
- La existencia o no de una solución.
- Si la solución encontrada por el sistema es correcta.
- Número de preguntas que ha necesitado hacer el usuario.
- Número de preguntas del usuario que ha contestado el sistema.
- Número de preguntas que se consideran (por parte de un evaluador experto) que han sido contestadas correctamente, incorrectamente o parcialmente correctas.
- Satisfacción del usuario sobre la base del cuestionario.
- Medida de comprensión de acuerdo a otros criterios como el CAS.

Otras medidas comúnmente utilizadas en la evaluación de los diferentes módulos de un sistema de diálogo son:

- **Reconocimiento Acústico** (Tasa de Acierto o Precisión de Palabras (*Word Accuracy, WA*), Tasa de Inserción de Palabras (*Word Insertions Rate*), Tasa de Sustitución de Palabras (*Word Substitutions Rate*), Tasa de Frases Correctas (*Sentence Accuracy*), etc.
-

La Tasa de Error (*Word Error rate: WER*) y el *Word Accuracy* pueden calcularse a partir del número total de sustituciones en las frases de evaluación (N_S), el número total de borrados (N_B), el número total de inserciones (N_I) y el número total de palabras en dichas frases (N_T):

$$Sus(\%) = 100 * \frac{N_S}{N_T}$$

$$Borr(\%) = 100 * \frac{N_B}{N_T}$$

$$Inser(\%) = 100 * \frac{N_I}{N_T}$$

$$WER(\%) = Subs(\%) + Inser(\%) + Borr(\%)$$

$$WA(\%) = 100\% - WER(\%)$$

El cálculo de estos porcentajes requiere disponer de la frase de referencia, lo que implica la necesidad de etiquetar manualmente las frases mencionadas por el usuario.

- **Comprensión del habla:** Porcentaje de frases comprendidas correctamente, no comprendidas o parcialmente comprendidas, y un análisis de los errores; porcentaje de frases correctamente analizadas; porcentaje de palabras fuera del diccionario; porcentaje de frases sin cobertura lingüística; etc.

Al igual que en el caso del reconocedor, es necesario etiquetar cada concepto semántico generado por el módulo de comprensión como correcto o incorrecto (conceptos correctos, insertados, borrados y substituidos).

Para la evaluación de la traducción semántica final generada por los módulos de comprensión desarrollados en los proyectos BASURDE y DIHANA se definieron las siguientes medidas (Minker, Waibel, y Mariani, 1999):

- *fc*. Porcentaje de frases cuya representación semántica final es igual que la de referencia.

$$\%fc = 100 * \frac{\text{num. de frases con representacion en frame correcta en la hipotesis}}{\text{numero de frases}}$$

- *ufc*. Porcentaje de unidades de frame correctas, considerando unidades de frame el propio tipo de frame y cada uno de los valores de sus atributos.

$$\%ufc = 100 * \left(1 - \frac{\text{num. inserciones} + \text{num. substituciones} + \text{num. borrados}}{\text{numero de unidades de frame en la referencia}} \right)$$

- P_f . Precisión a nivel de frame.

$$\%P_f = 100 * \left(\frac{\text{numero de unidades de frame correctas en la hipotesis}}{\text{numero de unidades de frame en la hipotesis}} \right)$$

- C_f . Cobertura a nivel de frame.

$$\%C_f = 100 * \left(\frac{\text{numero de unidades de frame correctas en la hipotesis}}{\text{numero de unidades de frame en la referencia}} \right)$$

- **Gestión del Diálogo.** (análisis de las estrategias para recuperarse de errores, para corregir/dirigir la interacción del usuario, manejo del contexto cuando se producen múltiples preguntas-respuestas asociadas a un escenario) (% de respuestas correctas, % de respuestas incorrectas, % de respuestas a medias, % de veces que el sistema actúa intentando solucionar un problema, % de veces que el usuario actúa intentando solucionar un problema, etc.).
- **Generación de respuestas.** Las medidas deben tratar de evaluar si el usuario entiende y asimila correctamente las informaciones que le ofrece el sistema. Ejemplos de medidas utilizadas son:
 - Número de veces que el usuario solicita la repetición de la respuesta facilitada por el sistema.
 - Tiempo de respuesta del usuario (cuanto peor se formulen las preguntas del sistema o mayor información suministre, mayor será este tiempo).
 - Número de veces que el usuario no contesta.
 - Tasa de palabras fuera de vocabulario (indica si debe aumentarse el vocabulario del reconocedor o reformular las preguntas).

Esta evaluación suele realizarse habitualmente mediante el uso de opiniones subjetivas de los usuarios, utilizando preguntas que miden la relevancia, calidad, satisfacción global del usuario, etc.

- **Síntesis de voz.** En la evaluación de la síntesis de voz suelen considerarse dos aspectos fundamentales: la inteligibilidad de la voz sintética y la naturalidad de la voz. La inteligibilidad es un factor fundamental para asegurar el éxito del sistema. Hoy en día existen sintetizadores capaces de suministrar altas prestaciones en ambos parámetros.

El estándar SSML (Speech Synthesis Markup Language) recoge especificaciones relativas a la síntesis, como el control de la prosodia, el énfasis en palabras o frases, etc.

En (Dybkjaer y Bernsen, 2000) se recogen 15 criterios que deben evaluarse para garantizar la usabilidad del sistema: (1) uso adecuado de las modalidades, (2) reconocimiento de la entradas de usuarios, (3) naturalidad del habla del usuario, en relación con la cobertura del vocabulario y las gramáticas utilizadas, (4) calidad de voz del sistema, (5) generación de respuestas adecuada, (6) realimentación adecuada, (7) uso adecuado de la iniciativa del diálogo relativa a la tarea o tareas del sistema, (8) naturalidad de la estructura del diálogo para la tarea(s), (9) Cobertura suficiente del dominio, (10) capacidades de razonamiento del sistema suficientes, (11) guía o ayuda durante la interacción suficiente,

(12) tratamiento de errores adecuado, (13) adaptación suficiente a las diferencias entre usuarios, (14) número de problemas durante la interacción, (15) satisfacción del usuario.

El grupo de evaluación EAGLES (Expert Advisory Group on Language Engineering Standards) (EAGLES, 1996) propone medidas cuantitativas (tasa de finalización de la tarea, tasa de éxito de la transacción, tiempo de respuesta del sistema, concisión de las respuestas del sistema...) y cualitativas (satisfacción del usuario, capacidad de adaptación a nuevos usuarios, capacidad para manejar la multimodalidad...). Además de proponer qué evaluar, se busca también establecer cómo evaluar e informar de los resultados, fijando un conjunto de parámetros (del sistema, de condiciones del test, de resultados del test) que permitan una comparación homogénea.

En el proyecto DISC (Spoken Language Dialogue Systems and Components) (Failenschmid et al., 1999), en la misma línea que EAGLES, se propone qué evaluar (el conjunto de propiedades) y cómo evaluar (el criterio a aplicar). La metodología se basa en el uso de plantillas y ciclos de vida.

La evaluación propuesta en el proyecto francés EVALDA (Devillers, Maynard, y Rosset, 2004), orientado a la evaluación de las tecnologías del lenguaje desarrolladas para ese idioma, consiste en la utilización de conjuntos de test procedentes de corpus reales, una representación semántica del diálogo y métricas de evaluación comunes. No se han publicado todavía sus resultados.

En (Degerstedt y Jönsson, 2006) se describe LINTEST, una herramienta para la evaluación de sistemas de diálogo mediante la utilización de un corpus basada en JUNIT (para la evaluación del sistema se definen ficheros con casos que se ejecutan para comprobar el funcionamiento correcto del sistema). Permite dos modos de funcionamiento: *batch mode* (se realizan tests de evaluación que generan como salida un fichero log de resultados) y *interactive mode* (permiten test más exhaustivos a modo de traza del sistema). En el trabajo se describe la utilización de la herramienta para el desarrollo y la evaluación del sistema BIRDQUEST (información sobre las aves de los países nórdicos).

2.4. Simulación de usuarios

En años recientes, una cuestión de gran interés ha sido el desarrollo de herramientas y técnicas que faciliten la evaluación de los sistemas de diálogo basándose en la generación automática de conversaciones entre el sistema de diálogo, y un módulo adicional llamado simulador del usuario, que representa la interacción de los usuarios con el sistema de diálogo y cuyos modelos estadísticos pueden estimarse a partir de un corpus de diálogos usuario-máquina.

La investigación de técnicas de modelado del usuario tiene una larga historia dentro de los campos del procesamiento del lenguaje natural y los sistemas de diálogo hablados, y en particular en las áreas relativas al diseño de sistemas adaptados al usuario. Tradicionalmente, la mayoría de trabajos referentes al modelado de los usuarios se ha caracterizado por la utilización de aproximaciones no estadísticas. En estas aproximaciones, el objetivo es construir un modelo representativo del usuario que describa su estado durante la interacción con el sistema. Para ello, es necesario incorporar al modelo información referente al objetivo del diálogo y la historia del diálogo (intercambio de información que se ha producido entre el usuario y el sistema hasta el instante actual del diálogo). Además,

dependiendo de la tarea, puede ser necesario incluir en el modelo información referente a las preferencias del usuario, grado de conocimiento sobre la tarea o el propio sistema de diálogo, conocimientos o capacidades, nivel de satisfacción con respecto al comportamiento del sistema, etc. La finalidad de realizar modelos de usuario no reside en modelar las características precisas de cada uno de los usuarios, sino servir como herramienta en la fase de desarrollo de un nuevo sistema de diálogo.

En (Zukerman y Litman, 2001) se realiza una revisión de la aplicación de técnicas de modelado del usuario en los campos más representativos de los sistemas de diálogo, como la generación de lenguaje natural (GLN), comprensión del lenguaje natural (CLN) y gestión del diálogo (GD). En el caso de la gestión del diálogo, las primeras aplicaciones del modelado de usuario se basan en modelos complejos determinados por el conocimiento de la tarea. Actualmente, los modelos de usuario se han ido simplificando, de modo que el modelado de usuario para la gestión del diálogo se realiza muy frecuentemente mediante unas reglas sencillas o mediante pares atributos-valores.

Además, el modelado estadístico de usuarios reales se ha sugerido como solución a la falta de los datos necesarios para realizar el entrenamiento y el test en el aprendizaje automático de la gestión del diálogo, pero hasta la fecha pocos de los modelos propuestos se han evaluado cuantitativamente. De hecho, no hay criterios establecidos para realizar esta evaluación. El usuario simulado también permite explorar estrategias de diálogo no presentes en el corpus. De esta forma, el gestor del diálogo puede llegar a desviarse de las estrategias iniciales y aprender nuevas metodologías potencialmente mejores.

Estas técnicas de aprendizaje automático tienen como objetivo aprender estrategias de gestión del diálogo óptimas a partir de un corpus de diálogos utilizando métodos de “prueba y error” en lugar de basarse en principios de diseño empíricos (diseño a mano de la estrategia basándose en una serie de reglas). No obstante, el tamaño de los corpus etiquetados suele ser demasiado pequeño para explorar suficientemente el espacio global de posibles estados y estrategias del diálogo. Además, no existe una garantía de que la estrategia óptima pueda estar presente en el corpus de diálogos disponible, con lo que puede argumentarse que no puede aprenderse una estrategia óptima a partir de un corpus prefijado, independientemente de su tamaño. Una solución interesante para este problema consiste en entrenar un modelo de usuario probabilístico para simular las intervenciones del usuario y utilizarlas en el aprendizaje mediante la interacción entre el gestor del diálogo y el usuario simulado. En esta línea pueden referenciarse los trabajos (Eckert, Levin, y Pieraccini, 1997) (Scheffler y Young, 1999) (Pietquin y Beaufort, 2005) (Pietquin y Dutoit, 2005) (Schatzmann et al., 2005) (Schatzmann, Georgila, y Young, 2005) (Torres, Sanchis, y Segarra, 2005).

En este contexto, la simulación de usuarios se realiza en dos fases. En la primera, el modelo de usuario se entrena a partir de un corpus de diálogos para aprender qué respuestas proporcionaría un usuario real en una determinada situación del diálogo. En esta fase suelen utilizarse técnicas de aprendizaje supervisado. En la segunda fase, el modelo de usuario entrenado se utiliza para predecir la respuestas a las acciones del usuario. El sistema aprendido interactúa con el usuario y optimiza la estrategia del diálogo basándose en la realimentación ofrecida por el usuario simulado. De esta forma, pueden adquirirse tantos diálogos de aprendizaje como se desee y además permite estrategias del diálogo que no están presentes en el corpus inicial de diálogos persona-máquina. Por tanto, posibilita que el gestor del diálogo se desvíe de las estrategias conocidas y aprenda una

estrategia de gestión mejor.

A la hora de estimar el modelo de usuario, deben tenerse en cuenta tanto factores observables (por ejemplo, la historia del diálogo hasta el momento actual) como no observables (objetivo del usuario, memoria, preferencias, etc). Para la construcción del modelo de usuario, existe un gran número de referencias sobre diferentes aproximaciones y metodologías:

- **Modelos de n-gramas**

La utilización de modelos estadísticos para predecir la próxima acción del usuario se sugirió por primera vez en (Eckert, Levin, y Pieraccini, 1997) (Eckert, Levin, y Pieraccini, 1998). En estos trabajos se introducen un modelo de n-gramas para predecir la acción de usuario en el instante t que es más probable dado el historial anterior de acciones del sistema y del usuario. En la práctica, la casuística existente y la falta de las suficientes muestras de datos imposibilitan la utilización de toda la historia del diálogo. Eckert, Levin y Pieraccini aproximan la historia completa mediante un modelo de bigramas.

El modelo propuesto tiene la ventaja de ser puramente probabilístico y completamente independiente de la tarea. El punto débil del modelo radica en la no definición de restricciones en el comportamiento del usuario simulado. De este modo, cualquier acción del usuario puede ser válida tras una acción del sistema, a pesar de toda la historia previa del diálogo. Con la utilización de bigramas, la respuesta generada por el modelo puede corresponderse correctamente con la última respuesta del sistema, pero puede carecer de sentido teniendo en cuenta toda la historia previa del diálogo.

En (Levin, Pieraccini, y Eckert, 2000) se describe cómo el modelo puro de bigramas puede modificarse para representar de forma más realista la historia del diálogo (Modelo de Levin). En lugar de permitir que cualquier respuesta del usuario pueda seguir a una determinada acción del sistema, sólo se estiman las probabilidades de algunos pares {respuesta del usuario - acción del sistema}, mientras que todas las demás probabilidades se consideran nulas. El conjunto de parámetros del modelo probabilístico del usuario caracterizan el nivel de cooperación y el grado de iniciativa del usuario simulado.

Al igual que en el modelo de bigramas, el modelo de Levin no asegura la consistencia entre diferentes acciones del usuario a lo largo del transcurso del diálogo debido a la suposición de que la respuesta del usuario depende únicamente del último turno del sistema. Como en el caso del modelo de bigramas, las acciones del usuario pueden no cumplir restricciones lógicas del diálogo, haciendo que los diálogos continúen indefinidamente debido a que el usuario cambia continuamente de objetivo o repite información previamente suministrada.

- **Modelos basados en grafos**

Scheffler y Young proponen en (Scheffler y Young, 2001a), (Scheffler y Young, 2001b) (Scheffler y Young, 2000) (Scheffler y Young, 1999) un modelo basado en grafos como solución a la inconsistencia en los objetivos del usuario que presenta el modelo de Levin. Para ello, se combinan reglas deterministas para las acciones dependientes

del objetivo y un modelo probabilístico para cubrir el resto de acciones del usuario durante el diálogo.

En el modelo de Scheffler y Young, todos los posibles “camino” que puede tomar un usuario durante el diálogo deben representarse en forma de red. Los arcos de la red simbolizan acciones y los nodos representan “puntos de elección”. Se establece una división de los nodos en nodos de elección probabilísticos (el simulador de usuario toma una decisión aleatoria en base a las probabilidades estimadas a partir de un corpus de datos etiquetados) y nodos de elección determinista, donde la ruta a tomar al llegar a ellos depende del objetivo del usuario. Este objetivo permanece fijo durante todo el diálogo.

El principal inconveniente de este modelo es su alta dependencia del conocimiento que se disponga de la tarea, dado que un aspecto crítico del modelo es la especificación de todos los posibles caminos en el diálogo, que supone un gran esfuerzo manual. Esta tarea puede automatizarse, en parte, si existe un prototipo inicial del sistema del diálogo y el rango de posibles acciones del usuario se definen y acotan correctamente.

■ **Redes Bayesianas**

Pietquin, Beaufort y Dutoit combinan características del modelo de Scheffler y Young y del modelo de Levin, con el principal objetivo de reducir el esfuerzo manual de la construcción de las redes con los puntos de elección (Pietquin y Beaufort, 2005) (Pietquin y Dutoit, 2005). La principal idea de su trabajo se basa en condicionar el conjunto de probabilidades indicadas en el modelo de Levin teniéndose en cuenta el objetivo y la memoria del diálogo.

De forma similar al modelo de Scheffler y Young, el objetivo del diálogo es una simple tabla que contiene pares $\langle \textit{nombre del atributo}, \textit{valor del atributo} \rangle$ con variables de estado asociadas. Estas variables se utilizan para estimar la probabilidad del usuario para cada uno de los valores de cada campo y para detectar la frecuencia en la que el usuario ha mencionado una determinada información durante el transcurso del diálogo.

En este trabajo, Pietquin selecciona a mano todos los parámetros del modelo utilizando principios empíricos y el sentido común. Los valores de las probabilidades se seleccionan también a mano. Se sugiere la utilización de una red bayesiana para implementar y visualizar el modelo propuesto. Las variables de entrada a la red son los tipos de acciones del sistema (por ejemplo: bienvenida, confirmación, respuesta...) y los nombres de los atributos mencionados por el usuario. Las variables de salida son los nombres de los atributos y sus valores proporcionados por el usuario y una variable booleana que indica si el usuario finaliza la interacción con el sistema o no. El objetivo del usuario y la función que memoriza la información que ha mencionado se tratan como variables internas de la red.

■ **Técnicas de Aprendizaje Automático (Machine-Learning)**

Georgila, Henderson y Lemon proponen la utilización de modelos de Markov, realizando una descripción más detallada de los estados del modelo, historias del diálogo

más amplias y empleando técnicas de aprendizaje automático (Georgila, Henderson, y Lemon, 2005).

El diálogo se describe como una secuencia de Estados de Información (Bos et al., 2003), cada uno de los cuales viene representado por un vector de características que describe el estado actual del diálogo, la historia previa del mismo y cualquier posible respuesta del usuario válida para dicho estado. Aunque la riqueza de información del modelo ayuda a compensar la suposición de Markov, se requiere una gran cantidad de datos de entrenamiento para estimar de manera fiable los parámetros del modelo.

En su trabajo, se presentan dos métodos diferentes para predecir la próxima acción del usuario dada una historia de estados de información. El primer método reutiliza el modelo de n -gramas propuesto en (Eckert, Levin, y Pieraccini, 1997), pero utilizándose valores de n de 2 a 5 para cubrir una historia más amplia del diálogo. Se menciona que los mejores resultados se obtienen con 4-gramas, es decir, utilizándose una historia de 4 estados de información para predecir la próxima acción del usuario. El espacio de estados utilizado en las experimentaciones es del orden de 10^{87} estados, con lo que existe un gran número de secuencias de estados no disponibles en los datos de entrenamiento.

El segundo método se basa en la utilización de una combinación lineal de características para realizar la correspondencia entre un estado s y un vector de características con valores reales $f(s)$. La mayoría de estos valores son binarios, indicando la presencia o ausencia de una determinada información (por ejemplo: destino, fecha de salida...) y el resto de variables toman valores continuos (por ejemplo, el WER estimado). En total, se utilizan 290 variables (Georgila, Henderson, y Lemon, 2005). Se utilizan técnicas de aprendizaje supervisado para estimar el conjunto de pesos wa para cada acción a que describen lo apropiado de utilizar cada vector de $f(s)$ para predecir a . Una vez se estiman los pesos, se calcula una función $P(a|s)$ para cada una de las acciones. Para ello se aplica una función exponencial normalizada al producto de $f(s)$ y wa .

Dado que cada Estado de Información incluye no sólo el estado actual del diálogo, sino también información sobre la historia previa del diálogo, pueden modelarse los aspectos que pueden contribuir en el comportamiento del usuario para predecir su siguiente acción.

- **Modelado del usuario como un proceso de decisión de Markov (MDP).**

Destacan los trabajos (Eckert, Levin, y Pieraccini, 1997) (Biermann y Long, 1996) (Singh et al., 2002). Tal y como se ha comentado en la Sección 2.2.2, un MDP puede describirse formalmente como un espacio de estados finito S , un conjunto finito de acciones A , un conjunto de probabilidades de transición T y una función de recompensa R . En cada instante t , el gestor del diálogo se encuentra en un estado $s_t \in S$, ejecuta una acción discreta $a_t \in A$, transita a un nuevo estado de acuerdo a la probabilidad $p(s_{t+1}|s_t, a_t)$ y recibe una recompensa r_{t+1} . De este modo, el gestor del diálogo interactúa con el simulador de usuarios, realizando transiciones entre los diferentes estados del modelo a medida que realiza acciones como respuesta a las intervenciones del usuario. El gestor del diálogo recibe una recompensa por cada

acción que lleva a cabo, siendo la estrategia de diálogo óptima la que maximice las recompensas recibidas a lo largo del tiempo.

■ Modelos Ocultos de Markov (HMMs)

Cuayahuitl, Renals, Lemon y Shimodaira presentan un método para la simulación de diálogos basada en HMMs en el que se generan tanto las acciones del usuario como las del sistema (Cuayáhuitl et al., 2005). El principal objetivo es ampliar un corpus pequeño de diálogos persona-máquina con nuevos diálogos simulados.

Se proponen diferentes variaciones de HMMs. La más avanzada de ellas es un HMM de entrada-salida (IOHMM). El modelo se caracteriza por un conjunto de estados visibles $S = S_1, S_2, \dots, S_n$ y un conjunto de observaciones $V = V_1, V_2, \dots, V_m$. Los estados S representan turnos de sistema y las observaciones V se corresponden con el conjunto de acciones del sistema. El estado en el instante t se denota mediante q_t , siendo $a_{s,t}$ la acción del sistema en el instante t . Las respuestas del usuario se representan mediante un conjunto de intenciones $H = H_1, H_2, \dots, H_l$ y la acción del usuario en el instante t se denota mediante $a_{u,t}$. El comportamiento del modelo viene regido por un conjunto de probabilidades de transición $P(q_{t+1}|q_t, a_{s,t})$ y un conjunto de probabilidades de salida $P(a_{s,t}|q_t, a_{u,t-1})$. Las respuestas del usuario se predicen utilizando un conjunto de probabilidades que modelan al usuario $P(a_{u,t}|q_t, a_{s,t})$

En lugar de entrenar un único modelo IOHMM genérico para simular cualquier tipo de diálogo, los diálogos del corpus inicial se agrupan según objetivos, entrenándose un submodelo para cada uno de los objetivos y utilizándose un modelo de bigramas para estimar la secuencia de objetivos.

- En (Ito et al., 2006) se describe un simulador de usuario diseñado para la evaluación de sistemas de diálogo que utilizan VoiceXML. Las características que debe cumplir el sistema a evaluar son:
 - Debe estar basado en VoiceXML, lo que implica que todos los posibles comportamientos del sistema puedan escribirse en VoiceXML.
 - El control del diálogo debe restringirse a iniciativa del sistema, con solicitudes de atributo en atributo.
 - El sistema realiza una confirmación tras cada intervención del usuario.
 - Los resultados generados por el reconocedor deben poderse observar directamente mediante un programa externo al sistema (por ejemplo, ficheros log).
- Dentro del proyecto BASURDE se elaboró un simulador de usuarios (Torres, Sanchis, y Segarra, 2005) (Torres, 2006) consistente en dos módulos: un gestor de diálogo de usuario (UDM) y un generador de respuestas de usuario (URG). El UDM es un módulo simétrico al gestor del diálogo estocástico definido para BASURDE. Este módulo recibe frames de sistema, realiza operaciones de lectura y escritura en su propio registro histórico (UHR), lee y realiza transiciones en el mismo modelo utilizado por el gestor del diálogo, aplica un conjunto de reglas para seleccionar las transiciones adecuadas, y genera los frames de usuario. El URG es un módulo

simétrico al generador de respuestas del sistema. Recibe los frames de usuario y genera las correspondientes frases en lenguaje natural

Al principio de cada diálogo, el simulador de usuario lee los parámetros y objetivos del escenario simulado y almacena esta información en el UHR. Seguidamente, lee la versión estática del modelo (sDM), realiza la búsqueda del estado siguiendo los objetivos marcados, y genera el frame correspondiente. En cada turno de diálogo, el simulador de usuario realiza la lectura de los frames de sistema, compara estos frames con los posibles actos de diálogo de sistema, transita a un nuevo estado de sistema en el sDM, actualiza el UHR con los datos suministrados por el sistema, transita a un nuevo estado de usuario en el sDM, y genera los frames de usuario.

La evaluación de las técnicas de modelado de usuarios es todavía un campo en desarrollo. Típicamente se utilizan métodos adoptados de otros campos de investigación como la recuperación de información y el aprendizaje automático. Una primera clasificación consiste en dividir las técnicas en métodos directos de evaluación que utilizan medidas como la Precisión y la Cobertura, y métodos indirectos que emplean medidas como la Utilidad.

Los métodos directos evalúan el modelo de usuario midiendo la calidad de sus predicciones. Con la precisión y la cobertura se evalúa si las respuestas generadas por el modelo de usuario concuerdan con las que proporcionaría un usuario real en la misma situación del diálogo (Schatzmann et al., 2006). Para ello, se extrae una partición de entrenamiento y otra de test a partir de un corpus de diálogos usuario real-máquina, se realiza el entrenamiento del modelo de usuario a partir de la partición correspondiente y se evalúan las respuestas generadas por el simulador para el conjunto de muestras en la partición de test, teniendo en cuenta la historia del diálogo y el objetivo del usuario.

La Cobertura (C) mide cuántas acciones de la respuesta del usuario real se predicen correctamente en la respuesta simulada:

$$C = 100 * \frac{\text{Acciones predichas correctamente}}{\text{Total acciones en la respuesta del usuario real}} \quad (2.33)$$

La Precisión (P) mide la proporción de acciones correctas entre todas las disponibles en la respuesta dada por el usuario simulado. Una respuesta simulada suele considerarse correcta si contiene al menos una de las acciones proporcionadas en la respuesta del usuario real.

$$P = 100 * \frac{\text{Acciones predichas correctamente}}{\text{Total acciones en la respuesta simulada}} \quad (2.34)$$

Para realizar estudios comparativos de sistemas entre diferentes aproximaciones utilizando para ello una única estadística, suele calcularse la medida f :

$$f = \frac{2PR}{P + R} \quad (2.35)$$

En (Schatzmann et al., 2006) se recogen resultados de la precisión y la cobertura de diferentes modelos de usuario. Los valores máximos se sitúan alrededor del 35 %, mientras que la medida para el modelo de bigramas es del 20 %. Una crítica de estas medidas es que realizan una alta penalización a las acciones no vistas en la respuesta simulada,

aunque las respuestas generadas sean aceptables y compatibles perfectamente con las que podría proporcionar un usuario real. De este modo, dado que la estrategia del diálogo debe funcionar correctamente para todos los tipos posibles de respuestas de usuario, y no sólo para el usuario más probable, la evaluación del modelo únicamente teniendo en cuenta la Precisión y la Cobertura es insuficiente.

Una solución a esta problemática consiste en la adquisición de un mayor número de diálogos mediante la interacción del modelo de usuario simulado preferiblemente con el gestor de diálogo utilizado para la adquisición (misma estrategia de gestión) y la definición de medidas estadísticas que permitan realizar un estudio comparativo del corpus simulado con respecto al adquirido con usuarios reales. En (Schatzmann et al., 2006) y (Scheffler y Young, 2001b) se proponen un conjunto de medidas estadísticas para llevar a cabo este tipo de evaluación:

- Características del diálogo: Número medio de turnos de diálogo, número medio de acciones por turno de diálogo, ratio de acciones de sistema frente a las del usuario, etc.
- Estilo del diálogo: Frecuencia de los diferentes actos de diálogo, grado de cooperación del usuario (proporción de valores de slots proporcionados cuando así se requieren), proporción de acciones encaminadas a conseguir el objetivo del diálogo frente al resto de acciones...
- Tasa de éxito y eficiencia de los diálogos: Tasa de consecución de los diferentes objetivos, tiempos transcurridos para cada uno de los objetivos, etc.

De nuevo, cabe destacarse que mediante estas medidas únicamente puede esbozarse el comportamiento del simulador, no existiendo rangos de medidas que indiquen si el modelo de usuario es lo suficientemente realista. Además no hay ninguna garantía de que el diálogo simulado sea realista aunque el valor de estas medidas coincida con el caso del corpus real.

En (Georgila, Henderson, y Lemon, 2005) se introduce el uso de la Perplejidad (PP) para la evaluación del modelo de usuario, midiendo si los diálogos simulados contienen secuencias de acciones similares a las contenidas en los diálogos usuario real-máquina. La definición de la Perplejidad se basa en la entropía H :

$$PP = 2^{\hat{H}} \quad (2.36)$$

representando la entropía la cantidad de información no redundante proporcionada por cada nueva acción (estado) en media:

$$\hat{H} = -\frac{1}{m} \log_2 P(a_1, a_2, \dots, a_m) \quad (2.37)$$

donde $P(a_1, a_2, \dots, a_m)$ es un estimador de la probabilidad de que el simulador lleve a cabo la secuencia de acciones a_1, a_2, \dots, a_m .

En (Cuayáhuitl et al., 2005) la comparación entre el corpus simulado y el adquirido con usuarios reales se lleva a cabo entrenando un HMM con cada uno de los corpus y midiendo la semejanza entre ambos corpus en base a la distancia entre los dos HMM. La distancia definida se rige por la desigualdad Kullback-Leibler:

$$D(P, Q) = \frac{D_{KL}(P||Q) + D_{KL}(Q||P)}{2} \quad (2.38)$$

donde D_{KL} es la distancia entre las distribuciones de probabilidad P y Q .

En cuanto a los métodos indirectos de evaluación, el principal objetivo es medir la *Utilidad* del modelo de usuario en el contexto del funcionamiento del sistema completo. Usualmente, se trata de evaluar el funcionamiento de la estrategia de diálogo aprendida mediante el simulador. Esta evaluación, en la práctica, en lugar de realizarse comprobando el funcionamiento de la estrategia con usuarios reales, suele llevarse a cabo verificando el funcionamiento de la nueva estrategia mediante la nueva interacción con el simulador de usuario. De este modo, se compara la estrategia inicial (normalmente fijada a mano) con la aprendida con el simulador, reaprendiendo el modelo de usuario a partir de una partición del corpus que no haya sido utilizada en el aprendizaje del simulador utilizado en la obtención de la nueva estrategia. El principal problema de esta evaluación reside en la dependencia del corpus adquirido con respecto al modelo de usuario, lo que no permite detectar si la nueva estrategia se adapta únicamente al tipo de usuario disponible en el modelo.

2.5. Adaptación al usuario y/o a la tarea

El funcionamiento de un sistema de diálogo puede variar notablemente de un usuario a otro, o incluso, para un mismo usuario durante su interacción con el sistema. De este modo, un tema de especial interés actualmente es el desarrollo de sistemas de diálogo capaces de interactuar en diferentes dominios, diferentes entornos o con la posibilidad de adaptarse a los diferentes perfiles o preferencias del usuario.

Para tratar este aspecto se han desarrollado diferentes metodologías, que tienen como principal finalidad la adaptación de las estrategias utilizadas por el sistema para la interacción con el usuario y la confirmación de los datos aportados. A continuación se comentan algunos trabajos realizados en esta temática:

- En (Pargellis, Kuo, y Lee, 1999) se presenta un sistema para crear y gestionar aplicaciones adaptadas al usuario que requieran un interfaz vocal. El sistema está compuesto por cuatro módulos: el generador automático de diálogo (ADG, Automatic Dialogue Generator), el gestor del perfil (PM, Profile Manager), el gestor de información y servicios (ISM, Information Services Manager) y el gestor de diálogo (DM, Dialogue Manager).

El PM codifica las preferencias del usuario utilizando una representación de los servicios e informaciones en los que el usuario está interesado. De este modo, el usuario define una aplicación personalizada, en cuanto a los contenidos como al formato de presentación de la información. El módulo ADG genera un modelo de diálogo basado en estados finitos a partir de la información sobre la tarea descrita en un conjunto de tablas. El módulo DM utiliza comandos en lenguaje VIL (Voice Interface Language) para llevar a cabo la gestión de diálogo.

En trabajos posteriores (Pargellis, Kuo, y Lee, 2004) se presentan ampliaciones del módulo. Este módulo permite modificar la estructura del diálogo siguiendo las preferencias anotadas por el usuario en su fichero correspondiente, facilita los cambios en

la gestión de la tarea seleccionada e, incluso, la migración a nuevas tareas (dada la centralización de la información correspondiente a la misma en una serie de ficheros que consultan los diversos módulos del sistema).

- Los investigadores del MIT han desarrollado la utilidad Speech-Builder (Glass y Weinstein, 2001), diseñada para especificar la información lingüística específica del dominio, de manera que se facilite la creación o adaptación de sistemas para nuevas tareas. Las funciones que pueden llevarse a cabo utilizando esta herramienta son:
 - Editar las pronunciaciones en el módulo de reconocimiento del habla.
 - Configurar un fichero de gramática y un fichero de conversión de análisis a representaciones semánticas.
 - Diseñar un gestor de diálogo genérico, enfocado únicamente a tratar las situaciones que se producen en las consultas a las bases de datos.

En (Polifroni, Chungand, y Seneff, 2003) se resumen distintos trabajos llevados a cabo por el MIT para el desarrollo de gestores de diálogo genéricos capaces de realizar las funciones esenciales del diálogo y adaptables a un dominio específico mediante ficheros externos. Para facilitar la adaptación, se describe un método para la organización de la información específica del dominio, mediante la definición automática de categorías semánticas.

- En el sistema SENECA se desarrollaron dos tipos de interacción con el usuario, en las que se tiene en cuenta si se trata de un usuario novato o experto en el sistema. Estos modos de interacción influyen en las respuestas generadas por el sistema, de modo que al usuario experto se le proporcionan *prompts* menos detallados y concisos. Para la elección de un modo de interacción u otro se tiene en cuenta el número de errores de reconocimiento contabilizados en las intervenciones previas del usuario.
- En el gestor de diálogo desarrollado para AT&T Communicator se realiza un control de la iniciativa del diálogo teniendo en cuenta las intervenciones que va realizando el usuario. El sistema asume inicialmente condiciones de diálogo “normales”, concediendo la mayor iniciativa posible al usuario. Si el sistema detecta problemas (por ejemplo, solicitudes repetidas de un mismo atributo), va concretando las preguntas utilizando mensajes más concisos. Tras un número prefijado de intentos, el sistema selecciona un modelo de interacción en el que se pide la confirmación de toda la información que haya aportado el usuario hasta el momento actual del diálogo y seguidamente se siguen solicitando atributos.
- El gestor de diálogo desarrollado para el sistema TOOT (Litman y Pan, 2002) implementa tres posibles iniciativas del diálogo (iniciativa del usuario, del sistema o mixta) y tres posibles tipos de estrategias de confirmación (explícita, implícita o sin posibilidad de confirmación).

Se desarrollaron dos versiones del sistema, una de ellas adaptativa. Para llevar a cabo la adaptación al usuario durante el transcurso del diálogo se tienen en cuenta las estadísticas de reconocimiento anotadas en un corpus de 120 diálogos adquirido en experimentaciones previas con usuarios no expertos. Cada uno de los turnos de

usuario de este corpus se etiquetó utilizando una medida acústica (entre 0 y 1) que simboliza la semejanza de la frase reconocida con la verdaderamente mencionada por el usuario. A partir de esta medida, se clasificaron los diálogos en “buenos” o “malos” según el funcionamiento global del reconocedor. Además, se extrajeron 23 características que evalúan el diálogo teniendo en cuenta cinco categorías: confianza acústica, eficiencia del diálogo (por ejemplo, número de turnos), calidad o naturalidad del diálogo (por ejemplo, número de veces que el usuario solicita ayuda), parámetros experimentales (por ejemplo, la estrategia inicial de diálogo seleccionada) y léxicos (por ejemplo, léxico relativo al reconocedor). A partir de estas características se realizó el aprendizaje de un clasificador automático de los diálogos cuyo funcionamiento se rige por reglas. Mediante el uso de estas reglas, el gestor de diálogo decide de forma automática qué estrategia de diálogo es más conveniente para el usuario actual:

- El sistema se inicializa siempre con una iniciativa de usuario sin confirmaciones.
- Si aplicando las reglas comentadas a los últimos cuatro turnos del usuario se clasifica el diálogo como “malo”, la estrategia del diálogo pasa a ser iniciativa mixta con confirmaciones implícitas.
- Si tras realizar esta primera adaptación, el diálogo vuelve a clasificarse como “malo” tras 4 turnos, la estrategia vuelve a restringirse adoptando una iniciativa del sistema con confirmaciones explícitas.
- Mientras el diálogo se clasifique como “bueno”, no se realizan cambios en la estrategia del diálogo.

El sistema de diálogo desarrollado se evaluó comparativamente con una versión no adaptativa del sistema, midiéndose parámetros definidos en el modelo de PARADISE. Participaron 12 usuarios (6 en cada una de las versiones del sistema). Los porcentajes de éxito del diálogo son del 23 % en el sistema no adaptativo y del 65 % en la versión adaptativa, explicado en gran parte por los mejores porcentajes de reconocimiento. La opinión sobre el funcionamiento del sistema suministrados por los usuarios también es más favorable para la versión adaptativa.

- En el sistema HMIHY se realizó un estudio sobre la predicción automática de las situaciones problemáticas del diálogo a partir de la clasificación del diálogo teniendo en cuenta una serie de características (Walker et al., 2000):
 - Características acústicas: gramáticas utilizadas, número palabras reconocidas, etc.
 - Características de comprensión del lenguaje: medidas de confianza, cobertura, inconsistencia, etc.
 - Características de gestión de diálogo: tipo de tarea, número de confirmaciones, número de repeticiones de preguntas, número de subdiálogos, etc.
 - Características etiquetadas a mano: código del usuario, sexo, edad, modalidad utilizada, porcentajes de reconocimiento, etc.

- Características globales del diálogo: Número total de turnos, duración de las llamadas, etc.

En total se definieron 240 características, utilizadas para el etiquetado de un corpus de 4.774 diálogos cuya adquisición se realizó de forma supervisada por un Mago de Oz, a partir de las cuales se utilizaron técnicas de aprendizaje automático (programa RIPPER) para el aprendizaje de un modelo de clasificación. Las problemáticas a detectar eran que el usuario cuelgue antes de cumplir el objetivo del diálogo, situaciones en las que ha sido necesaria la intervención del Mago para reconducir el diálogo y fallos en la estrategia que imposibilitaron cumplir el objetivo.

La evaluación de esta metodología se llevó a cabo utilizando validación cruzada. Mediante estas características se mejoraba un 23 % la identificación de las problemáticas respecto al baseline (64 %).

- El gestor de diálogo del sistema MIMIC dispone de una estrategia de iniciativa mixta adaptativa. MIMIC adapta automáticamente la estrategia del diálogo basándose en las características del diálogo, modelando su comportamiento teniendo en cuenta el contexto del diálogo. Además, el módulo de iniciativa de MIMIC está desacoplado de los objetivos del gestor y de los procesos de selección de las estrategias, lo que permite una adaptación más sencilla a otros dominios. Este módulo determina el grado de iniciativa basándose en el comportamiento del usuario, indicaciones que puedan existir en su turno actual y la historia del diálogo.

El comportamiento o papel del usuario hace que el gestor emplee iniciativas basadas en el sistema para usuarios no familiarizados con el sistema. En este grado de iniciativa, el sistema va solicitando al usuario un único dato en cada turno y acompaña su respuesta con una explicación de las acciones que puede llevar a cabo el usuario tras cada respuesta del sistema. Teniendo en cuenta la historia del diálogo, el gestor utiliza el comportamiento global del usuario para decidir si es necesario realizar un cambio en la estrategia del diálogo.

En (Chu-Carroll y Nickerson, 2000) se presentan los resultados de la evaluación de la versión adaptativa del sistema de diálogo con respecto a dos versiones no adaptativas del mismo sistema (una con iniciativa del sistema y otra con iniciativa por parte del usuario). Los resultados, obtenidos mediante encuestas al usuario y medición de las estadísticas de los diálogos adquiridos, muestran que la versión adaptativa posee un mejor comportamiento en términos de satisfacción del usuario, eficiencia del diálogo (número de turnos) y calidad del mismo (mejores tasas de reconocimiento).

- El sistema CU FOREX (Meng, Wai, y Pieraccini, 2003) permite dos modalidades de interacción dependiendo de la destreza del usuario:
 - *Directed dialog* (DD): Está diseñado para usuarios novatos. El sistema se encarga de guiar al usuario de forma detallada, indicándole incluso en algunos estados qué valores concretos debe aportar. En cada turno del diálogo, el sistema solicita un único atributo.

- *Natural Language Shortcut* (NLS): Está diseñado para usuarios expertos que desean realizar toda la consulta en un único turno, aportando toda la información necesaria en dicho turno.
- El sistema AthosMail (Jokinen, Kanto, y Rissanen, 2004), desarrollado en el ámbito del proyecto europeo DUMAS y cuya tarea es la consulta del correo electrónico, contiene un módulo de modelado del usuario, encargado de realizar la adaptación de las respuestas del sistema a los diferentes niveles de destreza de los usuarios. Los objetivos establecidos para el desarrollo de este módulo fueron:
 - Proveer flexibilidad y variabilidad en las respuestas del sistema.
 - Permitir al usuario la interacción con el sistema de una forma más natural.
 - Posibilitar a los desarrolladores la implementación y evaluación de técnicas de aprendizaje automático.

Para cumplir estos objetivos, en el diseño del modelo de usuario se tuvieron en cuenta las siguientes directrices:

- Utilizar una representación flexible para codificar los turnos de sistema, utilizada para su generación.
- El sistema posee la funcionalidad de almacenar las acciones del usuario y su comportamiento, estimando así los niveles de decisión que se utilizan para proporcionar recomendaciones al usuario.
- El sistema, además, incorpora un módulo aprendido automáticamente que realiza la clasificación de los mensajes de correo del usuario en función del contenido y preferencias anotadas del usuario.

Los componentes que conforman el módulo de modelado de usuario son:

- Priorizador de mensajes: Clasifica los mensajes de correo utilizando una lista en la que los detectados como más interesantes o prioritarios para el usuario se sitúan en la parte superior. Para ello, se tienen en cuenta las acciones llevadas a cabo previamente por el usuario para asignar unas medidas de prioridad a los nuevos mensajes.
- Averiguador del objetivo: Este componente sugiere al gestor qué objetivo posible persigue el usuario en aquellas situaciones en las que existe incerteza en los datos proporcionados por el reconocedor.
- Modelo cooperativo: Este modelo permite al sistema variar la iniciativa y complejidad de sus respuestas dependiendo del nivel de destreza anotado para el mismo, así como de las estadísticas de reconocimiento obtenidas de sus sesiones anteriores. Se definieron cuatro niveles de iniciativa para realizar el control del diálogo: directiva (control por parte del sistema), modo sugerencia (el sistema lleva el control del diálogo, pero está preparado para cambiar el curso del diálogo de acuerdo con las preferencias indicadas por el usuario), modo declarativo (el usuario tiene la iniciativa, pero el sistema puede solicitarle información

requerida), modo pasivo (el usuario tiene el control completo del diálogo, no realizando ningún tipo de sugerencias el sistema).

- Categorización de mensajes: Este componente se utiliza para comparar los mensajes entrantes con los existentes, y categorizarlos utilizando palabras claves detectadas en los mismos.
- Preferencias del usuario: Se almacenan características relativas al estilo de habla preferido por el usuario, velocidad y tipo de voz, remitentes y temas preferidos, etc.

La evaluación realizada del sistema se expone en (Jokinen et al., 2004). Se llevaron a cabo dos tipos de evaluaciones. En la primera de ellas, cinco expertos del diseño de sistemas interactivos proporcionaron su opinión sobre las diferentes modalidades de adaptación al usuario contempladas en el sistema. Para el segundo estudio se recogen las opiniones y estadísticas obtenidas tras el análisis de 104 diálogos adquiridos por 26 usuarios no familiarizados con el sistema, a partir de los cuales se identificaron una serie de errores que cometía el sistema (no detección de frases fuera de la tarea, solapamientos entre las respuestas del sistema e intervenciones del usuario, etc.).

- En (Giesemann y Waibel, 2006) se describe un gestor de diálogo desarrollado para un robot doméstico con la tarea de proporcionar recetas de cocina. La implementación del gestor se ha realizado mediante las herramientas TAPAS (Holzapfel, 2005), creadas para la elaboración del gestor de diálogo independiente del dominio y del idioma diseñado para el proyecto ARIADNE. Mediante estas herramientas se facilita la implementación rápida de prototipos dado que sólo es necesario desarrollar aquellos componentes dependientes del dominio. De este modo, permiten el desarrollo de gestores de diálogo independientes de la tarea y del idioma, separando los contenidos propios del dominio de los independientes del mismo.

Capítulo 3

La tarea DIHANA

3.1. Introducción

En este capítulo se describen las principales características de la tarea definida en el marco del proyecto DIHANA para el desarrollo de un sistema de diálogo hablado. El trabajo llevado a cabo para la definición de la tarea y la adquisición y etiquetado de un corpus de diálogos ha dado lugar a las siguientes publicaciones científicas (Alcácer et al., 2004c) (Alcácer et al., 2004b).

La tarea seleccionada para el sistema DIHANA consiste en el acceso a un sistema de información al que se le solicitan consultas telefónicas sobre trayectos en trenes regionales y de grandes líneas en el territorio español. El idioma utilizado para interactuar con el sistema es el castellano. Esta tarea también fue elegida en el proyecto BASURDE, antecesor de DIHANA. Las principales características de la tarea son:

- Tarea restringida semánticamente. Esta característica, conjuntamente con la anterior, permite abordar de manera adecuada la tarea de comprensión del lenguaje natural.
- Estructura del diálogo relativamente rica y abierta. De esta forma, la complejidad de la estructura del diálogo permite la selección de diferentes estrategias para el desarrollo de módulos de gestión del diálogo.

Los tipos de consulta definidos para la tarea son:

1. Horarios de salida o llegada de trenes;
2. Precios de ida o vuelta en trenes en general o bien con recargo por determinadas clases de billete;
3. Tiempo de recorrido de trenes;
4. Tipos de tren;
5. Servicios disponibles en los trenes.

Las siguientes consultas no están soportadas por el sistema:

- peticiones de reservas o compra de billetes;
- consultas sobre trenes de cercanías o internacionales;
- consultas sobre precios con descuento por fecha, grupos, bonos, etc.;
- consultas sobre dónde para un tren determinado

La tarea es similar a las seleccionadas en proyectos de investigación europeos como el proyecto ARISE del LIMSI, TABA de Philips o AMICA de AT&T.

3.2. Objetivos del proyecto DIHANA

El objetivo principal del proyecto DIHANA (diciembre 2002 - noviembre 2005) fue el estudio y desarrollo de un sistema robusto de diálogo para el acceso a sistemas de información mediante habla espontánea en diferentes entornos, objetivo que dio nombre al proyecto. La principal característica que debía poseer el sistema desarrollado era su elaboración mediante una estructura modular y distribuida que permitiese la disposición de los diferentes interfaces que conforman el sistema en diferentes máquinas y, por tanto, diferentes sedes geográficas. El capítulo cuarto de la tesis se dedica a la descripción del sistema de diálogo desarrollado para el proyecto. El objetivo del proyecto se concretó en dos marcos diferenciados:

- El estudio y desarrollo de metodologías para el modelado acústico en diferentes entornos, el modelado del lenguaje, el habla espontánea, la comprensión y el diálogo.
- La extensión del proyecto a otros ámbitos de aplicación, mediante el estudio y desarrollo de “front-ends” para entornos de telefonía móvil en automóvil.

Los principales objetivos científicos definidos en el proyecto fueron:

- Propuesta de metodologías que mejorasen las prestaciones de los sistemas de reconocimiento del habla en ambientes adversos (técnicas de realce de la señal, utilización de técnicas de adaptación/compensación de modelos acústicos, estudio y minimización del efecto Lombard).
 - Integración de los sistemas de reconocimiento de habla con soluciones que posibiliten la detección rápida de intervenciones por parte del usuario mientras el sistema está reproduciendo un mensaje.
 - Integración de los fenómenos asociados al habla espontánea en los distintos módulos de un sistema de diálogo, a nivel acústico, léxico y sintáctico.
 - Estudio del fenómeno de las disfluencias a nivel léxico, sintáctico y semántico.
 - Diseño de nuevos modelos de comprensión con técnicas de aprendizaje automático de modelos estocásticos de estados finitos y modelos de redes neuronales.
 - Desarrollo de nuevos modelos de diálogo estocásticos y modelos basados en redes de estados finitos.
-

En cuanto a los objetivos tecnológicos, se remarcaron tres líneas de trabajo fundamentales:

- Corpus de diálogos.
 - Adquisición y transcripción automática de un gran corpus oral de diálogo en lengua castellana, puesto a la disposición de la comunidad científica a la conclusión del proyecto.
 - Etiquetado del corpus anterior en términos de: unidades del habla espontánea, unidades semánticas y actos del diálogo.
 - Adquisición y etiquetado ortográfico de un corpus oral en el interior del automóvil.
- Entrada/salida de audio.
 - Desarrollo de un sistema manos libres-CCCS para la entrada/salida de audio del sistema de diálogo en el automóvil.
 - Adaptación on-line de los modelos acústicos del sistema de reconocimiento para su utilización conjunta con el sistema manos-libres.
 - Desarrollo de la cancelación activa de ruido para combatir el efecto Lombard.
- Sistema de diálogo.
 - Mejora del reconocedor de voz para su adaptación acústica a diferentes entornos.
 - Evolución de la arquitectura distribuida del sistema de diálogo: definición de un formato de comunicaciones entre los diferentes módulos del sistema, estudio de diferentes estándares (tipo VoiceXML) para la programación de servicios y comunicaciones, etc.
 - Mejora de los protocolos de comunicación (reconocimiento-comprensión y comprensión-diálogo).
 - Utilización de múltiples hipótesis en los diferentes módulos.
 - Desarrollo de un prototipo que integrase tanto los resultados tecnológicos como científicos.

Como resultados científicos tras la finalización del proyecto cabe destacar la publicación de las aportaciones en las diferentes líneas de trabajo en forma de informes técnicos y en más de 50 publicaciones científicas en revistas y congresos relevantes en el área.

Diferentes aspectos metodológicos en los que se profundizó se centraron en los campos del modelado acústico en diferentes entornos, tratamiento del habla espontánea, modelado del lenguaje, comprensión del lenguaje y gestión del diálogo.

Además se definieron aspectos tecnológicos y metodológicos para extender esta propuesta a otros ámbitos de aplicación diferentes, como la representación robusta de la voz, el acceso manos libres en entorno del automóvil y la adaptación de los modelos acústico-fonéticos.

En el proyecto participaron la Universidad del País Vasco, la Universidad de Zaragoza y la Universitat Politècnica de València.

3.3. Adquisición del corpus DIHANA

Tal y como se ha comentado en el apartado anterior, uno de los objetivos tecnológicos del proyecto DIHANA fue la adquisición de un corpus oral de diálogos en lengua castellana (Benedí et al., 2006) (Castro et al., 2004). La adquisición se planteó como una ampliación del corpus de diálogos adquirido en el proyecto BASURDE, definiendo nuevos escenarios y desarrollando una plataforma para realizar la adquisición.

La adquisición del corpus de diálogos tuvo como principal propósito llevar a cabo el posterior entrenamiento de los diferentes modelos estocásticos que caracterizan a un sistema de diálogo: modelos acústicos, léxicos y del lenguaje, y modelos de comprensión y diálogo. En estas metodologías, los parámetros de los modelos estocásticos se aprenden (estiman) automáticamente a partir de ejemplos reales de diálogos (el corpus adquirido), convenientemente procesados y etiquetados.

El corpus, con un total de 900 diálogos, fue adquirido mediante la técnica del Mago de Oz (Fraser y Gilbert, 1991), en la que una persona emula el comportamiento del gestor de diálogo siguiendo una estrategia prefijada. Para llevar a cabo la adquisición, y posterior análisis de los diálogos adquiridos, en condiciones controladas se utilizaron dos elementos fundamentales:

- Se elaboró una estrategia que el Mago debía seguir durante la adquisición. Las principales características de esta estrategia se muestran seguidamente en el Apartado 3.3.4 de este capítulo.
- Se diseñaron unos escenarios que indicaban al informante una situaciones concretas para el diálogo: un objetivo (la información que debe obtener el informante), una situación que motiva el interés en la información y un esquema que condiciona la petición.

Para la adquisición del corpus se utilizó un sistema de diálogo completo (módulo de reconocimiento automático del habla, módulo de comprensión semántica y módulo de síntesis de texto a voz), en el que el Mago de Oz desempeñaba las funciones del gestor del diálogo.

3.3.1. Tipos de escenarios definidos

Cada uno de los usuarios que participaron en la adquisición debían completar cuatro escenarios, en los que era necesario que el sistema suministrase la información marcada como objetivo. Los escenarios definidos pueden ser clasificados en tres categorías (A, B, o C) dependiendo del objetivo que describen:

- El tipo A tiene por objetivo obtener el horario de trenes de viajes de ida. En la consulta es necesario especificar el destino y la fecha del viaje; el origen, la hora y el tipo de tren pueden estar especificados o no; el tipo de tren puede ser motivo de consulta o no. La Figura 3.1 muestra un ejemplo de escenario tipo A.
- Los escenarios tipo B tienen por objetivo obtener el precio y, opcionalmente, el horario de trenes de viajes de ida. En la consulta es necesario especificar el destino y la fecha del viaje; el origen, la hora y el tipo de tren pueden estar especificados o no. La Figura 3.2 muestra un ejemplo de escenario tipo B.

Objetivo: obtener horario
Tipo de viaje: viaje de ida
Situación: usted quiere ir a ver un partido de fútbol en Barcelona
Punto de salida: Valencia
Punto de llegada: Barcelona
Restricciones: el partido es el próximo sábado a las 20:30 horas.
Preferencias: -

Figura 3.1: Ejemplo de escenario tipo A

Objetivo: obtener horario; obtener precio;
Tipo de viaje: viaje de ida
Situación: asistir a un concierto
Punto de salida: Castellón
Punto de llegada: Barcelona
Restricciones: viajar el sábado próximo;
en Euromed, si es posible
Preferencias: -

Figura 3.2: Ejemplo de escenario tipo B

- Los escenarios de tipo C son análogos a los de tipo B, pero se refieren a trayectos de ida y vuelta. La Figura 3.3 muestra un ejemplo de escenario tipo C.

Objetivo: obtener precio;
Tipo de viaje: viaje de ida; viaje de vuelta;
Situación: puente de la Constitución
Punto de salida: -
Punto de llegada: Cádiz
Restricciones: salir el 5 de diciembre;
regresar el día 8
Preferencias: tren con literas o coche cama;

Figura 3.3: Ejemplo de escenario tipo C

Las variables que definen la situación del escenario (horario, destino, origen...) pueden ser instanciadas en la propia descripción de la situación (*escenarios cerrados*), o no mencionarse implícitamente en el escenario (*escenarios abiertos*). La Figura 3.4 muestra un ejemplo de un escenario abierto y de un escenario cerrado.

Para cada tipo de escenario se definieron 10 objetivos concretos. Para cada uno de estos objetivos se establecieron 12 variantes (8 cerradas y 4 abiertas), en el caso de los escenarios tipo A y tipo B; y 6 variantes (4 cerradas y 2 abiertas), en el caso de los escenarios tipo C. De este modo, se obtuvieron un total de 300 escenarios distintos.

Para la adquisición se decidió que el número de escenarios que debía completar cada usuario fuese cuatro. En la distribución de los escenarios entre los informantes se respe-

<i>Objetivo:</i> obtener horario	<i>Objetivo:</i> obtener horario; obtener tipo de tren
<i>Tipo de viaje:</i> viaje de ida	<i>Tipo de viaje:</i> viaje de ida
<i>Situación:</i> viaje de trabajo	<i>Situación:</i> visita a un amigo
<i>Punto de salida:</i> -	<i>Punto de salida:</i> -
<i>Punto de llegada:</i> elegir una ciudad	<i>Punto de llegada:</i> Granada
<i>Restricciones:</i> elegir día de salida;	<i>Restricciones:</i> salir el próximo fin de semana
<i>Preferencias:</i> prefiere viajar de día.	<i>Preferencias:</i> -

Figura 3.4: Ejemplo de escenario abierto (izquierda) y cerrado (derecha)

taron las siguientes condiciones:

- los escenarios adquiridos por un informante debían corresponderse, como mínimo, a dos tipos diferentes;
- cada informante debía realizar escenarios cerrados y abiertos.

Además de realizar los escenarios, cada informante debía adquirir un subconjunto de frases acústicamente balanceadas (8 frases de un total de 200 frases distintas) y un subconjunto de frases de la tarea (8 frases de un conjunto de 1.800), generadas de forma sintética utilizando modelos sencillos de categorías. El objetivo de la adquisición de estas frases fue la estimación de los modelos acústicos y los modelos de lenguaje.

La Sección A.1 del Anexo 1 muestra un ejemplo de las fichas que debían completar los informantes.

3.3.2. La plataforma de adquisición de diálogos

Tal y como se ha comentado previamente, para la adquisición del corpus DIHANA se utilizó un sistema de diálogo completo (reconocedor automático del habla, módulo de comprensión semántica, generador de respuestas y sintetizador del habla), en el que el Mago de Oz desempeñaba las funciones del gestor del diálogo.

La plataforma desarrollada en el marco del proyecto para realizar la adquisición de diálogos (Miguel et al., 2003) estaba compuesta por diferentes componentes: un gestor de comunicaciones (línea telefónica), un sistema de parametrización y reconocimiento automático del habla, un gestor del diálogo (simulado por el Mago de Oz), un sistema de comprensión y un sistema de síntesis de voz. Todos los componentes, a excepción del gestor del diálogo, realizan automáticamente su tarea y, en gran parte, eran resultado del trabajo desarrollado para el proyecto BASURDE.

La interacción del usuario con la plataforma con la finalidad de adquirir un diálogo seguía los siguientes pasos:

- El usuario va realizando consultas telefónicas al sistema de manera que el Mago escucha lo que realmente ha mencionado el usuario.
- El reconocedor del habla recibe la señal vocal del usuario, proporcionando como salida la secuencia de palabras que con mayor probabilidad ha mencionado el usuario, conjuntamente con un valor numérico (medida de confianza) con el que se mide la fiabilidad con la que se ha reconocido dicha secuencia.

- El módulo de comprensión genera como salida la representación semántica de la frase reconocida, es decir, obtiene a partir de la frase reconocida la información que es relevante en el ámbito de la tarea.

Dado un estado del diálogo y la información proporcionada por el usuario, el Mago procesa esta información, la entiende, actualiza un registro que contiene la historia del diálogo y selecciona una de las posibles respuestas que la plataforma le indica teniendo en cuenta las acciones que ha realizado el Mago. La interacción con el usuario se lleva a cabo haciendo uso de un módulo de generación de respuestas escritas, que posteriormente son comunicadas a través de un sintetizador. El Mago supervisa la información que se genera durante el proceso del diálogo, la modifica si fuera necesario, e interacciona con el usuario siguiendo la estrategia definida.

En la Figura 3.5 puede verse el aspecto del interfaz gráfico de plataforma de adquisición que se desarrolló. Este interfaz comprende las diferentes fuentes de conocimiento que se han mencionado previamente.

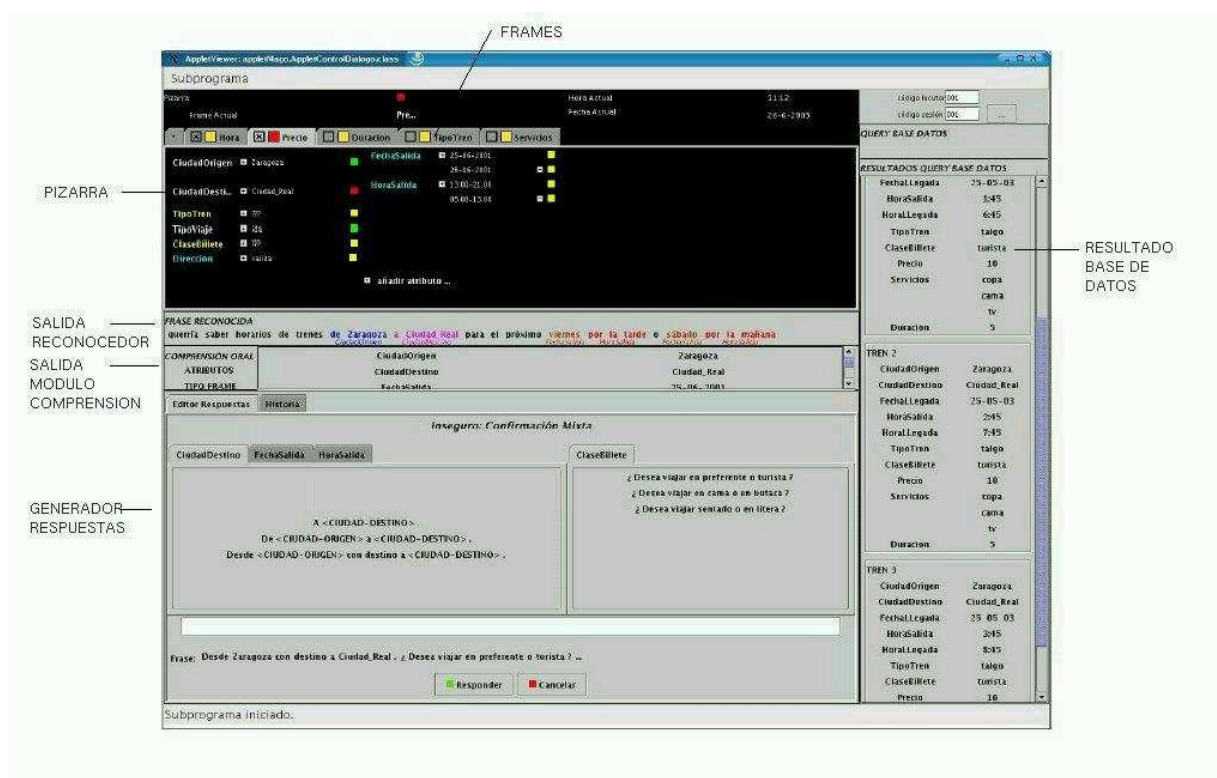


Figura 3.5: Plataforma de adquisición de diálogos (interfaz generado para el Mago de Oz)

3.3.3. Fases de la adquisición

La adquisición del corpus DIHANA se llevó a cabo en diferentes fases dependiendo del grupo al que pertenecían los informantes. Las diferentes fases de la adquisición se llevaron a cabo paralelamente en tres sedes (Bilbao, Valencia y Zaragoza). En cada una de las sedes se utilizó un Mago de Oz distinto, que empleó la misma estrategia y plataforma desarrolladas para la adquisición.

Los primeros usuarios que adquirieron fueron investigadores del proyecto DIHANA, con la finalidad de realizar pruebas iniciales, conocer cómo aplicaban la estrategia los Magos, y realizar una primera evaluación del sistema de adquisición, previa a la adquisición con usuarios externos.

Posteriormente adquirieron usuarios externos al proyecto. Todos ellos pensaban que estaban directamente adquiriendo a través de un sistema de diálogo completamente automático, sin la intervención de un Mago de Oz.

Las sesiones de la adquisición tenían una duración aproximada de entre 25 minutos y 50 minutos. En primer lugar, el usuario debía realizar una llamada para proceder a la grabación de las 16 frases acústicamente balanceadas. Si la grabación se realizaba correctamente, al terminar la última frase el sistema interrumpía la comunicación, informando al usuario que el proceso se había llevado a cabo de forma satisfactoria.

Finalizada la primera parte, y después de que el voluntario se hubiese preparado los escenarios, se efectuaban cuatro llamadas, tantas como escenarios había que grabar. La complejidad de los escenarios iba de menor a mayor, con motivo de que el usuario se familiarizara con el sistema. Durante el desarrollo de la grabación, un colaborador se encargaba de supervisar que se fueran cumpliendo los escenarios. No obstante, era el Mago de Oz el encargado de aceptar o rechazar la grabación de un escenario por el incumplimiento de algún objetivo, ya que, entre sus labores constaba también la supervisión del proceso.

Terminada la grabación de los cuatro escenarios, y comprobado que el proceso se había llevado correctamente para cada uno de los cuatro escenarios, el usuario participante en la adquisición completaba una encuesta en la que se le solicitaba su opinión sobre su experiencia utilizando el sistema.

3.3.4. La estrategia del Mago de Oz

En la estrategia definida para la adquisición, el Mago de Oz interacciona con el usuario teniendo en cuenta los niveles de confianza suministrados por el sistema, la información proporcionada por el usuario en el turno correspondiente y el estado de una estructura de datos que denominamos registro del diálogo (*dialog register (DR)*).

El registro del diálogo se define como una estructura de datos que contiene la información sobre los valores y medidas de confianza de los conceptos y atributos suministrados por el usuario a través de la historia previa del diálogo.

Si todos los datos del *DR* disponen de una medida de confianza asociada mayor que el umbral fijado (estado seguro), el gestor elige una de las tres interacciones siguientes:

- Confirmación Implícita y Consulta a la base de datos si el *DR* dispone del valor de uno de los conceptos y, al menos, de los valores de sus atributos mínimos. (Ej. *Le consulto horarios de trenes con salida en Valencia destino Zaragoza para el viernes 22 de septiembre.*)
- Solicitud de información al usuario si el *DR* no dispone de ningún concepto y/o de alguno de sus atributos mínimos (sin valor por defecto, en DIHANA, la ciudad de origen y la clase del billete). (Ej. *¿Me pregunta por horarios?*)
- Confirmación Mixta. (Ej. *Quiere horarios a Barcelona , ¿saliendo desde Granada?*). En las confirmaciones mixtas se incorporan referencias no únicamente a valores marcados como poco fiables (*Granada* en el ejemplo anterior), sino también a uno o más

conceptos con suficiente fiabilidad (*solicitud de horarios* y *Barcelona*), favoreciéndose una mayor naturalidad en el diálogo. Se realiza sobre el 30 % de turnos seguros en lugar de una Confirmación Implícita-Consulta.

Si el estado es inseguro (aquel en el que uno o más datos del *DR* poseen una confianza menor que el umbral), el gestor selecciona una de las dos interacciones siguientes:

- Confirmación Explícita del primero de los ítems inciertos que aparezca en el *DR*. (Ej. *¿Quiere viajar en Talgo?*)
- Confirmación Mixta para darle naturalidad al diálogo. Se realiza sobre el 30 % de turnos de diálogo inseguros en lugar de una Confirmación Explícita.

La Figura 3.6 resume los estados definidos en la estrategia del Mago de Oz definida en DIHANA. En (Galiano, 2004) puede encontrarse más información sobre la estrategia.

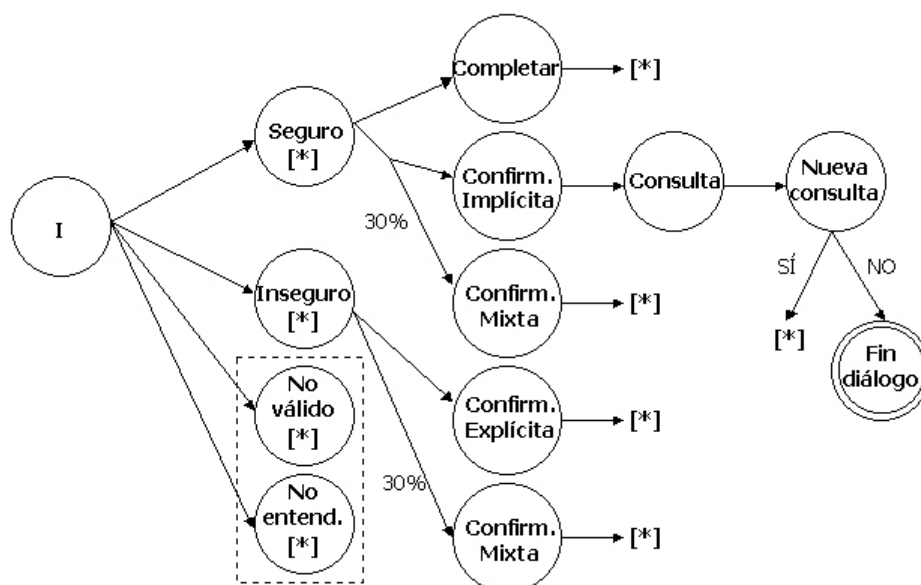


Figura 3.6: Diagrama que muestra los estados definidos para la estrategia del Mago de Oz en DIHANA

3.3.5. Ficheros almacenados durante la adquisición

Durante la adquisición de cada uno de los diálogos se utilizaron una serie de ficheros para almacenar cada una de las fuentes de información del mismo. El formato seleccionado para realizar este almacenamiento fue XML. De este modo, se generaron una serie de etiquetas específicas para almacenar las diferentes fuentes de información.

En total, por cada diálogo se generaron seis ficheros, cuya extensión indica la información que almacenan:

- *Ficheros .raw*: Almacenan la señal de audio del usuario, guardándose cada uno de las intervenciones del usuario en un fichero diferente.

- *Ficheros .fra*: Almacenan las frases acústicamente balanceadas que debían pronunciar los usuarios, previamente a la adquisición del corpus de la tarea. El Apartado A.2.1 del Anexo A muestra un ejemplo del fichero de frases.
- *Fichero .rec*: Contiene únicamente las frases generadas por el reconocedor tras cada uno de los turnos de usuario. El Apartado A.2.2 del Anexo A muestra un ejemplo del fichero del diálogo.
- *Fichero .com*: Almacena la salida generada por el módulo de comprensión semántica a partir de la frase proporcionada por el reconocedor tras cada intervención del usuario. El Apartado A.2.3 del Anexo A muestra un ejemplo del fichero de comprensión.
- *Fichero .dia*: Almacena el diálogo en formato texto (turnos de usuario y de sistema). Los turnos de usuario se corresponden con la salida proporcionada por el reconocedor. El Apartado A.2.4 del Anexo A muestra un ejemplo del fichero del diálogo.
- *Fichero .que*: Indica, para cada turno de sistema, si se produjeron accesos a la base de datos de trenes para responder al usuario sobre una determinada consulta. El Apartado A.2.5 del Anexo A muestra un ejemplo del fichero de accesos a la base de datos.
- *Fichero .piz*: Almacena el registro del diálogo actualizado por el Mago tras cada intervención del usuario. El Apartado A.2.6 del Anexo A muestra un ejemplo del fichero del registro del diálogo.

3.3.6. Evaluación recibida de los usuarios

La encuesta que debía completar cada usuario comprendía dos bloques diferenciados de preguntas. El primer bloque contenía una serie de preguntas referentes a la interacción del usuario con el sistema:

- ¿Entendió al sistema cuando éste hablaba?
- ¿El sistema comprendió lo que usted le decía?
- ¿Fue adecuado el ritmo de la interacción?
- ¿Supo usted cómo actuar en cada momento del diálogo?
- ¿Con qué frecuencia el sistema fue lento en su respuesta?
- ¿El sistema se comportó del modo esperado durante la conversación?

Las opciones proporcionadas al usuario eran: *Siempre, Habitualmente, La mayor parte de las veces, A veces, Nunca*.

La Tabla 3.1 muestra las estadísticas obtenidas de las respuestas a las preguntas anteriores dadas por los usuarios que participaron en la adquisición del corpus DIHANA. La Sección A.3.2 del Anexo A muestra gráficas explicativas de los resultados obtenidos,

¿Entendió al sistema cuando éste hablaba?				
Siempre	Habitualmente	La mayor parte de veces	A veces	Nunca
72,51 %	20,85 %	6,64 %	0 %	0 %
¿El sistema comprendió lo que usted le decía?				
Siempre	Habitualmente	La mayor parte de veces	A veces	Nunca
4,74 %	57,82 %	33,18 %	4,27 %	0 %
¿Fue adecuado el ritmo de la interacción?				
Siempre	Habitualmente	La mayor parte de veces	A veces	Nunca
9 %	44,55 %	28,44 %	15,17 %	2,84 %
¿Supo usted cómo actuar en cada momento del diálogo?				
Siempre	Habitualmente	La mayor parte de veces	A veces	Nunca
38,86 %	35,07 %	24,64 %	1,42 %	0 %
¿Con qué frecuencia el sistema fue lento en su respuesta?				
Siempre	Habitualmente	La mayor parte de veces	A veces	Nunca
6,16 %	12,80 %	23,70 %	55,45 %	1,9 %
¿El sistema se comportó del modo esperado durante la conversación?				
Siempre	Habitualmente	La mayor parte de veces	A veces	Nunca
14,69 %	50,24 %	31,75 %	3,32 %	0 %

Tabla 3.1: Estadísticas de las preguntas formuladas a los informantes que adquirieron el corpus DIHANA

en cada una de las tres sedes en las que se realizó la adquisición del corpus, para dichas preguntas.

En un segundo bloque se evaluaba la facilidad de obtener los objetivos marcados para cada uno de los escenarios:

- ¿Fue fácil conseguir el objetivo del escenario (1 al 4)?

Las posibles respuestas eran: *Muy fácil*, *Fácil*, *Así así*, *Difícil*, *No pude*. En el caso de que el usuario seleccionase esta última opción, se le solicitaba un comentario de los motivos que le dificultaron la adquisición.

La Tabla 3.2 muestra las estadísticas obtenidas para los escenarios adquiridos en el corpus. La Sección A.3.2 del Anexo A muestra gráficas explicativas de los resultados obtenidos en cada una de las tres sedes.

Estadísticas de los escenarios					
	Muy fácil	Fácil	Así así	Difícil	No pude
Resultados globales	23,58 %	52,49 %	16,00 %	5,69 %	2,25 %
Escenarios cerrados	22,10 %	52,94 %	16,58 %	5,70 %	2,67 %
Escenarios abiertos	26,50 %	51,59 %	14,74 %	2,11 %	2,11 %

Tabla 3.2: Estadísticas de los escenarios formulados a los informantes que adquirieron el corpus DIHANA

3.3.7. Características del corpus DIHANA

En la adquisición participaron 225 usuarios (153 hombres y 72 mujeres; 75 usuarios por sede), el número total de turnos de usuario que se adquirieron es 6.280 (una media de siete turnos de usuario por diálogo). La Tabla 3.3 muestra las principales características del corpus adquirido.

Número de locutores	225 hablantes (153 hombres y 72 mujeres)
Número de diálogos por hablante	4 diálogos
Número de turnos de usuario	6.280 turnos (48.631 palabras)
Media de turnos/diálogo	7 de usuario
Media de palabras/turno	7,74 palabras/turno de usuario
Vocabulario	823 palabras
Duración de la grabación	10,8 horas

Tabla 3.3: Principales características del corpus DIHANA

La Tabla 3.4 resume las características principales del corpus adquirido para el proyecto BASURDE. De la comparación de las características de ambos corpus, puede observarse la ampliación realizada en cuanto a número de diálogos adquiridos, número de usuarios que participaron en el proceso de adquisición y talla del vocabulario.

Número de usuarios	75
Número de diálogos	227
Número de diálogos por hablante	3 diálogos
Número de turnos de usuario	1.654 turnos (14.902 palabras)
Número de turnos de sistema	1.874 turnos
Vocabulario	637 palabras

Tabla 3.4: Principales características del corpus BASURDE

3.4. Etiquetado del corpus DIHANA

Previamente al aprendizaje de modelos estadísticos a partir de un corpus, es necesario realizar su correcto etiquetado. Para llevar a cabo el etiquetado del corpus DIHANA, la representación de los turnos de usuario y sistema se realizó en términos de actos de diálogo (Searle, 1969).

3.4.1. Etiquetado de los turnos de usuario: representación semántica de la tarea

Para el caso de los turnos de usuario, los actos de diálogo se corresponden con la interpretación semántica de la intervención del usuario en base a frames (conceptos y atributos).

Para la tarea se definieron un total de once conceptos. Siete de estos conceptos son dependientes de la tarea, correspondiéndose con los tipos de consulta que puede realizar el usuario al sistema (Tabla 3.5).

<i>(Hora)</i>
<i>(Hora-Salida)</i>
<i>(Hora-Llegada)</i>
<i>(Precio)</i>
<i>(Tipo-Tren)</i>
<i>(Duración)</i>
<i>(Servicios)</i>

Tabla 3.5: Conceptos dependientes de la tarea

Los cuatro conceptos restantes son independientes de la tarea, incluyéndose el concepto incompleto (utilizado en los casos en los que no se detecta ningún concepto en la intervención del usuario), Afirmación, Negación y No-entendido (utilizado para representar los casos en los que el usuario solicita la repetición de la información suministrada por el sistema o indica que no la ha entendido). Los conceptos no dependientes de la tarea vienen reflejados en la Tabla 3.6.

<i>()</i>
<i>(Afirmación)</i>
<i>(Negación)</i>
<i>(No-Entendido)</i>

Tabla 3.6: Conceptos no dependientes de la tarea

Además, se definieron un total de 18 atributos, o conjunto de datos que el usuario puede aportar en el contexto de la tarea como modificadores de la consulta que desea realizar. Los atributos definidos se muestran en la Tabla 3.7.

En cuanto a la normativa general definida para realizar el etiquetado de las intervenciones del usuario, merecen destacarse los siguientes puntos:

- La representación semántica definida es no secuencial, es decir, en el proceso de representación se sigue el orden de conceptos y atributos mostrado en las Tablas 3.5, 3.6 y 3.7.), independientemente del orden de aparición de la información en la frase original del usuario.
- Se decidió realizar una traducción lo más literal posible de la intervención del usuario, con la idea de no expresar conceptos o valores que no se hubieran mencionado explícitamente.
- El valor de los atributos puede aparecer negado, utilizándose la fórmula *NO(valor)*.
- Un mismo atributo puede tener más de un valor asociado.

<i>Dirección</i>
<i>Ciudad</i>
<i>Ciudad-Origen</i>
<i>Ciudad-Destino</i>
<i>Hora</i>
<i>Hora-Salida</i>
<i>Hora-Llegada</i>
<i>Fecha</i>
<i>Fecha-Salida</i>
<i>Fecha-Llegada</i>
<i>Tipo-tren</i>
<i>Tipo-Viaje</i>
<i>Clase-Billete</i>
<i>Número-Relativo-Orden</i>
<i>Número-Relativo-Orden-Salida</i>
<i>Número-Relativo-Orden-Llegada</i>
<i>Precio</i>
<i>Servicios</i>

Tabla 3.7: Atributos definidos para la representación semántica en la tarea DIHANA

Turno del Usuario:

No a Palencia, a Valencia

Interpretación semántica:

Tipo-Viaje: Nil

()

Ciudad-Destino: NO(Palencia) Valencia

Con respecto a la normativa referente a conceptos y atributos específicos, merece mencionarse:

- Los conceptos independientes de la tarea ((*Afirmación*), (*Negación*), (*No-entendido*)) reciben un tratamiento especial, ocupando la primera posición de la representación y no disponiendo de atributos asociados.

Turno del Usuario:

Sí, me gustaría saber el precio

Interpretación semántica:

(Afirmación)

Tipo-Viaje: Nil

(Precio)

- La presencia del atributo *Tipo-Viaje* es obligatoria siempre y cuando en la representación semántica del turno de usuario aparezcan uno o más conceptos dependientes de la tarea. El atributo toma el valor *Nil* en caso de no mencionarse explícitamente que se trata de un viaje de ida, vuelta o ida-vuelta.

- Los conceptos (*Hora*), (*Hora-Salida*) y (*Hora-Llegada*) se utilizan para representar una consulta de horarios por parte del usuario. En los casos en los que el usuario no manifieste explícitamente que desea consultar horarios de salida (concepto (*Hora-Salida*)) o de llegada ((*Hora-Llegada*)), se utiliza la representación genérica (*Hora*).

Turno del Usuario:

Quería consultar horarios de Palencia a Oviedo

Interpretación semántica:

Tipo-Viaje: Nil

(Hora)

Ciudad-Origen: Palencia

Ciudad-Destino: Oviedo

Turno del Usuario:

Quiero saber a qué hora llega y cuánto cuesta

Interpretación semántica:

Tipo-Viaje: Nil

(Hora-Llegada)

(Precio)

- El atributo *Dirección* se incorpora a la representación semántica si en el turno de usuario aparecen las palabras salida y/o llegada. Si la salida se puede asociar de forma natural a los atributos definidos específicamente para la salida o llegada, no se indica la dirección.

Turno del Usuario:

Horarios para salir en AVE

Interpretación semántica:

Tipo-Viaje: Nil

(Horario)

Dirección: salida

Tipo-Tren: AVE

Turno del Usuario:

Precios para salir el 17 de diciembre

Interpretación semántica:

Tipo-Viaje: Nil

(Precio)

Fecha-Salida: [?-17-12-????,?-17-12-????]

- La casuística y formatos definidos para la traducción de intervalos horarios (atributos *Hora*, *Hora-Salida*, *Hora-Llegada*) e intervalos de fechas (atributos *Fecha*, *Fecha-Salida*, *Fecha-Llegada*) viene descrita en (Griol et al., 2006). La principal diferencia en cuanto a la representación definida para el proyecto BASURDE consiste en la realización de una traducción lo más literal posible, sin la utilización de calendarios o referencias para llevar a cabo la misma.

Turno del Usuario:

Me interesaría uno que saliera a media mañana entre las nueve y las doce.

Interpretación semántica:

Tipo-Viaje: Nil

()

Hora-Salida: (media)mañana [09.00.AM-12.00.AM]

Turno del Usuario:

Preferiría que fuese en navidad, del lunes dieciocho al viernes veintiséis de diciembre.

Interpretación semántica:

Tipo-Viaje: Nil

()

Fecha: [navidad] [lunes-18-12-????-viernes-26-12-????]

3.4.2. Etiquetado de los turnos de sistema

El conjunto de etiquetas definidas para realizar el etiquetado de los turnos de sistema en DIHANA, sigue la misma idea que el conjunto diseñado para el proyecto BASURDE (Martínez et al., 2002). De nuevo, el etiquetado de actos de diálogo se define a tres niveles, inspirándose en la idea del *Interchange Format* (Fukada et al., 1998). Cada etiqueta abstrae la información fundamental para el diálogo de lo que se conoce como *segmento*: unidad con significado propio dentro del contexto del diálogo. Cada turno de diálogo puede estar compuesto de uno o varios segmentos.

El primer nivel del etiquetado se corresponde con la intencionalidad del segmento, también conocido como *acto de habla*. Contiene etiquetas referentes a un diálogo general independientemente de la tarea: (*Apertura, Cierre, Indefinida, No-Entendido, Espera, Nueva-Consulta, Afirmación, Negación, Pregunta, Confirmación y Respuesta*). Las normas de etiquetado definidas para la utilización de las etiquetas del primer nivel son:

- **Apertura:** Se aplica al primer turno del sistema.

Bienvenido al servicio de información de trenes. ¿En qué puedo ayudarle?

(M:Apertura:Nil:Nil)

- **Cierre:** Se aplica al turno tras el cual el sistema finaliza el diálogo.

Gracias por utilizar este servicio. Feliz viaje.

(M:Cierre:Nil:Nil)

- **Indefinida:** Se aplica a segmentos/turnos con frases fuera de la tarea o sin ningún sentido. No son habituales.

Este sistema no está preparado para realizar reservas.

(M:Indefinida:Nil:Nil)

- **No-entendido:** Para el segmento que expresa que no se ha entendido la información proporcionada por el usuario (típicamente, “¿puede repetir?” o frase sinónimas).

Lo siento, no le he entendido. Por favor, hable claro y despacio.

(M:No-entendido:Nil:Nil)

- **Espera:** Para el segmento que indica que el sistema va a acceder a la base de datos en busca de la información solicitada (normalmente, “Un momento, por favor.”).

Un momento, por favor.

(M:Espera:Nil:Nil)

- **Nueva-consulta:** Se aplica al segmento en el que se da al usuario la posibilidad de continuar el diálogo (generalmente, “¿Desea algo más?”).

¿Desea algo más?

((M:Nueva_consulta:Nil:Nil)

- **Pregunta:** Cualquier petición al usuario que no incluya datos específicos referidos al dato preguntado (en ese caso sería *Confirmación:*).

¿A qué ciudad desea viajar?

(M:Pregunta:Ciudad:Nil)

- **Confirmación:** Cualquier petición de información al usuario que incluya datos específicos (los datos a confirmar) o que busque asegurar el tipo de consulta que debe proporcionar el sistema (confirmación del concepto u objetivo del diálogo).

¿Quiere saber horarios?

(M:Confirmación:Hora_salida:Nil)

Quiere horarios de trenes a Granada, ¿desde Valencia?

(M:Confirmación:Hora_salida:Destino)(M:Confirmación:Origen:Origen)

- **Respuesta:** Cualquier aporte de datos al usuario; incluye la respuesta vacía (es decir, cuando se informa al usuario de que no hay trenes que cumplan las condiciones pedidas).

El único tren es un talgo que sale a las 0 y 27 de la noche. ¿Desea algo más?

(M:Respuesta:Hora_salida:Hora_salida,Número_trenes,Tipo_tren)(M:Nueva_consulta:Nil:Nil)

El segundo nivel se refiere al repositorio de información implícito al segmento, es decir, el conjunto de datos que se emplea (en el caso de generar una información con datos concretos) o modifica (en el caso de que se den los datos que restringen la información buscada). Este segundo nivel está íntimamente relacionado con el concepto de *frame* como repositorio de datos utilizado en el módulo de comprensión semántica. Las etiquetas definidas para el segundo nivel son: *Hora-salida*, *Hora-Llegada*, *Precio*, *Tipo-tren*, *Origen*, *Destino*, *Fecha*, *Servicios*, *Clase*, *Tiempo-Recorrido* y *Nil*. Las normas establecidas para el segundo nivel son las siguientes:

- Presenta el valor *Nil* (ausencia de valor) para las etiquetas de primer nivel *Apertura*, *Cierre*, *Indefinida*, *No-Entendido*, *Espera* o *Nueva-Consulta*.
- Presenta el valor asociado al dato que se busca satisfacer en *Pregunta*.
- Presenta el valor asociado al dato del que se busca información en *Respuesta* (lo cual viene determinado en turnos previos); esto incluye la respuesta vacía. La respuesta del tipo “Un único tipo de tren” presenta el valor *Tipo-tren*.
- Para *Confirmación*, si no es sobre un dato específico (confirmación implícita), presenta el valor del concepto con el que se está trabajando (igual que para *Respuesta*). Típicamente, para las confirmaciones del tipo “Le consulto horarios para el día DÍA desde ORIGEN a DESTINO”.
- Para *Confirmación* sobre un dato específico (confirmación explícita), presenta el valor asociado al dato que se está confirmando (igual que para *Pregunta*).

El tercer nivel de etiquetado indica los datos concretos utilizados en el segmento. Este nivel está relacionado con los datos concretos (*cases*, *atributos* o *slots*) utilizados dentro de los repositorios de información. Como valores posibles del etiquetado, no se indican qué datos concretos son, sino el grupo genérico al que pertenecen (por ejemplo, CIUDAD pero no VALENCIA). Las etiquetas definidas para el tercer nivel son: *Hora-salida*, *Hora-Llegada*, *Precio*, *Tipo-tren*, *Origen*, *Destino*, *Fecha*, *Número-orden*, *Número-Trenes*, *Servicios*, *Clase*, *Tipo-Viaje*, *Tiempo-Recorrido* y *Nil*.

En cuanto a la normativa definida para el tercer nivel, presenta el valor *Nil* para las etiquetas mostradas para el mismo caso en el segundo nivel. Para el resto de los casos, las etiquetas se corresponden con los datos dados en el segmento. Para la respuesta vacía, el valor del tercer nivel se corresponde a *Número-Trenes*. En (Alcácer et al., 2004a) puede encontrarse más información sobre el etiquetado del diálogo definido en DIHANA.

La Figura 3.7 muestra un diálogo del corpus DIHANA, con su etiquetado correspondiente.

M1: Bienvenido al servicio de información de trenes, ¿ En qué puedo ayudarle?
 (M:Apertura:Nil:Nil)
 U1: Hola, me gustaría saber horarios para Madrid el jueves veintidós de mayo.
TIPO-VIAJE:nil
(HORA)
CIUDAD-DESTINO:Madrid
FECHA:[jueves-22-05-????,jueves-22-05-????]
 M2: Le consulto horarios de trenes con salida en Valencia destino Madrid saliendo el sábado, día 22 de mayo de 2004. Un momento por favor.
 Hay varios trenes. El primero sale a las 7 y 45 de la mañana y llega a las 11 y 14 de la mañana y el último sale a las 6 y 45 de la tarde y llega a las 10 y 18 de la noche.
 ¿Desea algo más?
 (M:Confirmación:Hora_salida:Destino,Día,Origen)
 (M:Espera:Nil:Nil)
 (M:Respuesta:Hora_salida:Hora_llegada,Hora_salida,Número_relativo_orden,Número_trenes)
 (M:Nueva_consulta:Nil:Nil)
 U2: Sí, me gustaría saber trenes de vuelta y su precio también.
(AFIRMACIÓN)
TIPO-VIAJE:vuelta
(PRECIO)
 M3: ¿ Quiere saber horarios ?
 (M:Confirmación:Hora_salida:Nil)
 U3: Sí.
(AFIRMACIÓN)
 M4: Le consulto horarios de trenes con salida en Madrid destino Valencia saliendo el sábado, día 22 de mayo de 2004. Un momento por favor.
 (M:Confirmación:Hora_salida:Destino,Día,Origen)
 (M:Espera:Nil:Nil)
 Hay varios trenes. El primero sale a las 8 de la mañana y llega a las 11 y 27 de la mañana y el último sale a las 7 de la tarde y llega a las 10 y 29 de la noche. ¿Desea algo más?
 (M:Respuesta:Hora_salida:Hora_llegada,Hora_salida,Número_relativo_orden,Número_trenes)
 (M:Nueva_consulta:Nil:Nil)
 U4: Sí, saber los precios de los trenes tanto de ida como de vuelta.
(AFIRMACIÓN)
TIPO-VIAJE:ida_y_vuelta
(PRECIO)
 M5: Le consulto precios. Un momento por favor.
 (M:Confirmación:Precio:Nil)
 (M:Espera:Nil:Nil)
 Hay varios trenes . En clase turista, el tren más caro es un alaris que cuesta 36 con 50 euros la ida y el más barato es un regional que cuesta 18 con 80 euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento.
 ¿Desea algo más?
 (M:Respuesta:Precio:Clase_billete,Número_trenes,Precio,Tipo_tren)
 (M:Nueva_consulta:Nil:Nil) pal: 55-59
 U5: No.
(NEGACIÓN)
 M6: Gracias por utilizar este servicio. Feliz viaje.
 (M:Cierre:Nil:Nil)

Figura 3.7: Ejemplo del etiquetado definido para el corpus DIHANA

Capítulo 4

Sistema de diálogo DIHANA

4.1. Introducción

Tal y como se ha comentado en el capítulo anterior, uno de los objetivos principales del proyecto DIHANA fue el desarrollo de un sistema de diálogo hablado en el contexto del dominio del proyecto. El desarrollo de este sistema ha sido necesario para poder evaluar las metodologías de gestión de diálogo que se presentan en esta tesis.

Como puede observarse en el capítulo segundo de la tesis, los sistemas de diálogo mediante la voz de dominio restringido se basan generalmente en tres aspectos fundamentales: la vía de comunicación es la línea telefónica, la tarea está semánticamente restringida al dominio del sistema y el control del sistema se realiza a través de una iniciativa mixta, es decir, el diálogo no está totalmente dirigido por el sistema, sino que el usuario puede tomar la iniciativa y orientar el diálogo en base a sus preguntas. Además, para garantizar la utilidad del sistema suele imponerse el lenguaje natural, con un léxico lo más amplio posible y sin una sintaxis rígida impuesta por el sistema, así como la aceptación del lenguaje natural.

El esquema utilizado para el desarrollo de estos sistemas, como se ha expuesto previamente, suele englobar una serie de módulos genéricos para tratar las múltiples fuentes de conocimiento, que deben coordinarse para responder a los requerimientos del usuario. Para el desarrollo del sistema DIHANA se ha definido una arquitectura en estrella similar a GALAXY. De este modo, existe un hub central que actúa de centralizador del conjunto de mensajes generados por el resto de módulos que componen el sistema. Para la comunicación entre los diferentes módulos del sistema se ha desarrollado un formato de comunicación basado en el envío de paquetes en XML.

En las publicaciones (Hurtado et al., 2005) (Griol et al., 2006a) (Griol et al., 2006b) (Sanchis et al., 2006) (Griol et al., 2007) se describen las características más relevantes de cada uno de los módulos que conforman el sistema de diálogo desarrollado, la arquitectura y formato de comunicación definidos, las diferentes etapas llevadas a cabo para su realización y los procedimientos definidos para realizar su evaluación.

4.2. Arquitectura del sistema y protocolo de comunicación

Como se ha reseñado en el capítulo de introducción de esta tesis, un sistema de diálogo debe llevar a cabo un número de tareas específicas, partiendo cada una de las mismas de una determinada fuente de información. El hecho de plantear una arquitectura modular facilita la consecución de estas tareas. Asimismo, permite la substitución de los diferentes módulos del sistema de forma sencilla y su disposición en diferentes máquinas, de forma que para su inclusión en el sistema DIHANA simplemente haga falta especificar la dirección IP y el puerto de la computadora donde se encuentra instalado el módulo correspondiente.

Para la implementación del sistema de diálogo en DIHANA se ha optado por una arquitectura siguiendo el paradigma cliente-servidor, donde el gestor central actúa como cliente, gestionando las comunicaciones, y el resto de módulos actúan como servidores. La Figura 4.1 muestra la arquitectura del sistema de diálogo desarrollado para DIHANA.

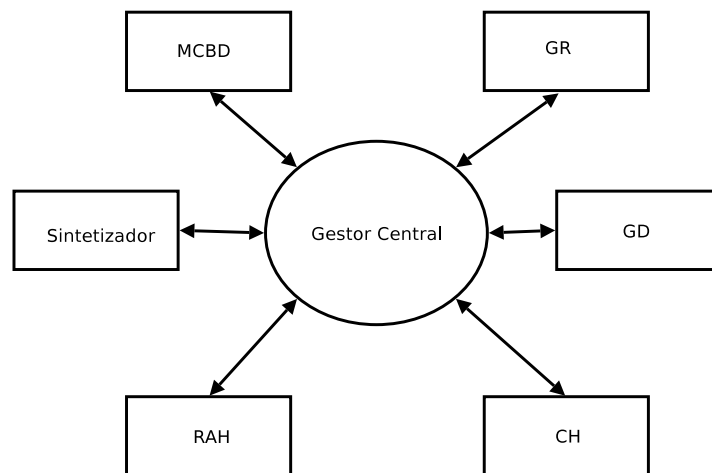


Figura 4.1: Arquitectura definida para el sistema de diálogo DIHANA

A continuación se mencionan brevemente algunas características básicas de los módulos relevantes del sistema, que se detallan de una forma más amplia a lo largo de este capítulo:

- Módulo de Reconocimiento Automático del Habla (RAH): Facilita la secuencia de palabras que se ajusta a la señal vocal de entrada con una mayor probabilidad. Los investigadores de la Universidad de Zaragoza han desarrollado un reconocedor automático del habla expresamente para el proyecto. Asimismo, se ha realizado la integración del reconocedor del habla Sphinx-II y el entrenamiento de los modelos acústicos con SphinxTrain, ambos desarrollados por la Carnegie Mellon University.
- Módulo de comprensión del habla (CH): Extrae el significado de las palabras reconocidas en el módulo anterior, expresándolo en términos de un lenguaje semántico especificado para la tarea. Para la definición de la semántica de la tarea se utiliza el concepto de frame: cada intervención del usuario genera frames que representan el significado de la intervención.

- Gestor de diálogo (GD): Decide qué paso debe dar el sistema tras cada intervención del usuario. Para ello se basa en la interpretación semántica generada por comprensión, el historial del diálogo, el estado actual del sistema, información obtenida de la base de datos, la estrategia definida, etc. Para esta tesis, se han desarrollado dos estrategias de gestión basadas en reglas y una basada en la utilización de modelos estadísticos aprendidos a partir del corpus DIHANA.
- Módulo de consulta a la base de datos de la aplicación (MCBD): Recibe peticiones de consulta a la base de datos por parte del gestor de diálogo, las procesa y devuelve el resultado al gestor. El sistema utiliza una base de datos en PostgreSQL con 11 tablas en las que se agrupan los trayectos que comparten orígenes, destinos, fechas de salida, etc.
- Módulo de generación de respuestas (GR): 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. La estrategia principal que sigue el sistema consiste en la utilización de plantillas de respuestas adaptadas a los tipos de consulta.
- Módulo de síntesis texto-voz (STV): Transforma la frase de respuesta al usuario en señal de audio. Se ha realizado la integración de un sintetizador en castellano desarrollado expresamente para el proyecto. Además, se ha integrado el sistema de síntesis del habla Festival.
- Módulo central gestor de comunicaciones (Communication manager): Este módulo es el responsable de establecer las comunicaciones con el resto de módulos del sistema, encargándose de recibir la totalidad de mensajes enviados y de encaminarlos al servidor destino.

La comunicación entre los diversos módulos se lleva a cabo mediante el envío de paquetes en formato XML a través de los sockets dedicados. El formato definido para los paquetes XML contiene una cabecera en la que se indica el módulo generador del paquete y el módulo al que va dirigido. Seguidamente se definen etiquetas específicas para cada módulo para realizar el encapsulado de la información que se remite.

La Figura 4.2 muestra dos ejemplos de paquetes XML definidos para operar en el sistema siguiendo el formato de comunicación desarrollado. El primero de ellos va dirigido del módulo de comprensión (CH) al gestor de diálogo (GD). El segundo paquete ha sido generado por el gestor de diálogo y tiene como destinatario el módulo de consulta a la base de datos (MCBD). En la Sección B.1 del Anexo B se muestran los paquetes fundamentales que conforman la comunicación realizada por los diversos módulos del sistema. Una descripción completa de la librería de comunicaciones definida para el sistema puede encontrarse en (Lleida, 2005) (Hurtado, Blat, y Griol, 2006).

4.3. Reconocedor automático del habla

Se conoce por Sphinx un conjunto de herramientas desarrolladas por el Departamento de Informática de la Carnegie Mellon University. Entre estas herramientas se encuentra

```

<?xml version="1.0" encoding="ISO-8859-15"?>
<paquete destino="GD" origen="CH" version="1.0">
  <comprension>
    <segmentacion_semántica>
      <segmento etiqueta="consulta" valor="quiero">
        <confianza valor="0.0219018941131"/>
        <confianza valor="0.126746506986"/>
      </segmento>
      <segmento etiqueta="hora_s" valor="horarios">
        <confianza valor="0.14695464419"/>
        <confianza valor="0.0763993948563"/>
      </segmento>
    </segmentacion_semántica>
    <frames>
      <frame tipo="(HORA-SALIDA)">
        <confianza_frame valor="0.14695464419"/>
        <confianza_frame valor="0.0763993948563"/>
      </frame>
    </frames>
  </comprension>
</paquete>

```

```

<?xml version="1.0" encoding="ISO-8859-15"?>
<paquete destino="BD" origen="GD" version="1.0">
  <consulta_bd>
    <frames>
      <frame tipo="(HORA-SALIDA)">
        <atributo tipo="CIUDAD-ORIGEN">
          <valor_atributo valor="valencia"/>
        </atributo>
        <atributo tipo="CIUDAD-DESTINO">
          <valor_atributo valor="bilbao"/>
        </atributo>
        <atributo tipo="INTERVALO-FECHA-SALIDA">
          <valor_atributo valor="24-11-2004/24-11-2004"/>
        </atributo>
      </frame>
    </frames>
  </consulta_bd>
</paquete>

```

Figura 4.2: Ejemplo de paquetes XML definidos para la comunicación en el sistema de diálogo DIHANA

el SphinxTrain (www.speech.cs.cmu.edu/SphinxTrain), aplicación desarrollada para realizar el entrenamiento de los modelos acústicos que pueden utilizarse con los diferentes reconocedores desarrollados: Sphinx-2, Sphinx-3 y Sphinx-4.

Sphinx-2 (o Sphinx-II) se recomienda para aplicaciones de mano, portables y para sistemas de escritorio o con acceso telefónico que requieran tiempos de respuesta cortos. Sphinx-3 es un reconocedor más lento, pero más preciso. Un posible entorno de aplicación es la transcripción de noticias. Sphinx-4 es un reconocedor desarrollado por completo en lenguaje Java fruto de la colaboración de varias universidades (CMU, MIT y UCSC) y empresas (Sun, Mitsubishi Electric y Hewlett Packard). Sphinx-4 es capaz de reconocer habla continua y discreta. Incluye implementaciones de características como el pre-énfasis, ventanas Hamming, transformada rápida de Fourier (FFT), bancos de filtro de frecuencia Mel, transformada discreta de cosenos (DCT), diversas normalizaciones y extracción de características por cepstrales. Soporta modelos de lenguaje ASCII y binarios de unigramas,

bigramas, trigramas, formato JSGF (Java Speech API Grammar Format) y gramáticas formato ARPA FST. Por último, proporciona diversas utilidades para el post-procesado de los resultados del reconocedor (medidas de confianza, transformación de gramáticas en formato ECMAScript a formato JSGF) y para la visualización de formas de onda y espectrogramas.

Para el proyecto DIHANA se ha utilizado Sphinx-2. Se trata de un reconocedor del habla en tiempo real que soporta modelos acústicos semicontinuos y continuos. Sus características principales son las siguientes:

- Reconocimiento del habla continua.
- Independiente del usuario.
- Proporciona el mejor resultado o los n mejores.
- Modelos acústicos semicontinuos y continuos.
- Modelos de lenguaje de bigramas, trigramas o de estados finitos.

Además de un reconocedor de voz, Sphinx2 proporciona un conjunto de librerías desde el conjunto de funciones para el reconocimiento del habla como librerías para tratar el audio a bajo nivel. Estas librerías, escritas en C, están disponibles tanto para el sistema operativo Unix/Linux como para Windows. Además, hay disponible un conjunto de ficheros de demostraciones, destacando entre ellas el reconocedor del habla online, el offline y el servidor de reconocimiento del habla.

En (Griol et al., 2006a) se describe el trabajo desarrollado por Sergio Grau para la implementación de un reconocedor automático del habla para DIHANA utilizando Sphinx-2. En DIHANA, se entrenaron modelos acústicos semicontinuos para el castellano utilizando 25 fonemas más silencio. Para realizar este aprendizaje se utilizó el corpus DIHANA. Como modelo de lenguaje se utilizó un modelo de trigramas con descuento Witten-Bell (Witten y Bell, 1991). Este modelo se construyó utilizando el corpus DIHANA y el toolkit CMU-Cambridge Language modeling (Clarkson y Rosenfeld, 1997). En este modelo, se aplicó al corpus un proceso de categorización (ver Anexo B.2) definiendo categorías como CIUDAD, TIPO-TREN, DÍA, MES, etc. Estas categorías redujeron la talla del vocabulario a 649 palabras distintas. Utilizando Sphinx-2 como decodificador, el WER obtenido fue 14.07%.

4.4. Módulo de comprensión

Los módulos de comprensión desarrollados para DIHANA (definidos y construidos en el proyecto en BASURDE) se basan en modelos estocásticos estimados mediante técnicas de aprendizaje automático (Segarra et al., 2002) (Llisterri y Machuca, 2006) (García, 2003) (Hurtado, 2004).

Al igual que en muchos otros sistemas de diálogo, la representación semántica definida para la tarea se basa en el concepto de frame. Por tanto, el modulo de comprensión recibe como entrada la frase generada en el proceso de reconocimiento y suministra como salida uno o más frames. El proceso de comprensión se realiza en dos fases:

1. La primera fase traduce la frase de entrada en una secuencia semántica definida en un lenguaje intermedio (ISL), utilizando técnicas de aprendizaje automático para estimar un modelo de comprensión estocástico.
2. La segunda fase traduce esta secuencia semántica en su correspondiente frame. Este proceso se basa en la utilización de reglas que garantizan la secuencialidad en la traducción a frames canónicos.

La Figura 4.3 muestra las dos fases definidas para realizar la comprensión semántica.

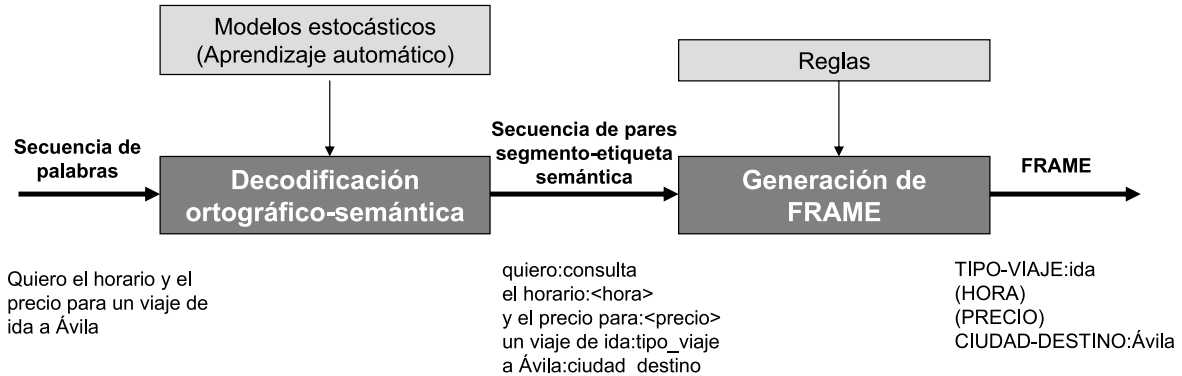


Figura 4.3: Esquema del proceso de comprensión desarrollado para el proyecto BASURDE

La segmentación de la frase de entrada se realiza en un número de intervalos igual al número de unidades semánticas en la traducción semántica correspondiente. Llamando W al conjunto de palabras del vocabulario y V al alfabeto de unidades semánticas, los datos de entrenamiento son un conjunto de parejas (u, v) que cumplen:

$$\begin{aligned}
 u &= u_1, u_2 \dots u_n & u_i &= w_{i_1} w_{i_2} \dots w_{i_{|u_i|}} \\
 w_{i_j} &\in W, i = 1, \dots, n, j = 1, \dots, |u_i| \\
 v &= v_1 v_2 \dots v_n, v_i \in V, i = 1, \dots, n
 \end{aligned}$$

De este modo, cada frase de W dispone de un par (u, v) asociado, donde v es la secuencia de unidades semánticas y u la secuencia de segmentos en los que se ha dividido la frase. La Figura 4.4 muestra un ejemplo de traducción.

u ₁ : me podría decir	v ₁ : consulta
u ₂ : los horarios de trenes	v ₂ : <hora_salida>
u ₃ : para	v ₃ : marcador_destino
u ₄ : Barcelona	v ₄ : ciudad_destino
Par de entrada (u,v) = (u ₁ u ₂ u ₃ u ₄ ,v ₁ v ₂ v ₃ v ₄)	
Salida v = consulta <hora_salida> marcador_destino ciudad_destino	

Figura 4.4: Ejemplo de traducción a lenguaje semántico (1ª fase de comprensión)

Teniendo disponible un corpus de entrenamiento, el aprendizaje del traductor secuencial se lleva a cabo a través del aprendizaje de dos modelos: un modelo para el lenguaje semántico y un conjunto de modelos, uno por cada unidad semántica v_i .

Una vez se ha segmentado una frase de entrada y se ha asociado a cada segmento u una etiqueta v , la segunda fase del módulo de comprensión consiste en transformar estos pares en uno o varios frames. Para ello, se aplican una serie de reglas que reorganizan los contenidos, eliminan la información no relevante o mal reconocida, los valores cuantitativos se convierten en cualitativos, etc.

4.4.1. Primera fase del módulo de comprensión

Tal y como se ha comentado previamente, la primera fase del módulo de comprensión realiza la traducción automática de la frase de entrada en base a un lenguaje semántico intermedio. Para ello, se utilizan modelos estocásticos. El lenguaje intermedio utiliza 35 unidades semánticas, organizadas en las siguientes categorías:

1. **Conceptos:** Consultas sobre contenidos de la tarea ($\langle hora \rangle$, $\langle fecha \rangle$, $\langle precio \rangle$, $\langle duración \rangle$, $\langle tipo_tren \rangle$ y $\langle servicios \rangle$).
2. **Atributos:** Restricciones a aplicar a la consulta realizada ($fecha$, $fecha_s$, $fecha_a$, $hora$, $hora_s$, $hora_a$, $ciudad$, $ciudad_origen$, $ciudad_destino$, $tipo_tren$, $tiempo_recorrido$, $precio$, $tipo_viaje$, $clase_billete$, $numero_relativo_orden$ y $servicios$).
3. **Símbolos con meta-información sobre el turno de usuario:** Informan del tipo de intervención del usuario ($\langle afirmación \rangle$, $\langle negación \rangle$ y $\langle no_entendido \rangle$).
4. **Marcadores:** Indican la aparición de un concepto o atributo en el posterior turno de usuario o modifican directamente su valor (m_origen , $m_destino$, m_salida , $m_llegada$ y not).
5. **Otros símbolos:** Se trata de símbolos sin información semántica relevante para la tarea ($nada$, $cortesía$ y $consulta$).

La sección B.2 del Anexo B recoge el vocabulario contemplado para las categorías semánticas definidas en el módulo de comprensión desarrollado en DIHANA.

Para realizar el aprendizaje de la transducción secuencial se han definido dos modelos para los pares de entrenamiento (u, v) :

1. Modelo para el lenguaje semántico L_s : Representa las secuencias de unidades semánticas permitidas. El modelo regular A_s para el lenguaje L_s se estima a partir de las unidades semánticas v de las muestras de entrenamiento.
2. Modelo regular A_{v_i} para cada una de las categorías semánticas v_i , obtenido a partir del conjunto de segmentos de las muestras de entrenamiento asociado a la categoría v_i .

En (Segarra et al., 2002) y (Hurtado et al., 2004) se detallan diferentes posibilidades para la estimación de estos modelos. En dos de ellas (Segarra et al., 2002), el modelo semántico se estima como un modelo clásico de bigramas. En la primera aproximación, los modelos de las unidades semánticas se estimaron en forma de bigramas; en la segunda, se estimaron como un autómata estocástico de estados finitos, que se aprende automáticamente a partir de un algoritmo de Inferencia Gramatical. En (Hurtado et al., 2004) se detalla la aplicación de modelos de multigramas para la comprensión del lenguaje.

4.4.2. Segunda fase del módulo de comprensión

La segunda etapa del módulo de comprensión consiste en convertir la traducción semántica obtenida en la etapa anterior en forma de frames (conceptos y atributos). Para ello, se reorganiza la información, se reconvierten valores relativos y cualitativos en cuantitativos, etc. Todo el proceso se fundamenta en la aplicación de reglas. El conjunto de conceptos y atributos definido para realizar la adquisición del corpus DIHANA aparecen reflejados en las Tablas 3.5, 3.6 y 3.7.

Para realizar esta fase de la traducción, se han utilizado diccionarios que contienen la conversión a valores canónicos (nombres de ciudades, fechas, intervalos horarios...) y se han definido formatos para representar los posibles valores definidos para los diferentes atributos. Sin duda alguna, el caso que presenta una mayor complejidad en esta fase de comprensión se corresponde con la traducción de intervalos de fechas, dado el gran número de casos posibles con respecto al resto de atributos. La principal diferencia con respecto a la traducción de fechas diseñada para el proyecto BASURDE se corresponde con el hecho de no utilizar ningún tipo de calendario para proceder a la traducción, de forma que no exista una fecha de referencia como punto de partida de la traducción. Asimismo, se decidió realizar traducciones literales de las intervenciones del usuario siempre que su representación semántica pudiese dar lugar a malas interpretaciones (ej. incorporar a la traducción nombres de las festividades, estaciones del año, intervalos horarios, etc). En (Griol et al., 2006) puede encontrarse información referente a la segunda fase de la traducción realizada por el módulo de comprensión, en cuanto a reglas y formatos de traducción definidos para cada atributo.

La Figura 4.5 muestra el resultado de aplicar la segunda fase del módulo de comprensión al ejemplo mostrado en la Figura 4.4.

(HORA-SALIDA) CIUDAD-DESTINO:Barcelona

Figura 4.5: Ejemplo de traducción a frames (2ª fase de comprensión)

4.5. Gestión de diálogo

En el ámbito del proyecto DIHANA se han desarrollado dos estrategias para el diseño de gestores de diálogo, que constituyen el núcleo central de la tesis.

La primera aproximación se basa en la utilización de metodologías basadas en reglas. Utilizando estas metodologías se han desarrollado dos gestores de diálogo, que se resumen en el capítulo quinto de la tesis.

La segunda aproximación se basa en un modelo estadístico aprendido a partir del corpus DIHANA. Esta aproximación se detalla en el capítulo sexto de la tesis.

El capítulo séptimo muestra los resultados obtenidos en la evaluación de los diferentes gestores de diálogo desarrollados.

4.6. Generación de respuestas

El generador de respuestas realiza la traducción de las representaciones semánticas de los turnos de sistema a frases en lenguaje natural en castellano. La selección adecuada de las respuestas que generará el sistema es fundamental para evitar errores de entendimiento por parte del usuario, así como para contemplar de forma correcta las diferentes situaciones del diálogo definidas en la estrategia de gestión.

La entrada del generador de respuestas está compuesta por conceptos y atributos, con la posibilidad de incluir medidas de confianza asociadas, lo que permite la generación de respuestas en lenguaje natural detalladas, en las que los atributos se mencionan o no dependiendo de su confianza asociada. Las respuestas del sistema incluyen información procedente de la salida generada por el gestor de diálogo, así como del resultado de las consultas realizadas a la base de datos.

La técnica utilizada usualmente para el desarrollo de generadores de respuestas consiste en disponer de un fichero de plantillas de respuestas asociadas a cada uno de los diferentes tipos de conceptos, en las que aparecen reflejados los nombres de los diferentes atributos. Éstos se sustituyen por los valores de los mismos, obtenidos del registro del diálogo, a la hora de mostrar la respuesta al usuario. Cada uno de los conceptos dispone de su conjunto de plantillas asociado, intentando dar respuesta a las posibles situaciones que pueden darse para cada uno de ellos. De esta forma, el generador de respuestas se convierte en el módulo con funcionamiento más simple del total de componentes del sistema de diálogo.

Por último, merece destacarse que la casuística de respuestas posibles del sistema puede reducirse substancialmente realizando su generación a partir de la concatenación de segmentos cortos, tal y como se presenta en el tercer apartado del capítulo.

4.6.1. Utilización de plantillas de respuestas

La utilización de plantillas de respuestas es la solución más habitual para la construcción de un generador de respuestas. Ésta fue la solución adoptada para realizar la adquisición del corpus DIHANA. Para dicha adquisición se elaboró un fichero de respuestas, agrupadas según los siguientes casos, definidos a partir de la estrategia del Mago de Oz desarrollada para DIHANA (Sección 3.3.4):

- Intervención del usuario fuera del ámbito de la tarea: Se trata de respuestas del tipo *“Este servicio no está preparado para este tipo de consultas. ¿Desea realizar otra consulta?”*, *“Este servicio no está preparado para atender consultas sobre trenes de cercanías o internacionales. ¿Desea realizar otra consulta ?”*, etc.
 - Nivel de entendimiento insuficiente: Conjunto de respuestas asociadas a los casos en el que el sistema no entienda al usuario. (Ej. *“Puede repetir de manera clara lo último que ha dicho, por favor”*, *“Lo siento, no le he entendido. Por favor, hable claro y despacio”*).
 - Completar atributos cuando el estado de la pizarra es seguro: Las respuestas vienen agrupadas para cada uno de los posibles atributos obligatorios definidos en la tarea. (Ej. *“¿A qué ciudad desea viajar?”*, *“¿Qué día desea salir?”*).
-

- Confirmación implícita cuando el estado de la pizarra es seguro: Las respuestas se clasifican según los diferentes tipos de consultas definidos en la tarea, diferenciándose en el número de atributos que se mencionan para los distintos subcasos. (Ej. *“Le consulto horarios de trenes con salida en CIUDAD-ORIGEN destino CIUDAD-DESTINO saliendo FECHA-SALIDA. Un momento por favor.”*, *“Le consulto la duración del trayecto. Un momento por favor.”*).
- Confirmación mixta cuando el estado de la pizarra es seguro: Las respuestas se agrupan por frames, distinguiendo subcasos dependiendo de la fiabilidad asociada al concepto y a los atributos que se incorporan a la confirmación. (Ej. *“Quiere horarios de trenes a CIUDAD-DESTINO, ¿desde CIUDAD-ORIGEN?”*).
- Confirmación explícita cuando el estado de la pizarra es inseguro: Se clasifican según el concepto o atributo que se confirma (Ej. *“¿Me pregunta por horarios?”*, *“¿Quiere viajar desde CIUDAD-ORIGEN?”*).
- Confirmación mixta cuando el estado de la pizarra es inseguro: La casuística es idéntica a la del caso de estado de la pizarra seguro (Ej. *“Saliedo desde CIUDAD-ORIGEN, ¿ha dicho precios?”*).
- Resultado de la consulta a la base de datos: Las respuestas se agrupan según el tipo de consulta, teniéndose en cuenta la cardinalidad de la respuesta (no existen trenes, un único tren/tipo de tren, dos trenes/tipos de tren, más de dos trenes/tipos de tren) y el número de atributos disponibles que se incorporan a la respuesta (Ej. *“No hay ningún tren que cumpla estos requerimientos. Desea cambiar alguna de las características solicitadas?”*, *“Hay varios trenes. El más rápido sale HORA-SALIDA y el más lento HORA-SALIDA. ¿Desea algo más?”*).

El conjunto de plantillas definido para cada uno de los casos anteriores viene reseñado en la Sección C.1 del Anexo C.

El generador de respuestas desarrollado con el objetivo de interactuar con el gestor de diálogo basado en reglas que imita la estrategia del Mago de Oz utiliza los mismos conjuntos de respuestas diseñados para la adquisición. El generador actúa siguiendo los siguientes pasos:

- Realiza la lectura de los frames del sistema (la salida del gestor de diálogo).
 - Consulta el registro del diálogo actual para conocer los valores de los diferentes conceptos y atributos disponibles en él, así como los valores de confianza asignados a cada uno de ellos.
 - A partir del análisis del registro del diálogo y de la información suministrada por el gestor de diálogo se determina en qué estado de los definidos en la estrategia se encuentra el diálogo. De este modo, dado el formato de salida definido para el gestor y el estado actual del diálogo, el generador de respuestas realiza la traducción de dicho formato al conjunto de casos definido para la adquisición.
-

- Seleccionado el estado, se escoge aleatoriamente una de las respuestas definidas del conjunto disponible para el estado. En caso de encontrarse en un estado de consulta a la base de datos, se tiene en cuenta la cardinalidad de la respuesta obtenida para realizar esta elección.
- La frase en lenguaje natural se remite al módulo de síntesis de texto a voz.

Para el caso del gestor de diálogo estadístico, se ha tenido en cuenta el etiquetado a tres niveles realizado para representar las posibles respuestas del sistema. Estas respuestas etiquetadas, tal y como se ha visto en la explicación de este gestor, constituyen la salida generada por el perceptrón multicapa, utilizado en el proceso de clasificación propuesto para seleccionar la respuesta del sistema. De este modo, se ha desarrollado un generador de respuestas que utiliza una agrupación de las plantillas de respuestas en función de su etiquetado. Los pasos seguidos por este generador de respuestas son aún más sencillos que en el caso anterior:

- Realiza la lectura del etiquetado a tres niveles seleccionado a la salida del perceptrón multicapa.
- Utiliza este valor como clave para acceder a un diccionario donde se almacenan las respuestas en función del etiquetado definido para la tarea.
- Se selecciona aleatoriamente una de las respuestas definidas del conjunto disponible para el registro del diccionario correspondiente a la clave. En caso de encontrarse en un estado de consulta a la base de datos, se tiene en cuenta la cardinalidad de la respuesta obtenida para realizar esta elección.
- La frase en lenguaje natural se remite al módulo de síntesis de texto a voz.

De este modo, la estructura del generador de respuestas se limita a un diccionario de respuestas organizadas teniendo en cuenta su distinto etiquetado. El conjunto de plantillas que componen el diccionario definido para la tarea DIHANA viene reseñado en la Sección C.2 del Anexo C.

4.6.2. Utilización de árboles de respuestas

Un paso adelante en la utilización de plantillas de respuestas, consiste en su generación a partir de gramáticas incontextuales, en las que la respuesta final se produce mediante la concatenación de segmentos cortos. Para cubrir el mayor número de casos con un número limitado de frases se siguen las siguientes reglas, definidas teniendo en cuenta la estrategia del diálogo detallada para el segundo modelo de gestión basado en reglas:

1. Todos los diálogos comienzan con la misma frase estándar: *“Bienvenido al sistema de información de trenes. ¿En qué puedo ayudarle?”*.
2. Mientras el sistema no detecte la aparición de un concepto relativo a la tarea, la respuesta es siempre la misma: *“Por favor, indíqueme que tipo de información necesita”*.

3. En la generación de respuestas intervienen el tipo de concepto actual y el número de atributos mencionados. Para un determinado concepto pueden darse los siguientes casos:

- a) El usuario no ha mencionado alguno de los atributos obligatorios para el tipo de concepto actual. En este caso se van solicitando cada uno de dichos atributos siguiendo un orden predeterminado. Esta fase se repite hasta que se disponga de un valor para el total de atributos obligatorios, independientemente del valor de su medida de confianza. Para ello, se utilizan los árboles de petición de atributos. Cada uno de los conceptos relativos a la tarea dispone de su árbol de respuestas para realizar la petición de atributos. La respuesta final se selecciona recorriendo el árbol correspondiente teniendo en cuenta la información disponible en el registro del diálogo. El primer concepto necesario para llevar a cabo la consulta requerida por el usuario no disponible en el registro, se selecciona para realizar su petición. En la Figura 4.6 se muestra un ejemplo de un árbol de petición de atributos definido para las consultas de horarios.

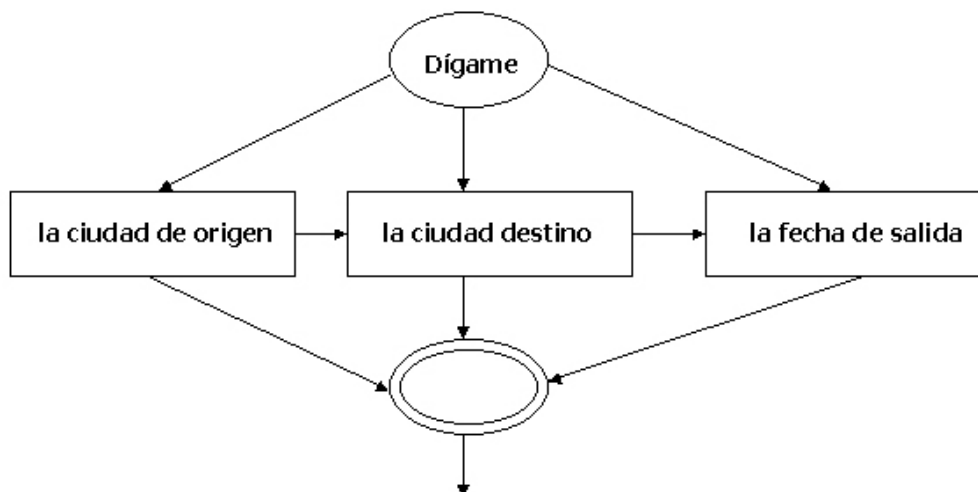


Figura 4.6: Ejemplo de un árbol de respuestas de petición de atributos

- b) En el caso de que se disponga de un valor para todos los atributos obligatorios del concepto actual se genera una respuesta para confirmar aquellos atributos cuya medida de confianza sea inferior a un umbral prefijado, utilizándose el árbol de confirmación de respuestas generado para el concepto correspondiente. Al igual que en el caso de la petición de atributos, cada concepto dependiente de la tarea dispone de su árbol de confirmación de atributos asociado. Para generar la respuesta final, el árbol se recorre teniendo en cuenta las confianzas asociadas a los diferentes atributos y el número de ellos que se desee confirmar en la respuesta. La Figura 4.7 muestra un ejemplo de estos árboles, definido en este caso para llevar a cabo confirmaciones en las consultas de horarios.
- c) En el caso de que la confianza del total de atributos resultado de la traducción semántica supere el umbral anterior, no se realizan confirmaciones y directamente se realiza la consulta a la base de datos, una vez se dispone del valor

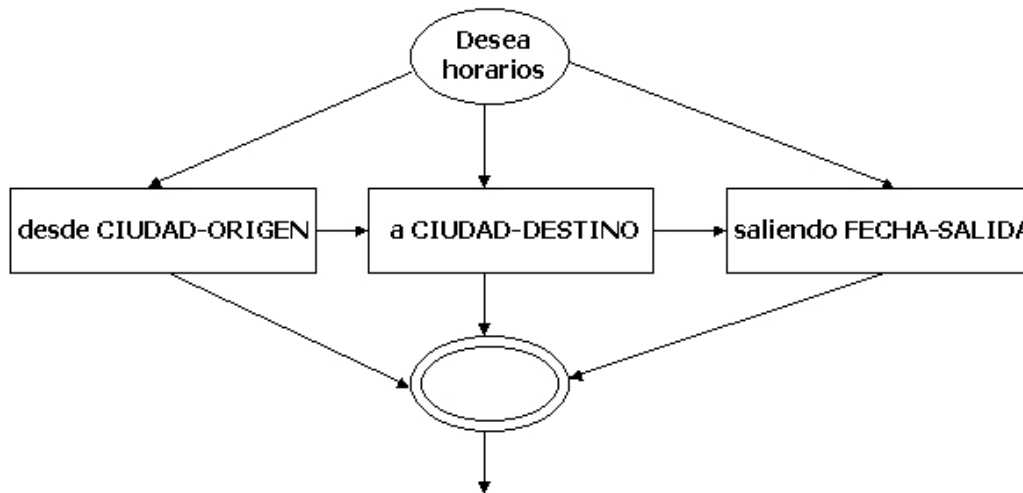


Figura 4.7: Ejemplo de un árbol de respuestas de confirmación

de la totalidad de atributos obligatorios para el concepto actual. Para generar la respuesta del sistema se utiliza el conjunto de plantillas definidas para las consultas a la base de datos, teniéndose en cuenta la cardinalidad del resultado obtenido tras efectuar la consulta.

d) La frase en lenguaje natural se remite al módulo de síntesis de texto a voz.

La utilización de los árboles de respuestas simplifica el número de respuestas necesario para contemplar todas las respuestas definidas para la tarea. Asimismo, facilita la introducción de nuevos idiomas de una forma más sencilla.

4.7. Síntesis de texto a voz

La frase en lenguaje natural proporcionada por el generador de respuestas se remite a un módulo de conversión de texto a voz. Los investigadores de la Universidad de Zaragoza desarrollaron un sintetizador específico para el dominio del proyecto. La síntesis de texto a voz se lleva a cabo en dos fases. En la primera, se analiza el texto de entrada para generar su transcripción fonética, incluyendo información adicional sobre la duración, entonación y ritmo. La segunda fase procesa la salida obtenida en la fase anterior y genera las señales adecuadas. La forma de onda final se obtiene mediante la concatenación de segmentos de voz previamente grabados en forma de difonemas. Finalmente, la forma de onda se modifica para adaptarla a la prosodia del texto.

Así mismo, se ha realizado la integración del sistema de síntesis de voz multilingüe Festival (www.cstr.ed.ac.uk/projects/festival) desarrollado por investigadores de la CMU y del CSTR de la Universidad de Edimburgo. Festival ofrece un conjunto de APIs para la conversión texto a voz: a nivel de Shell, a partir de un intérprete de comandos, una librería de C++, desde Java y un interfaz para Emacs. En el proyecto FestVox de la CMU (festvox.org) se han desarrollado herramientas para generar nuevas voces.

El sistema está escrito en C++. Utiliza la librería Edinburgh Speech Tools para el diseño de la arquitectura a bajo nivel y un intérprete de comandos desarrollado mediante

el lenguaje de programación Scheme (SIOD). Se trata de un software libre.

4.8. Consulta a la base de datos

Para almacenar el sistema de información del proyecto DIHANA se ha diseñado una base de datos siguiendo el modelo relacional de datos. Para el diseño de esta base de datos se han analizado tanto las necesidades de información propias del proyecto DIHANA como las soluciones planteadas por otros sistemas reales de información sobre trenes. El resultado ha sido una base de datos donde la información está estructurada en once tablas distintas que contienen información sobre estaciones, tipos de trenes, clases de billetes, recorridos de trenes, precios de billetes y servicios al usuario, así como las interrelaciones existentes entre todos estos elementos.

La base de datos está diseñada de forma que no contenga información sobre viajes individuales, al contrario, se agrupan aquellos viajes que poseen orígenes, destinos, horarios y precios iguales, independientemente de su fecha. Este diseño permite tener en una base de datos manejable información sobre más de 400.000 viajes independientes. Como contrapartida se complica considerablemente la implementación de las consultas a la base de datos.

La mayor parte de la información de esta base de datos se corresponde con información real sobre trenes de largo recorrido obtenida de la página web de RENFE (www.renfe.es). La implementación se ha realizado utilizando el sistema de gestión de base de datos PostgreSQL (www.postgresql.org).

El módulo de consulta a la base de datos recibe como entrada una petición de información por parte del gestor de diálogo y le devuelve como resultado una estructura de datos representando la información solicitada. El módulo de consulta a la base de datos se encarga tanto de la consulta propiamente dicha como de la interpretación y estructuración de los resultados.

Una vez recibida una petición de información los pasos seguidos por este módulo son los siguientes:

1. Determinación del concepto sobre el que se desea obtener información. En función del concepto se realizará una consulta u otra a la base de datos.
2. Construcción de una consulta SQL en función de la información solicitada.
3. Conexión a la base de datos y realización de la consulta.
4. Procesamiento de la información obtenida. La información obtenida en el paso anterior debe ser procesada para obtener la información concerniente a los trayectos concretos solicitados. Finalmente, se realiza una adecuación de los datos a las estructuras esperadas por el gestor de diálogo como respuesta.

La Figura 4.8 muestra un ejemplo de generación de la sentencia SQL a partir de la entrada suministrada por el gestor de diálogo, correspondiente a la consulta realizada por el usuario.

```
(HORA-SALIDA)
CIUDAD-ORIGEN:barcelona
CIUDAD-DESTINO:madrid
HORA-SALIDA:8:30-17:00
FECHA-SALIDA:10-06-2004
TIPO-TREN:talgo arco

SELECT *
FROM Programacion RX
WHERE EXISTS (SELECT *
FROM Programacion PX, Estacion EO, Estacion ED
WHERE PX.id_prg = RX.id_prg AND
- ciudad de origen
PX.origen = EO.id AND EO.ciudad = 'barcelona' AND
- ciudad de destino
PX.destino = ED.id AND ED.ciudad = 'madrid' AND
- fecha de salida
PX.fecha_ini <= '2004-06-10' AND PX.fecha_fin >= '2004-06-10' AND
- hora de llegada
PX.hora_sal BETWEEN '8:30' AND '17:00')
AND
EXISTS (SELECT * FROM (Tren TX JOIN Servicio_ttren STX ON TX.tipo=STX.tren)
WHERE TX.id = RX.tren AND
- tipo de tren
(TX.tipo = 'talgo' OR TX.tipo = 'arco'));
```

Figura 4.8: Generación de la sentencia SQL

4.9. Módulo central de gestión de las comunicaciones

Este módulo es el responsable de establecer las comunicaciones con el resto de módulos del sistema, encargándose de recibir la totalidad de mensajes enviados (excepto los de audio) y de dirigirlos al servidor destino. De esta forma, actúa como encaminador de los mensajes que se transmiten los diversos módulos del sistema, siendo al mismo tiempo el encargado de establecer las comunicaciones.

Se ha diseñado un interfaz gráfico que permite monitorizar el funcionamiento del módulo central. A la hora de mostrar la información en pantalla se han diferenciado tres bloques de contenidos:

- Bloque de control: Muestra el estado de los diversos módulos que componen el sistema y la totalidad de mensajes transmitidos.
- Bloque de información al usuario: Muestra las frases reconocidas y respuestas generadas por el sistema tras cada una de las intervenciones del usuario.
- Bloque de configuración: Mediante este bloque pueden configurarse la dirección IP y el puerto utilizados para el establecimiento de los sockets con cada uno de los módulos del sistema.

La Figura 4.9 muestra el interfaz gráfico del módulo central gestor de las comunicaciones.

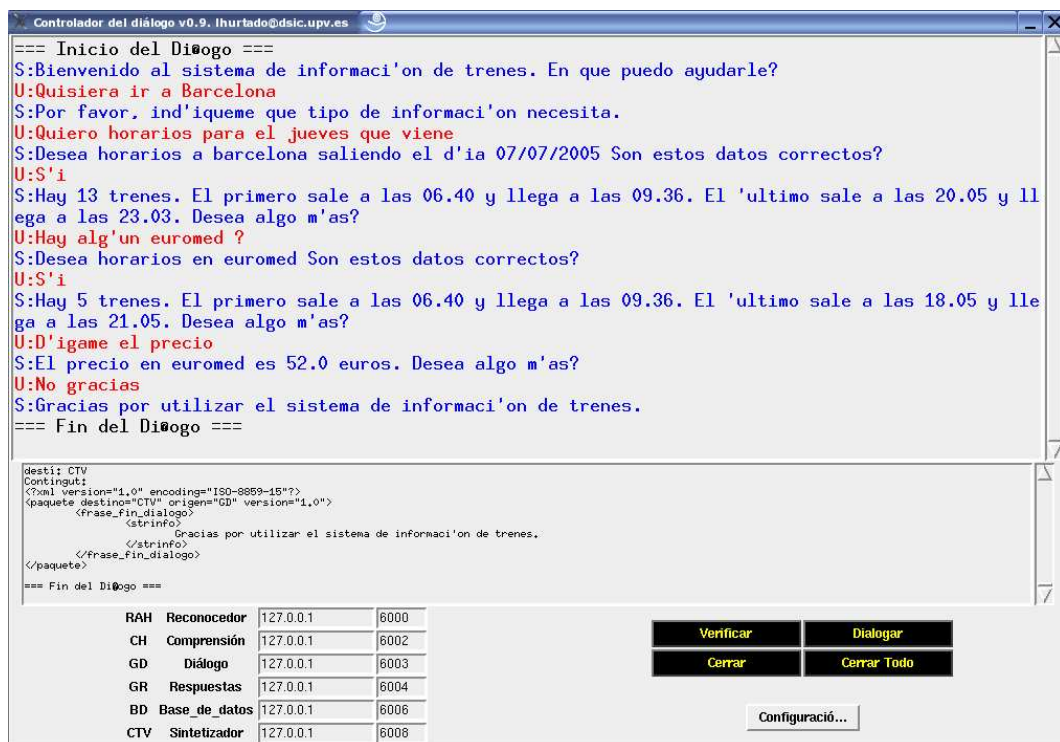


Figura 4.9: Modulo gestor de las comunicaciones del sistema de diálogo DIHANA

Capítulo 5

Gestión del diálogo mediante metodologías basadas en reglas

5.1. Introducción

A lo largo de este capítulo se detallan dos gestores de diálogo diseñados mediante modelos basados en reglas. La utilización de estos modelos supone una solución sencilla, rápida y robusta al problema de la gestión. El principal inconveniente radica en el número de limitaciones que debe imponerse a la tarea para que el número de reglas no crezca de manera indefinida. Para tareas complejas, la utilización de modelos basados en reglas deja de ser viable, ya que el número de casos posibles favorece el aprendizaje a través de modelos estadísticos antes de plantearse definir a mano el gran número de reglas necesario para resolver satisfactoriamente la gestión de diálogo.

La primera estrategia, que hemos denominado gestión Completa-Confirma, se basa en la solicitud de los valores mínimos para la realización de una determinada consulta y la posterior confirmación de los valores que el sistema tenga anotados como poco fiables.

El segundo gestor, denominado Dihana-Oz, implementa la estrategia definida para la adquisición del corpus de diálogo DIHANA mediante la técnica de Mago de Oz, en la que un humano imita el comportamiento del sistema. Esta estrategia se basa en la confirmación de los valores suministrados por el usuario cada vez que el gestor considera que su fiabilidad no es suficiente, utilizándose diferentes tipos de confirmación. En ambos casos se utilizan medidas de confianza, proporcionadas por el módulo de comprensión, para determinar el nivel de fiabilidad de los datos.

Las dos estrategias se evaluaron a partir de la realización de 120 diálogos y la posterior obtención de una serie de medidas comparativas. Los resultados de esta evaluación se recogen en el capítulo séptimo de la tesis.

Dado el gran número de trabajos que se desarrollan actualmente siguiendo el estándar VoiceXML dentro del ámbito de la aplicación de metodologías basadas en reglas para el diseño de sistemas de diálogo, el Anexo D resume las principales especificaciones de este estándar (Sección D.6) e incluye un resumen de las implementaciones de las especificaciones de interfaces vocales definidas por el W3C (Sección D.6). Asimismo, se muestra un ejemplo práctico de aplicación de VoiceXML para el desarrollo de una centralita telefónica (Sección D.6). Diferentes trabajos llevados a cabo para la incorporación de VoiceXML en el marco del proyecto DIHANA vienen reseñados en los artículos (López-Cózar y Granell,

2004) (López-Cózar et al., 2004).

Las estrategias desarrolladas en esta línea de trabajo han posibilitado la elaboración de tres publicaciones. La publicación (Griol et al., 2004) presenta el primero de los gestores desarrollados. En la publicación (Griol et al., 2005) se describe el segundo de los gestores y se realiza una evaluación comparativa de ambos. Por último, la publicación (Griol et al., 2005) compara el funcionamiento de estos gestores con una metodología de gestión estocástica desarrollada previamente en el contexto del proyecto BASURDE.

5.2. Definiciones comunes para el conjunto de gestores de diálogo desarrollados

5.2.1. Estandarización: gestores multitarea

La idea fundamental en la que se basan los modelos de gestión de diálogo desarrollados consiste en la conveniencia de elaborar gestores estándares, que posibiliten su adaptación a nuevas tareas, portables y con la capacidad de poder incorporar nuevos idiomas de forma sencilla. Esta suposición conlleva un punto añadido a las características de modularidad y portabilidad introducidas previamente para el desarrollo del sistema de diálogo.

La implementación práctica de estas ideas se materializa mediante la utilización de diccionarios y ficheros de definición de la semántica de la tarea, en los que se almacenan los parámetros fundamentales del sistema (conceptos definidos en la tarea, listado de atributos, atributos obligatorios para cada tipo de consulta...), y en la determinación de un formato estándar para la salida del gestor del diálogo. Todo ello viene apoyado por la definición de un protocolo de comunicación entre los módulos del sistema.

5.2.2. Técnicas de detección y corrección de errores

Tal y como se ha mencionado en el capítulo de introducción de la tesis, al partir cada módulo de la salida de los módulos previos, puede producirse fácilmente la propagación de errores entre los diferentes módulos del sistema. En un principio, el reconocedor de voz debe tratar los efectos del habla espontánea y de ambientes con ruido, con lo que usualmente la frase que devuelve a su salida incorporará errores de reconocimiento. A partir de la frase reconocida, el módulo de comprensión puede incorporar nuevos errores, provocando que la representación semántica de la información (entrada del gestor de diálogo) contenga errores.

Existen diferentes estrategias para realizar el tratamiento de estos posibles errores en la información de entrada de los módulos. Una primera alternativa consiste en la utilización de confirmaciones implícitas, únicamente válida para los casos en los que se dispone de información con alta fiabilidad, debido a que la confirmación implícita de datos erróneos genera confusión en el usuario.

La segunda estrategia consiste en la utilización de confirmaciones explícitas de los atributos, que posee como inconveniente el hecho de producir diálogos excesivamente largos, con un número excesivo de turnos de sistema.

La alternativa que utilizan los modelos de gestión desarrollados se basa en la utilización de confirmaciones implícitas únicamente para aquella información que se ha detectado que

tiene poca fiabilidad tanto en el proceso de reconocimiento como en comprensión. Para ello, es necesario suministrar al gestor de diálogo información indicando que datos se han reconocido y comprendido satisfactoriamente y cuáles no. Para llevar a cabo esta tarea se utilizan medidas de confianza. Disponiendo de valores de confianza de las palabras, suministradas por el reconocedor de voz, se pueden obtener medidas de confianza en comprensión asociadas a los atributos y conceptos.

En (García et al., 2003) se resume la aproximación desarrollada para la incorporación de medidas de confianza en el módulo de comprensión utilizado en el sistema de diálogo DIHANA. En esta aproximación, se consideran dos medidas de confianza para cada par (u_i, v_i) obtenido en la transducción.

La primera de ellas (AVG) se calcula combinando la media de la probabilidad del unigrama de cada palabra $w_{i_j} \in W$ en el segmento $u_i = w_{i_1}w_{i_2} \dots w_{i_{u_i}}$ asociado a una unidad semántica dada v_i ($P_{v_i}(w_{i_j})$), y la medida de confianza proporcionada por el reconocedor para la palabra ($P_r(w_{i_j})$).

El valor $P_r(w_{i_j})$ mide la semejanza entre la medida acústica de w_{i_j} obtenida durante el proceso de reconocimiento y la medida acústica de w_{i_j} proporcionada por un modelo específico basado en sílabas para dicha palabra. De esta forma, el valor AVG representa una medida de la pertenencia de cada palabra del segmento u_i al lenguaje asociado a la unidad semántica v_i , y la medida de confianza suministrada por el reconocedor. La medida AVG se define de la siguiente forma: sea $(u, v) = (u_1u_2 \dots u_n, v_1v_2 \dots v_n)$ el par asociado a una frase de entrada de W^* donde $u_i = (w_{i_1}w_{i_2} \dots w_{i_{u_i}})$. La medida AVG asociada al par (u_i, v_i) viene dada por:

$$AVG(u_i, v_i) = \lambda \left(\frac{\sum_{j=1}^{|u_i|} (P_{v_i}(w_{i_j}))}{|u_i|} \right) + (1 - \lambda) \left(\frac{\sum_{j=1}^{|u_i|} (P_r(w_{i_j}))}{|u_i|} \right)$$

donde el parámetro λ puede obtenerse experimentalmente.

La segunda medida es la probabilidad del modelo de lenguaje semántico (SLMP). Este valor representa la probabilidad de la unidad semántica v_i en el contexto de la secuencia de unidades semánticas $v_1v_2 \dots v_n$ asociada a la frase de entrada en el proceso de transducción. La medida SLMP se define de la siguiente forma: sea $(u, v) = (u_1u_2 \dots u_n, v_1v_2 \dots v_n)$ el par asociado a una frase de entrada de W^* y sea v_i la unidad semántica asociada al segmento u_i de la frase obtenida. La probabilidad del modelo de lenguaje semántico asociada al par (u_i, v_i) viene dada por $SMLP(u_i, v_i) = P(v_i | v_i - 1)$.

Tal y como se ha mencionado en el capítulo tercero de la tesis, el módulo de comprensión integrado en el sistema de diálogo DIHANA procesa en dos fases la frase generada por el reconocedor. En la primera fase, en la que se obtiene la segmentación de la frase en términos de una secuencia de unidades semánticas, las dos medidas de confianza se incorporan al modelo estocástico aprendido. De esta forma, esta fase provee a su salida una secuencia de unidades semánticas y sus correspondientes medidas de confianza asociadas.

La segunda fase del proceso de comprensión, en la que se aplican una serie de reglas para generar la representación final en forma de frames, obtiene a su salida un listado de conceptos y/o atributos con sus medidas de confianza asociadas. El criterio definido para realizar la asignación de estas medidas a cada una de unidades de la representación semántica es el siguiente:

- Si el concepto/atributo se ha generado a partir de un único segmento de los generados en la primera fase, el concepto/atributo hereda la medida de confianza asociada al segmento.
- Si el concepto/atributo se ha generado a partir de varios segmentos generados en la primera fase, su medida de confianza se obtiene de la media de las medidas de confianza correspondientes a los segmentos.

5.2.3. Métodos de simplificación semántica

Dado que el sistema de diálogo diseñado para DIHANA está orientado a completar los datos mínimos para hacer una única consulta a la base de datos, debe tratarse la situación en la que el usuario realiza más de una consulta en el mismo turno. En este caso, se aplica un preprocesado de la representación semántica del turno de usuario, de forma que se provea una entrada más flexible al gestor. La única simplificación llevada a cabo consiste en la selección de un único concepto cuando en la salida del módulo de comprensión hay varios disponibles. Las técnicas de simplificación semántica utilizadas, por tanto, conllevan la pérdida de información con respecto a la secuencia semántica suministrada por comprensión. El sistema se vale de dos técnicas:

- **Selección incondicional:** La secuencia de unidades semánticas suministradas por comprensión se simplifica teniendo en cuenta una jerarquía en la ordenación de los conceptos y seleccionando aquel que aparezca en un nivel superior. Para realizar la simplificación del número de conceptos se tiene en cuenta la aparición o no del concepto actual en la secuencia de conceptos existente en la salida generada por el módulo de comprensión. En caso de aparición, se podan el resto de conceptos de la secuencia.
- **Selección condicionada a las confianzas:** Al igual que en el método anterior, se utiliza la poda de conceptos para simplificar la representación semántica original. La técnica que se sigue a la hora de realizar esta poda consiste en ordenar los conceptos de acuerdo con sus medidas de confianza y descartar los menos fiables. Para ello, se tiene en cuenta que cada uno de los conceptos puede disponer de más de una medida de confianza asociada, calculándose la media de las medidas suministradas y permaneciendo aquel concepto que disponga de una fiabilidad mayor.

La Figura 5.1 muestra un ejemplo para los tipos de simplificación semántica descritos. Entre corchetes aparecen las dos medidas de confianza utilizadas para cada ítem (medidas sintáctica y semántica proporcionadas por el módulo de comprensión), calculándose la media a la hora de operar con las mismas.

Una alternativa eficiente a los métodos de simplificación definidos consiste en el encolado de los conceptos en lugar de su eliminación. De esta forma, una vez el gestor ha respondido a una consulta sobre un determinado concepto, consultaría la cola de conceptos restantes sobre los que el usuario solicita información para seleccionar uno de ellos siguiendo un criterio definido (el primero añadido a la cola, el último que se ha incorporado, el que disponga de una mayor confianza asociada...) y actuar de acuerdo con el nuevo concepto disponible. Además, las medidas de confianzas pueden ser útiles para decidir

Frase de entrada en lenguaje natural
Hola, podría decirme horarios y precios para ir a Bilbao
Representación semántica mediante conceptos y atributos
<i>(HORA-SALIDA) [0.53, 0.67]</i> <i>CIUDAD-DESTINO:Bilbao [0.77, 0.51]</i> <i>(PRECIO) [0.35, 0.42]</i> <i>CIUDAD-DESTINO:Bilbao [0.77, 0.51]</i>
Selección incondicional (Nota: El concepto actual es (PRECIO)).
<i>(PRECIO) [0.35, 0.42]</i> <i>CIUDAD-DESTINO:Bilbao [0.77, 0.51]</i>
Selección condicionada a las confianzas
<i>(HORA-SALIDA) [0.53, 0.67]</i> <i>CIUDAD-DESTINO:Bilbao [0.77, 0.51]</i>

Figura 5.1: Ejemplos de las técnicas de simplificación semántica definidas

el encolado o la eliminación de conceptos detectados por comprensión, de forma que se disponga de un umbral que indique que operación es conveniente realizar para cada uno de los conceptos detectados en una misma intervención de usuario.

5.2.4. Registro del diálogo

Para realizar correctamente su función, los gestores de diálogo se apoyan en una estructura de datos que denominamos **Registro de diálogo (RD)**, y que estructura la información en base a tres bloques diferenciados:

- **Concepto actual.** Por concepto actual, entendemos el concepto sobre el cual el usuario realiza la consulta. En el caso de detectar más de un concepto en la respuesta del módulo de comprensión, se aplican las técnicas de simplificación semántica detalladas.
- **Atributos.** Cada uno de los conceptos dispone de un conjunto de atributos obligatorios, considerándose el resto como modificadores de la consulta básica establecida para cada tipo de concepto.
- **Medidas de confianza asociadas.** Se almacenan las confianzas asociadas a conceptos y atributos, incluidas en la representación semántica generada por el módulo de comprensión.

Esta información es fundamental a la hora de establecer la estrategia del diálogo: decidir si es necesario realizar una confirmación y qué datos deben confirmarse, conocer qué atributos restan por completar para poder realizar la consulta, etc.

Para cada uno de los conceptos definidos para la tarea, se ha detallado un conjunto de atributos obligatorios, necesarios para poder realizar la consulta a la base de datos. Dicho conjunto se muestra en la Figura 5.2.

CONCEPTOS	ATRIBUTOS MÍNIMOS
<i>HORA-SALIDA</i> <i>HORA-LLEGADA</i>	<i>CIUDAD-ORIGEN</i> <i>CIUDAD-DESTINO</i> <i>FECHA-SALIDA</i>
<i>PRECIO</i>	<i>CIUDAD-ORIGEN</i> <i>CIUDAD-DESTINO</i> <i>FECHA-SALIDA</i> <i>CLASE-BILLETE</i>
<i>TIEMPO-RECORRIDO</i>	<i>CIUDAD-ORIGEN</i> <i>CIUDAD-DESTINO</i>
<i>TIPO-TREN</i>	<i>CIUDAD-ORIGEN</i> <i>CIUDAD-DESTINO</i>
<i>SERVICIOS</i>	<i>CLASE-BILLETE</i> <i>TIPO-TREN</i>

Figura 5.2: Atributos obligatorios definidos para cada uno de los conceptos en DIHANA

La solicitud de atributos obligatorios incluye la totalidad de atributos mínimos restantes, pudiéndose completar el valor de todos ellos en un único turno de usuario. Por último, merece destacarse que la iniciativa del diálogo es mixta, teniéndose en cuenta toda la información aportada por el usuario independientemente de la pregunta realizada por el sistema.

5.3. Estrategia Completa-Confirma

El primer modelo desarrollado para la gestión de diálogo sigue un proceso iterativo en base al siguiente algoritmo:

1. Lee la salida del módulo de comprensión (representación semántica del turno de usuario mediante frames).
2. Realiza la extracción de conceptos y atributos a partir de la salida de comprensión.
3. Actualiza el registro del diálogo con los datos proporcionados por el usuario.
4. Actualiza el registro del diálogo cuando se realiza una consulta a la base de datos. La totalidad de los resultados obtenidos al realizar una consulta a la base de datos se almacenan en el registro del diálogo.
5. Genera la salida, siguiendo un formato definido, y la suministra al generador de respuestas, para que construya la respuesta asociada.

5.3.1. Ejemplo de Diálogo

Las siguientes figuras muestran un ejemplo de diálogo adquirido utilizando el gestor de diálogo Completa-Confirma. Las siglas utilizadas se corresponden con *U* (Entrada de Usuario), *S* (Salida del Generador de Respuestas), *DR* (Registro del Diálogo) y *C* (Representación semántica generada por el módulo de comprensión).

En primer lugar, el diálogo se inicializa con un *prompt* de presentación. Los atributos disponibles en el registro inicial son *CIUDAD-ORIGEN* y *CLASE-BILLETE*, como se observa en la Figura 5.3.

<p>DR_0: CIUDAD-ORIGEN:Valencia [1] CLASE-BILLETE:turista [1] S_1: Bienvenido al sistema de información de trenes. ¿En qué puedo ayudarle?</p>
--

Figura 5.3: Inicialización del diálogo

Tras cada intervención del usuario, se extraen los conceptos y atributos de la secuencia semántica suministrada por comprensión. En caso de no detectar ningún concepto, el generador de respuestas devuelve una respuesta estándar (Figura 5.4).

<p>U_1: Quiero ir a Bilbao. C_1: () [0,81, 0,77] CIUDAD-DESTINO:Bilbao [0,33, 0,36] DR_1: CIUDAD-ORIGEN:Valencia [1] CLASE-BILLETE:turista [1] CIUDAD-DESTINO:Bilbao [0,35] S_2: Por favor, indíqueme que tipo de información necesita.</p>
--

Figura 5.4: Solicitud del tipo de consulta

Una vez el gestor ha detectado un concepto en la interpretación semántica del turno de usuario, se solicitan los atributos obligatorios para dicho concepto, y de cuyo valor no se disponga en el estado actual del diálogo, como muestra la Figura 5.5.

<p>U_2: Quisiera horarios. C_2: (HORA-SALIDA) [0,71, 0,67] DR_2: (HORA-SALIDA) [0,69] CIUDAD-ORIGEN:Valencia [1] CLASE-BILLETE:turista [1] CIUDAD-DESTINO:Bilbao [0,35] S_3: Dígame la fecha de salida.</p>

Figura 5.5: Solicitud de los atributos obligatorios

Tras completar la totalidad de atributos obligatorios, se realiza una confirmación explícita, aplicable únicamente a aquellos atributos cuya medida de confianza asociada sea inferior al umbral predeterminado. Para el ejemplo citado, se observa que el valor de la media de las medidas de confianza de los atributos *CIUDAD-DESTINO* y *FECHA-SALIDA* es inferior al umbral 0.5, fijado para el ejemplo (Figura 5.6).

U_3 : Me gustaría viajar el lunes.
 C_3 : () [0,11, 0,45]
 FECHA-SALIDA:18-04-2007 [0,43, 0,32]
 DR_3 : (HORA-SALIDA) [0,69]
 CIUDAD-ORIGEN:Valencia [1]
 CLASE-BILLETE:turista [1]
 CIUDAD-DESTINO:Bilbao [0,35]
 FECHA-SALIDA:18-04-2007 [0,38]
 S_4 : ¿Desea horarios saliendo el día 18-05-2007 desde Bilbao?

Figura 5.6: Confirmación explícita basada en las medidas de confianza

En caso de recibir una confirmación positiva por parte del usuario, se realiza la consulta a la base de datos, actualizando el registro del diálogo y mostrando el resultado de la consulta conjuntamente con una respuesta estándar, como se observa en la Figura 5.7.

U_4 : Sí.
 C_4 : (AFIRMACIÓN) [0,61, 0,85]
 DR_4 : (HORA-SALIDA) [0,69]
 CIUDAD-ORIGEN:Valencia [1]
 CLASE-BILLETE:turista [1]
 CIUDAD-DESTINO:Bilbao [1]
 FECHA-SALIDA:18-04-2007 [1]
 S_5 : Hay varios trenes. El primero sale a las 08:54 y el último a las 23:45. ¿Desea algo más?

Figura 5.7: Resultado de la consulta a la base de datos

El diálogo finaliza tras detectar el concepto *CIERRE* en la representación semántica generada por comprensión (Figura 5.8).

U_5 : No gracias. Adiós.
 C_5 : (CIERRE) [0,61, 0,55]
 DR_5 : (HORA-SALIDA) [0,69]
 CIUDAD-ORIGEN:Valencia [1]
 CLASE-BILLETE:turista [1]
 CIUDAD-DESTINO:Bilbao [1]
 FECHA-SALIDA:18-04-2007 [1]
 S_6 : Gracias por utilizar el sistema de información de trenes.

Figura 5.8: Finalización del diálogo

5.4. Estrategia Dihana-Oz

El segundo gestor desarrollado, Dihana-Oz, implementa la estrategia que se elaboró para la adquisición con la técnica de Mago de Oz del corpus DIHANA. En esta estrategia, dados el estado del registro de diálogo y la información proporcionada por el usuario en el turno correspondiente, el gestor interacciona con el usuario en base a los niveles de confianza suministrados por el sistema. Las acciones que puede seleccionar el gestor se resumen a continuación:

- Si todos los datos del registro del diálogo disponen de una medida de confianza asociada mayor que el umbral fijado (estado seguro), el gestor elige una de las tres interacciones siguientes:
 1. Confirmación Implícita y Consulta a la base de datos si dispone de concepto y, al menos, de los valores de sus atributos mínimos.
 2. Completar si no dispone de valor para el concepto actual y/o alguno de sus atributos mínimos (sin valor por defecto).
 3. Confirmación Mixta, para dar una mayor naturalidad en el diálogo.
- Si el estado es inseguro (aquel en el que uno o más datos del registro del diálogo poseen una confianza menor que el umbral), el gestor selecciona una de las dos interacciones siguientes:
 1. Confirmación Explícita del primero de los ítem inciertos que aparezca en el registro del diálogo.
 2. Confirmación Mixta para darle naturalidad al diálogo.

Para realizar la implementación de la estrategia descrita cumplen un papel esencial el registro del diálogo y los métodos de simplificación semántica, descritos previamente. Asimismo, el almacenamiento de la última respuesta generada por el sistema permite resolver satisfactoriamente las confirmaciones de conceptos y atributos.

5.4.1. Ejemplo de Diálogo

Las siguientes figuras muestran un ejemplo de diálogo adquirido con el gestor Dihana-Oz. Las siglas utilizadas se corresponden con las citadas para el modelo de gestión anterior.

En primer lugar, el diálogo se inicializa con un *prompt* de presentación. Los atributos disponibles en el registro inicial son *CIUDAD-ORIGEN* y *CLASE-BILLETE*, como se observa en la Figura 5.9.

<p>DR_O: CIUDAD-ORIGEN:Valencia [1] CLASE-BILLETE:turista [1] S_1: Bienvenido al sistema de información de trenes. ¿En qué puedo ayudarle?</p>
--

Figura 5.9: Inicialización del diálogo

En caso de no detectar ningún concepto tras la intervención del usuario, el gestor de diálogo realiza la petición del tipo de consulta (Figura 5.10).

<p>U_1: Quiero ir a Bilbao. C_1: () [0,81, 0,77] CIUDAD-DESTINO:Bilbao [0,33, 0,36] DR_1: CIUDAD-ORIGEN:Valencia [1] CLASE-BILLETE:turista [1] CIUDAD-DESTINO:Bilbao [0,35] S_2: ¿Quiere saber horarios?</p>

Figura 5.10: Solicitud del tipo de consulta

Una vez el gestor confirma un concepto en la interpretación semántica del turno de usuario, comprueba si existen en el registro valores con una confianza inferior al umbral prefijado. Para el ejemplo, se observa que el valor de la media de las medidas de confianza del atributo *CIUDAD-DESTINO* es inferior al umbral 0.5, generándose una confirmación del valor correspondiente (Figura 5.11).

<p>U_2: Sí. C_2: (AFIRMACIÓN) [0,61, 0,85] DR_2: (HORA-SALIDA) [1] CIUDAD-ORIGEN:Valencia [1] CLASE-BILLETE:turista [1] CIUDAD-DESTINO:Bilbao [0,35] S_3: ¿Desea ir a Bilbao?</p>

Figura 5.11: Confirmación de los atributos

Tras recibir una confirmación positiva, y comprobar que no existen valores en el registro con baja fiabilidad, se solicitan los atributos obligatorios para completar la consulta solicitada, como muestra la Figura 5.12.

<p>U_3: Sí. C_3: (AFIRMACIÓN) [0,71, 0,67] DR_3: (HORA-SALIDA) [1] CIUDAD-ORIGEN:Valencia [1] CLASE-BILLETE:turista [1] CIUDAD-DESTINO:Bilbao [1] S_4: ¿Qué día desea salir?</p>
--

Figura 5.12: Solicitud de los atributos obligatorios

Tras incorporar al registro del diálogo atributos cuya medida de confianza asociada sea inferior al umbral predeterminado, el gestor selecciona de nuevo uno de los posibles estados de confirmación (Figura 5.13).

*U*₄: Me gustaría viajar el cuatro de mayo.
*C*₄: () [0,34, 0,43]
 FECHA-SALIDA:04-05-2007 [0,33, 0,46]
*DR*₄: (HORA-SALIDA) [1]
 CIUDAD-ORIGEN:Valencia [1]
 CLASE-BILLETE:turista [1]
 CIUDAD-DESTINO:Bilbao [1]
 FECHA-SALIDA:04-05-2007 [0,40]
*S*₅: ¿Desea salir el 04-05-2007?

Figura 5.13: Confirmación de los atributos

En caso de recibir una confirmación positiva por parte del usuario, se realiza la consulta a la base de datos y se genera una respuesta que muestre el resultado de la misma, tal y como se observa en la Figura 5.14.

*U*₅: Sí.
*C*₅: (AFIRMACIÓN) [0,61, 0,85]
*DR*₅: (HORA-SALIDA) [1]
 CIUDAD-ORIGEN:Valencia [1]
 CLASE-BILLETE:turista [1]
 CIUDAD-DESTINO:Bilbao [1]
 FECHA-SALIDA:04-05-2007 [1]
*S*₆: Un momento por favor. Hay varios trenes. El primero sale a las 08:54 y el último a las 23:45 . ¿Desea algo más?

Figura 5.14: Resultado de la consulta a la base de datos

El diálogo finaliza tras detectar el concepto *CIERRE* en la representación semántica generada por comprensión (Figura 5.15).

*U*₆: No, gracias. Adiós.
*C*₆: (CIERRE) [0,61, 0,55]
*DR*₆: (HORA-SALIDA) [1]
 CIUDAD-ORIGEN:Valencia [1]
 CLASE-BILLETE:turista [1]
 CIUDAD-DESTINO:Bilbao [1]
 FECHA-SALIDA:04-03-2007 [1]
*S*₇: Gracias por utilizar el sistema de información de trenes.

Figura 5.15: Finalización del diálogo

La observación de los ejemplos expuestos para las dos estrategias de diálogo muestra la importancia de conocer el tipo de consulta que desea realizar el usuario antes de realizar otras operaciones. Sin embargo, mientras la primera de las técnicas espera a completar

el conjunto de atributos mínimos antes de confirmar valores del registro del diálogo con baja confianza, la segunda de ellas confirma dichos valores nada más se han incorporado al registro. Este hecho favorece que el número medio de turnos de sistema y de confirmaciones explícitas sea mayor en la segunda estrategia. El capítulo séptimo de la tesis muestra los resultados de la evaluación de los gestores basados en reglas descritos en este capítulo.

Capítulo 6

Gestión del diálogo mediante metodologías estadísticas

6.1. Introducción

Este capítulo resume el trabajo llevado a cabo para desarrollar una metodología estadística para la gestión de diálogo, aprendida a partir de un corpus de diálogos.

Tal y como se ha comentado en el capítulo de introducción de la tesis, la utilización de técnicas estadísticas para el desarrollo de los diferentes módulos que componen un sistema de diálogo tiene un interés creciente durante los últimos años. Estas aproximaciones suelen basarse en la modelización de los diferentes procesos de forma probabilística, estimando los parámetros correspondientes del modelo a partir de un corpus de diálogos.

La motivación por entrenar modelos estadísticos a partir de datos reales es clara. Los avances en el campo de los sistemas de diálogo hacen que los procesos de diseño, implementación y evaluación de las estrategias de gestión del diálogo sean cada vez más complejos, lo que ha posibilitado que el foco de interés de la comunidad científica se desplace de forma creciente de los métodos empíricos a las técnicas basadas en modelos aprendidos a partir de datos. Estos modelos pueden entrenarse a partir de diálogos reales, pudiendo modelar de esta forma la variabilidad en los comportamientos de los usuarios. El objetivo de estas metodologías es desarrollar sistemas con un comportamiento más robusto, con mayor portabilidad, escalables y que presenten un gran número de ventajas de cara a su adaptación al usuario o a nuevas tareas.

Este tipo de metodologías se han aplicado tradicionalmente dentro de los campos de reconocimiento automático del habla y comprensión del lenguaje. La aplicación de metodologías estadísticas para modelar el comportamiento del gestor de diálogo está proporcionando resultados interesantes en años más recientes, existiendo un número reducido de aproximaciones que modelan el diálogo de forma estadística.

La metodología estadística desarrollada para esta tesis se basa en la modelización de las secuencias de actos de diálogos de usuario y de sistema, y la introducción de una partición en el espacio de todas las posibles secuencias de actos de diálogo, de forma que el proceso de aprendizaje del modelo estadístico a partir de un corpus de diálogos sea manejable. Esta partición se define teniendo en cuenta la información suministrada por el usuario a lo largo del diálogo. Para su definición, además, se consideran las medidas de confianza proporcionadas por el reconocedor automático del habla y el módulo de

comprensión.

Los trabajos que se relatan en el capítulo han dado lugar a las siguientes publicaciones científicas (Hurtado et al., 2005), (Castro et al., 2005), (Granell et al., 2005), (Griol et al., 2006), (Hurtado et al., 2006), (Griol et al., 2006a), (Griol et al., 2006b).

6.2. Descripción formal de la metodología estadística de gestión del diálogo

Tal y como se ha comentado previamente, la metodología estadística que se propone para desarrollar un gestor de diálogo se basa en la modelización estadística de las secuencias de actos de diálogo (de usuario y de sistema). El aprendizaje del gestor se realiza a partir de un corpus de diálogos etiquetado en actos de diálogo.

El gestor de diálogo implementado genera las respuestas del sistema basándose únicamente en la información suministrada por los turnos de usuario y la información contenida en el modelo. Las fuentes de información consideradas en el modelo son: el contenido del registro del diálogo, la última respuesta generada por el sistema y la representación semántica de la última intervención del usuario. Así mismo, la salida generada por el gestor se corresponde con uno de los posibles actos de diálogo definidos como conjunto de respuestas del sistema.

En este tipo de aproximación, la probabilidad de obtener un buen modelo varía dependiendo del número de actos de diálogo definidos y, por tanto, de la cantidad de información representada por cada acto de diálogo. Si se considera únicamente un número reducido de actos de diálogo que representen acciones generales en el diálogo, puede llevarse a cabo el aprendizaje de un buen modelo, pero la información representada en él no sería suficiente para gestionar de forma completa el diálogo. En este caso, se debería proporcionar al gestor la información específica relacionada con la tarea mediante un conjunto de reglas fijadas a mano.

Sin embargo, etiquetando los turnos mediante actos de diálogo que consideren no únicamente los fines generales de las frases del usuario, sino también las peticiones específicas relacionadas con la tarea (los conceptos y valores de los atributos proporcionados en el turno), puede obtenerse un gestor operativo que modelice las secuencias de los turnos detallados de usuario y de sistema. Lógicamente, en este caso, el principal problema es que el número de los actos del diálogo aumenta exponencialmente con relación al número de conceptos (y de atributos), y el espacio de las diversas situaciones del diálogo que deben tenerse en cuenta es demasiado grande.

Una descripción formal del modelo estadístico propuesto es la siguiente:

Sea A_i la salida del sistema de diálogo (turno de sistema) en el instante i , expresada en términos de actos de diálogo. Sea U_i la representación semántica del turno de usuario (la salida generada por el módulo de comprensión para la intervención del usuario) en el instante i , expresada en términos de frames. El diálogo comienza con un turno de sistema que da la bienvenida al usuario y le ofrece sus servicios (A_1). El diálogo puede representarse mediante una secuencia de pares (*turno-sistema, turno-usuario*):

$$(A_1, U_1), \dots, (A_i, U_i), \dots, (A_n, U_n)$$

donde A_1 es el turno de bienvenida del sistema, y U_n es el último turno de usuario. De ahora en adelante, se denota el par (A_i, U_i) como S_i , el estado de la secuencia del diálogo en el instante i .

En este contexto, consideramos que, en el instante i , el objetivo del gestor del diálogo es encontrar la mejor respuesta de sistema A_i . Esta selección es un proceso local para cada instante i y tiene en cuenta la secuencia de estados de diálogo que preceden a dicho instante. Esta selección se realiza maximizando:

$$\hat{A}_i = \operatorname{argmax}_{A_i \in \mathcal{A}} P(A_i | S_1, \dots, S_{i-1}) \quad (6.1)$$

donde el conjunto \mathcal{A} contiene todas las posibles respuestas del sistema.

6.3. Definición y codificación del registro del diálogo

Tal y como se ha visto en el capítulo anterior, definimos el registro del diálogo (*dialog register* (DR)) como una estructura de datos que contiene la información sobre los valores de los conceptos y atributos suministrados por el usuario a través de la historia previa del diálogo. Toda la información almacenada en el DR_i (registro del diálogo en un instante dado i) es un resumen de la información suministrada por el usuario a lo largo de la secuencia S_1, \dots, S_{i-1} . Cabe destacar que diferentes secuencias de estados pueden conducir al mismo DR . La definición del registro del diálogo nos permite considerar la historia completa del diálogo en la selección de la nueva respuesta del sistema, disponiendo de la totalidad de información relativa a la tarea aportada por el usuario hasta el instante actual del diálogo. De este modo, la adaptación de la metodología en el ámbito de una tarea específica consiste en considerar en el registro del diálogo el conjunto de conceptos y atributos definido en la semántica de la misma.

Para una secuencia de estados de un diálogo, existe su correspondiente secuencia de DR :

$$\begin{array}{ccccccc} S_1, & \dots, & S_i, & \dots, & S_n \\ \uparrow & & \uparrow & & \uparrow & & \uparrow \\ DR_0 & & DR_1 & & DR_{i-1} & & DR_{n-1} \end{array}$$

donde DR_1 contiene la información inicial por defecto del gestor del diálogo (*Origen* y *Clase* en el caso de DIHANA), y los siguientes registros DR_i se actualizan teniendo en cuenta la información suministrada por el usuario durante la evolución del diálogo.

Dado que el número de posibles secuencias de estados es muy grande, teniendo en cuenta el concepto de DR podemos establecer una partición en el espacio de las secuencias de estados, es decir, en la historia del diálogo hasta el instante i . La partición se realiza bajo la siguiente premisa: *dos secuencias de estados se consideran equivalentes si conducen al mismo DR* . De este modo, obtenemos una gran reducción en el número de historias diferentes en los diálogos a expensas de la pérdida del orden cronológico en el que se suministró la información. Consideramos que ésta es una pérdida menor, dado que el orden en el que el usuario suministró la información no es un factor relevante para determinar la próxima respuesta del sistema A_i , sino únicamente de qué información se dispone hasta el instante actual del diálogo y que confianza se le ha asignado a cada uno de los conceptos y atributos proporcionados por el usuario.

Tras aplicar las consideraciones anteriores y establecer la relación de equivalencia en las historias de los diálogos, la selección de la mejor A_i viene dada por:

$$\hat{A}_i = \operatorname{argmax}_{A_i \in \mathcal{A}} P(A_i | DR_{i-1}, S_{i-1}) \quad (6.2)$$

Cada turno de usuario suministra al sistema información relativa a la tarea, es decir, el usuario solicita información sobre un concepto específico y/o suministra los valores de determinados atributos. No obstante, un turno de usuario puede además aportar otro tipo de información, como por ejemplo información independiente de la tarea. Éste es el caso de los turnos correspondientes a los actos de diálogo *Afirmación*, *Negación* y *No-Entendido*. Este tipo de información implica una toma de decisiones diferente a una mera actualización del *DR*. Por ejemplo, si el usuario aporta en un determinado turno el acto de diálogo *Afirmación*, es necesario conocer a qué concepto/s o atributo/s le ha solicitado confirmación el sistema en el turno anterior. Por esta razón, para la selección de la mejor respuesta del sistema A_i , tenemos en cuenta el *DR* generado desde el turno 1 al turno $i - 1$, y explícitamente consideramos el último estado S_{i-1} .

Para la tarea DIHANA, el *DR* se ha definido como una secuencia de 15 campos, cada uno de ellos asociado a un determinado concepto o atributo. Los conceptos y atributos que componen el *DR* se muestran en la Figura 6.1.

Conceptos	Atributos
<i>Hora</i>	<i>Origen</i>
<i>Precio</i>	<i>Destino</i>
<i>Tipo-Tren</i>	<i>Fecha-Salida</i>
<i>Tiempo-Recorrido</i>	<i>Fecha-Llegada</i>
<i>Servicios</i>	<i>Hora-Salida</i>
	<i>Hora-Llegada</i>
	<i>Clase</i>
	<i>Tipo-Tren</i>
	<i>Número-Orden</i>
	<i>Servicios</i>

Figura 6.1: Estructura del registro del diálogo (*DR*) definido para la tarea DIHANA

Para que el gestor de diálogo determine la siguiente respuesta, asumimos que no son significativos los valores exactos de los atributos. Estos valores son importantes para acceder a la base de datos y construir la respuesta del sistema en lenguaje natural. Sin embargo, la única información necesaria para determinar la siguiente acción del sistema es la presencia o no de conceptos y atributos, así como su nivel de confianza. Por tanto, la información que almacena el *DR* es una codificación de cada uno de sus campos en términos de tres valores, $\{0, 1, 2\}$, de acuerdo con el siguiente criterio:

- **0:** El usuario no ha realizado una consulta sobre el concepto o no ha proporcionado el valor del atributo correspondiente.
- **1:** El concepto o atributo está presente con una medida de confianza superior a un umbral prefijado (un valor entre 0 y 1). Las medidas de confianza se generan

durante los procesos de reconocimiento y comprensión, tal y como se ha explicado en el capítulo anterior.

- **2:** El concepto o atributo está presente, pero con una medida de confianza inferior al umbral.

De este modo, cada DR puede representarse como una cadena de longitud 15 cuyos elementos pueden tomar valores del conjunto $\{0, 1, 2\}$.

La Tabla 6.1 muestra la reducción del número de estados que se consigue para el corpus DIHANA con la introducción del DR .

DR distintos	261
Pares (DR, S) distintos	1.212
Secuencias S_1, \dots, S_{i-1}	4.290

Tabla 6.1: Reducción del espacio de estados tras la introducción del DR en DIHANA

6.4. Soluciones propuestas para tratar el problema de la cobertura del modelo

Las aproximaciones basadas en la estimación de un gestor a partir de un corpus de datos deben tratar el problema de modelar todas las posibles situaciones que pueden ocurrir durante el diálogo, utilizando únicamente un corpus de aprendizaje. Una vez se ha aprendido el modelo, se dispone de una representación de las secuencias de pares (DR, S) que se han observado durante la fase de entrenamiento. Si se llevan a cabo diálogos que contemplan secuencias vistas en el corpus de entrenamiento, el sistema funcionará correctamente. No obstante, el gestor del diálogo puede bloquearse si durante el transcurso de un diálogo se alcanza una situación (par (DR, S)) no vista durante la fase de entrenamiento. Por tanto, debe tenerse en cuenta la posibilidad de que el usuario proporcione una situación no prevista en el modelo. Para realizar la obtención de la mejor respuesta de sistema, mediante el uso de la Ecuación 6.2, y contemplar el problema de la cobertura del modelo, se han desarrollado dos estrategias:

- En la primera aproximación, las probabilidades del modelo propuesto se obtienen mediante una estimación por máxima verosimilitud a partir del corpus de diálogos etiquetados. En el caso de que se produzca una situación no vista durante el entrenamiento del modelo, se aplica una función distancia para la selección del par (DR, S) más cercano en el modelo a aquel en el que se ha producido la situación no vista.
- Una segunda alternativa se basa en modelar el problema de la cobertura mediante un proceso de clasificación. El tratamiento de la cobertura se basa en la realización de una nueva partición en el espacio de secuencias de actos de diálogo posibles. Cada clase de esta partición agrupa todas las secuencias que proporcionan la misma respuesta del sistema. Durante el transcurso de un nuevo diálogo, cuando se alcanza

una situación no observada durante el aprendizaje, ésta se clasifica en una de las clases de este conjunto, y la respuesta del sistema en ese momento es la respuesta asociada a la clase seleccionada. La función de clasificación puede definirse de diferentes formas. En el marco de la tesis se han evaluado cuatro definiciones de esta función: un clasificador naive Bayes multinomial, clasificadores basados en n-gramas, un clasificador basado en la técnica de inferencia gramatical MGGI y un clasificador basado en redes neuronales.

6.4.1. Definición de una función distancia

Tal y como se ha comentado, en esta aproximación, las probabilidades del modelo propuesto se obtienen mediante una estimación por máxima verosimilitud a partir del corpus de diálogos etiquetados. En el caso de que se produzca una situación (un par (DR, S)) no vista durante el entrenamiento del modelo, se aplica una función distancia para que el gestor de diálogo seleccione el par (DR', S') más cercano que esté incluido en el modelo. La distancia que definimos se corresponde con la siguiente expresión:

$$d((DR, S), (DR', S')) \approx d(DR, DR') = \sum_{k=1}^n f(dr_k, dr'_k) \quad (6.3)$$

En la definición de esta función asumimos:

- La distancia es independiente de los términos S y S' .
- En relación con la distancia entre los registros de diálogo codificados DR y DR' , en la definición de la función f se han considerado que la inserción de un valor de un atributo que realmente no ha sido proporcionado por el usuario tiene una mayor penalización que la eliminación de un valor para dicho atributo. De este modo, es más conveniente requerir de nuevo al usuario información que ya haya proporcionado previamente, que utilizar como segura información que el usuario no ha proporcionado realmente. El conjunto de substituciones posibles se muestra seguidamente:
 - Substitución de un 0 por un 1: Se trata de la incorporación de datos con un nivel de fiabilidad suficiente. De este modo, se incorpora al registro información fiable no aportada realmente por el usuario.
 - Substitución de un 0 por un 2: Se corresponde con la incorporación de datos con una baja fiabilidad.
 - Substitución de un 1 por un 0: Consiste en la eliminación de datos marcados con una fiabilidad suficiente.
 - Substitución de un 1 por un 2: Se trata de la disminución de la fiabilidad del valor de la información.
 - Substitución de un 2 por un 1: Aumento de la fiabilidad del valor de información correspondiente.
 - Substitución de un 2 por un 0: Eliminación de un dato con poca fiabilidad.

La Figura 6.2 muestra los pesos obtenidos experimentalmente para tratar cada una de las substituciones descritas. Para la obtención de los valores de la matriz, se realizó una experimentación consistente en variar los pesos y comparar la respuesta proporcionada automáticamente por el gestor con la de referencia anotada en el corpus.

		<u>DR</u>		
		0	1	2
<u>DR'</u>	0	0	150	25
	1	1000	0	200
	2	100	100	0

Figura 6.2: Matriz de pesos para el cálculo de la distancia entre secuencias

En la Figura 6.3 puede observarse un ejemplo de obtención de las cinco secuencias existentes en el corpus con menor distancia a una secuencia generada por el gestor, no presente en el corpus obtenido en la adquisición de DIHANA.

<u>Secuencia generada por el gestor:</u>	
Pregunta-Día*1110021200-10000*001	
<u>Conjunto de secuencias con distancia menor:</u>	
SECUENCIA	DISTANCIA
Pregunta-Día*1110021 0 00-10000*001	25
Pregunta-Día*11100 0 1000-10000*001	50
Pregunta-Día*11100 0 1100-10000*001	225
Pregunta-Día*11100 1 000-10000*001	225
Pregunta-Día*111 1 011100-10000*001	1400

Figura 6.3: Ejemplo de cálculo de las secuencias con menor distancia

En la Figura 6.4 se muestra un ejemplo de diálogo generado siguiendo esta primera alternativa. Para cada turno de usuario puede observarse la información codificada que utiliza el gestor del diálogo para seleccionar la siguiente acción. Esta codificación se corresponde con los últimos actos de diálogo del sistema (A_i); los actos de diálogo *Afirmación*, *Negación* o *No-Entendido* presentes en el turno de usuario, en caso de que existan (U_i); y el *DR* actualizado con la información suministrada por el usuario. Como se ha comentado previamente, el primer turno contiene información por defecto correspondiente a los atributos *CIUDAD-ORIGEN* y *CLASE-BILLETE*.

<i>MD:</i>	A_1 : (Apertura:Nil:Nil)	U_1 : ()	<i>DR:</i> 000000-1000001000
<i>Sistema</i> ₁ :	Bienvenido al sistema de información de trenes. ¿En qué puedo ayudarle?		
<i>Usuario</i> ₁ :	Quiero ir a Barcelona.		
<i>MD:</i>	A_2 : (Apertura)	U_1 : ()	<i>DR:</i> 00000-1100001000
	\hat{A}_2 : (Confirmacion:Hora-Salida:Nil)		
<i>Sistema</i> ₂ :	¿Quiere saber horarios?		
<i>Usuario</i> ₂ :	Sí, en Euromed.		
<i>MD:</i>	A_2 : (Confirmación)	U_2 : (Afirmación)	<i>DR:</i> 10000-1100001200 - situación no vista - <i>DR'</i> : 10000-1100001000
	\hat{A}_3 : (Question:Fecha-Salida:Nil)		
<i>Sistema</i> ₃ :	¿Qué día desea salir?		
<i>Usuario</i> ₃ :	El uno de agosto.		
<i>MD:</i>	A_3 : (Pregunta)	U_3 : ()	<i>DR:</i> 100000-1120001200
	\hat{A}_4 : (Confirmacion:Fecha-Salida:Fecha-Salida)		
<i>Sistema</i> ₄ :	¿Desea salir el 01/08/2005?		
<i>Usuario</i> ₄ :	Sí.		
<i>MD:</i>	A_4 : (Confirmación)	U_4 : (Afirmación)	<i>DR:</i> 100000-1110001200
	\hat{A}_5 : (Confirmacion:Tipo-Tren:Tipo-Tren)		
<i>Sistema</i> ₅ :	¿Desea viajar en un Euromed?		
<i>Usuario</i> ₅ :	Sí.		
<i>MD:</i>	A_5 : (Confirmación)	U_5 : (Afirmación)	<i>DR:</i> 10000-1110001100
	\hat{A}_6 : (Respuesta:Hora-Salida:Numero-Trenes,Tipo-Tren,Hora-Salida)(Nueva-Consulta:Nil:Nil)		
<i>Sistema</i> ₆ :	Hay varios trenes. El primero sale a las 08:54 y el último a las 23:45. ¿Desea algo más?		
<i>Usuario</i> ₆ :	No, gracias.		
<i>MD:</i>	A_6 : (Respuesta,Nueva-Consulta)	U_6 : (Negación)	<i>DR:</i> 10000-1110001100
	\hat{A}_7 : (Cierre:Nil:Nil)		
<i>Sistema</i> ₇ :	Gracias por utilizar el sistema de información de trenes.		

Figura 6.4: Ejemplo de diálogo generado con el gestor estadístico y la función distancia

En el tercer turno tiene lugar una situación que no está presente en el corpus procesado. De este modo, se utilizan los mecanismos diseñados para combatir la falta de cobertura y poder seleccionar el par (DR, S) más apropiado. En la Figura 6.4, el DR más cercano en el modelo se representa mediante (DR') . La diferencia entre el DR y el DR' es que en el DR' se ha descartado el valor correspondiente al atributo *TIPO-TREN*, preguntando el gestor por la *FECHA-SALIDA*, ignorando el tipo de tren suministrado por el usuario (*Euromed*). Cabe destacar que esta simplificación es aplicable únicamente al turno actual, manteniéndose la información sobre el atributo *TIPO-TREN* en el DR . De esta forma, con el cálculo de DR' se consigue dar una respuesta lo más apropiada posible para la situación no existente en el corpus, respondiéndose al usuario mediante una de las respuestas concordante con el contenido del registro DR' , pero teniéndose en cuenta el registro DR para continuar el diálogo.

6.4.2. Problema de clasificación

Como alternativa a la función distancia para el tratamiento de las situaciones no vistas, se han realizado diferentes trabajos que tratan la selección de la mejor respuesta del sistema como un problema de clasificación. Tal y como se ha comentado previamente, este planteamiento se basa en la realización de una nueva partición en el espacio de estados de diálogo posibles (pares (DR, S)). Cada clase agrupa todos los estados que proporcionan el mismo conjunto de respuestas del sistema. Una vez la fase de entrenamiento ha finalizado, se obtiene un conjunto de clases \mathcal{C} , de forma que cada clase tiene asociada una respuesta diferente del sistema. Durante el transcurso de un nuevo diálogo, cuando tiene lugar una situación no vista, es decir, cuando el par (DR, S) no haya sido observado en la fase de

entrenamiento, el gestor utilizará una función de clasificación para asociar a la entrada una clase, y por tanto, una respuesta.

Para la tesis se han evaluado cuatro aproximaciones para la definición de la función de clasificación, basadas en el aprendizaje de un modelo para cada una de las clases $c \in \mathcal{C}$. Estas aproximaciones son un clasificador naive Bayes multinomial, clasificadores basados en n-gramas, un clasificador basado en la técnica de inferencia gramatical MGGI y un clasificador basado en redes neuronales.

Sea (DR, S) un par no visto durante el aprendizaje, y x la cadena que codifica este par. La clasificación se realiza mediante la siguiente maximización:

$$\begin{aligned}\hat{c} &= \operatorname{argmax}_{c \in \mathcal{C}} P(c|x) \\ &= \operatorname{argmax}_{c \in \mathcal{C}} \frac{P(c)P(x|c)}{P(x)} \\ &= \operatorname{argmax}_{c \in \mathcal{C}} P(c)P(x|c)\end{aligned}\tag{6.4}$$

6.4.2.1. Clasificador Multinomial naive Bayes

Los clasificadores naive Bayes se han utilizado frecuentemente en tareas de recuperación de información, clasificación de textos y aprendizaje automático.

Un clasificador naive Bayes es un clasificador probabilístico sencillo basado en la aplicación del Teorema de Bayes con la hipótesis de que todos los atributos considerados para las muestras son independientes unos de otros en el contexto de las clases definidas (hipótesis naive Bayes). Aunque esta hipótesis no se cumple en un gran número de tareas prácticas, el clasificador naive Bayes realiza a menudo la clasificación de forma correcta. Gracias a la hipótesis de independencia, los parámetros para cada atributo se pueden aprender de forma separada, lo que simplifica considerablemente el aprendizaje, especialmente cuando el número de atributos es grande. De este modo, una de las ventajas de este tipo del clasificador es que requiere una cantidad pequeña de datos de entrenamiento para estimar los parámetros, las medias y varianzas de las variables, necesarios para la clasificación. Asumiendo que se trabaja con variables independientes, sólo es necesario determinar las varianzas de las variables para cada clase y no la matriz completa de covarianzas.

El modelo probabilístico de naive Bayes es el siguiente:

Sea C la variable a clasificar, definiéndose un conjunto de clases $c \in C$ para el clasificador. Sean $F_1 \dots F_n$ un conjunto de características a partir de las cuales se condiciona la clase de pertenencia. Utilizando el Teorema de Bayes:

$$P(C|F_1, \dots, F_n) = \frac{P(C)P(F_1, \dots, F_n|C)}{P(F_1, \dots, F_n)}\tag{6.5}$$

El numerador de la ecuación anterior coincide con la probabilidad de la unión:

$$\begin{aligned}P(C)P(F_1, \dots, F_n|C) &= P(C, F_1, \dots, F_n) \\ &= P(C)P(F_1|C)P(F_2|C, F_1)P(F_3|C, F_1, F_2)P(F_4, \dots, F_n|C, F_1, F_2, F_3)\end{aligned}$$

Asumiendo la independencia de cada una de las características con respecto al resto, la probabilidad conjunta puede expresarse de la forma siguiente:

$$\begin{aligned} P(C, F_1, \dots, F_n) &= P(C)P(F_1|C)P(F_2|C)P(F_3|C)\dots \\ &= P(C) \prod_{i=1}^n P(F_i|C) \end{aligned}$$

Finalmente, la Ecuación 6.5 puede escribirse del siguiente modo:

$$P(C|F_1, \dots, F_n) = \frac{1}{Z} P(C) \prod_{i=1}^n P(F_i|C) \quad (6.6)$$

Los parámetros del modelo (probabilidades de clase y distribuciones de probabilidad de las características) pueden aproximarse por las frecuencias relativas a partir de los datos de un corpus de entrenamiento, utilizando máxima verosimilitud.

Un clasificador estadístico para C clases en un espacio E es un conjunto

$$g_c : E \rightarrow \mathbb{R}$$

La regla de clasificación o decisión asociada a g consiste en asignar cada $x \in E$ a

$$c_g(x) = \operatorname{argmax}_{c \in C} g_c(x)$$

La aplicación de esta regla produce una partición de E en C regiones de decisión:

$$R_c = \{x \in E : g_c(x) \text{ para todo } c' \neq c\}, \quad c = 1, \dots, C$$

El clasificador de Bayes utiliza como criterio de clasificación la máxima probabilidad a posteriori:

$$g_c(x) = p(c|x) = \frac{p(x,c)}{p(x)} \equiv p(x,c) \equiv p(c)p(x|c) \equiv \log p(c) + \log p(x|c)$$

$$c_B(x) = \operatorname{argmax}_{c \in C} g_c(x)$$

Existen dos modelos de probabilidad principales a partir de los cuales suelen definirse clasificadores de Naive Bayes: el modelo de Bernoulli y el multinomial. Una variable aleatoria X sigue una distribución de Bernoulli de parámetro p , $X \sim B_e(p)$, si su función de probabilidad es:

$$p(x) = p^x q^{1-x} \text{ donde } p \in [0, 1] \text{ y } q = 1 - p$$

Se entiende como clasificador de Bernoulli al de Bayes con funciones de densidad de probabilidad condicionales de Bernoulli:

$$p(x|c) \sim Be_D(p_c) \quad c = 1, \dots, C$$

Se trata de un clasificador lineal:

$$\begin{aligned}
 c^*(x) &= \operatorname{argmax}_{c \in C} p(c|x) \\
 &= \operatorname{argmax}_{c \in C} \sum_d (\log p_{cd} - \log(1 - p_{cd}))x_d + (\log p(c) + \sum_d \log(1 - p_{cd})) \\
 &= \operatorname{argmax}_{c \in C} g_c(x)
 \end{aligned}$$

con: $g_c(x) = w_c^t x + w_{c_0}$

donde $w_c = \log p_c - \log(1 - p_c)$ y $w_{c_0} = \log p(c) + \sum_d \log(1 - p_{cd})$

Sea una población de D tipos de elementos en proporciones dadas por:

$$p = (p_1, \dots, p_D) \in [0, 1]^D, \quad \sum_d p_d = 1.$$

Sea X_d el número de elementos de tipo d obtenidos tras seleccionar x_+ elementos al azar y con reemplazamiento. Decimos que $X = (X_1, \dots, X_D)^t$ es una variable aleatoria multinomial de parámetros x_+ y p , $X \sim Mult(x_+, p)$, con función de probabilidad:

$$p(x) \frac{x_+!}{x_1! \dots x_D!} p_1^{x_1} \dots p_D^{x_D}$$

Se entiende como clasificador multinomial al de Bayes con funciones de densidad de probabilidad condicionales multinomiales:

$$p(x|c) \sim Mult_D(x_+, p_c) \quad c = 1, \dots, C$$

Se trata de un clasificador lineal:

$$\begin{aligned}
 c^*(x) &= \operatorname{argmax}_{c \in C} p(c|x) \\
 &= \operatorname{argmax}_{c \in C} \log p(c) + \sum_d x_d \log p_{cd} \\
 &= \operatorname{argmax}_{c \in C} g_c(x)
 \end{aligned}$$

con: $g_c(x) = w_c^t x + w_{c_0}$

donde $w_c = \log p_c$ y $w_{c_0} = \log p(c)$

En (McCallum y Nigam, 1998) se exponen y aplican los modelos de Bernoulli y multinomial para desarrollar un clasificador de Naive Bayes en el ámbito de la clasificación de texto.

Para el proyecto DIHANA, hemos utilizado el clasificador naive Bayes en su modelo de eventos multinomial, estimando sus parámetros a partir de un conjunto de muestras etiquetadas. La clasificación de nuevas muestras se lleva a cabo aplicando la regla de decisión de Bayes, seleccionando la clase que tenga asociada una mayor probabilidad.

La variable x que aparece en la Ecuación 6.4 se compone de los siguientes términos:

- Los dos primeros niveles del etiquetado de la última respuesta dada por el sistema (A_{i-1}): Esta información se modela mediante una variable multinomial, que posee tantos bits como posibles combinaciones de estos dos niveles (51).

$$\vec{x}_1 = (x_{11}, x_{12}, x_{13}, \dots, x_{151}) \in \{0, 1\}^{51}$$

- Registro del diálogo (DR): El DR almacena un total de quince características (5 conceptos y 10 atributos). Cada una de estas características pueden tomar los valores $\{0, 1, 2\}$. De este modo, cada campo del DR puede modelarse utilizando una variable multinomial con tres bits.

$$\vec{x}_i = (x_{i1}, x_{i2}, x_{i3}) \in \{0, 1\}^3 \quad i = 2, \dots, 16$$

- Información independiente de la tarea (actos de diálogo *Afirmación*, *Negación*, y *No-Entendido*): Estos tres actos de diálogo se han codificado de forma idéntica a las características almacenadas en el DR . De esta forma, cada uno de estos tres actos de diálogo puede tomar los valores $\{0, 1, 2\}$ y modelarse utilizando una variable multinomial con tres bits.

$$\vec{x}_i = (x_{i1}, x_{i2}, x_{i3}) \in \{0, 1\}^3 \quad i = 17, \dots, 19$$

De este modo, la variable x puede representarse mediante el vector de características:

$$\vec{x} = (\vec{x}_1, \vec{x}_2, \vec{x}_3, \dots, \vec{x}_{19})$$

y su probabilidad viene dada por la expresión:

$$P(\vec{x}) = P(\vec{x}_1)P(\vec{x}_2)P(\vec{x}_3) \cdots P(\vec{x}_{19})$$

Las probabilidades de las variables multinomiales se calculan mediante:

$$P(\vec{x}_1) = \prod_{d=1}^{51} p_{1,d}^{x_{1,d}}$$

$$P(\vec{x}_i) = \prod_{d=1}^3 p_{i,d} \quad i = 2, \dots, 19^{x_{i,d}}$$

donde los coeficientes siguen la expresión:

$$p_{i,d} = \frac{N(x_{i,d} = 1)}{N}$$

siendo N es el número de muestras de la clase, y $N(x_{i,d} = 1)$ es el número de muestras de dicha clase con un valor 1 en la posición d .

6.4.2.2. Clasificadores basados en n-gramas y clasificador MGGI

En estos clasificadores, se estima un autómata de estados finitos probabilístico para cada una de las clases $c \in \mathcal{C}$, a partir de sus correspondientes muestras de entrenamiento. La variable x contiene la información mostrada para el clasificador multinomial. Cuando sucede una situación no vista durante el transcurso de un diálogo, la próxima respuesta del sistema se selecciona aplicando la Ecuación 6.4. La muestra se analiza por cada uno de los autómatas aprendidos, seleccionándose la respuesta asociada al autómata que proporciona la mayor probabilidad.

Para este trabajo, hemos utilizado tres tipos de clasificadores de estados finitos: modelos de bigramas, modelos de trigramas y modelos MGGI (García et al., 1990) (Segarra y Hurtado, 1997). La metodología MGGI (Morphic Generator Grammatical Inference) es una técnica de inferencia gramatical que nos permite obtener una variedad de lenguajes regulares a partir de un conjunto de muestras positivas del lenguaje a modelar. Básicamente la metodología MGGI se basa en las siguientes características:

- La definición de una función de etiquetado g , es decir, cada símbolo de cada muestra de entrada se especializa (etiqueta) de acuerdo con la función de etiquetado. Diferentes definiciones de g generarán diferentes modelos.
- Se aprende un lenguaje regular 2-Explorable en Sentido Estricto utilizando las muestras etiquetadas. El resultado puede representarse como un autómata finito determinista. El alfabeto de este modelo es el alfabeto etiquetado, no el original.
- Aplicando un homomorfismo h (generalmente el inverso de g) se procede a desetiquetar el modelo aprendido en el paso anterior. La idea es simple: se etiquetan las frases de entrenamiento utilizando la función g , se aprende un modelo de 2-Explorables en Sentido Estricto y se desetiqueta utilizando el homomorfismo h .

El lenguaje regular L generado a partir de un conjunto de muestras R_+ cumple que:

$$L = h(l(g(R_+)))$$

donde $l(g(R_+))$ es el lenguaje generado aplicando el algoritmo para los 2-Explorables en Sentido estricto con R_+ .

El MGGI crea un autómata no determinista que admite cadenas sobre el alfabeto original. El automata generado es más restrictivo que el generado por el 2-EE equivalente. Los modelos MGGI son modelos más grandes que los 2-EE para las mismas muestras de entrenamiento y el lenguaje que aceptan es un subconjunto del lenguaje aceptado por los 2-EE. En realidad, el lenguaje inferido utilizando la metodología MGGI fluctúa entre el lenguaje 2-Explorable en sentido estricto inferido a partir de las muestras sin etiquetar y el aceptor de prefijos aprendido con esas mismas muestras, dependiendo de la función g de etiquetado utilizada.

Diferentes funciones de etiquetado (g) generan distintos modelos de lenguaje. Básicamente, en MGGI se utilizan dos tipos de etiquetados: los que etiquetan de forma automática las palabras dependiendo de su posición en la frase, y los que etiquetan las palabras de la frase dependiendo de su función. A diferencia del etiquetado posicional, el etiquetado sintáctico necesita generalmente de un experto. El MGGI intenta, utilizando la función

etiquetado, especializar el modelo de bigramas. Puede consultarse información sobre la aplicación de modelos basados en MGGI en diferentes procesos dentro del proyecto DIHANA en (Justo, Torres, y Hurtado, 2006) (Grau et al., 2006).

En el ámbito de la gestión de diálogo en DIHANA, definimos la función de etiquetado g de tal manera que especializa cada campo del DR agregando información sobre su posición. La Figura 6.5 muestra un ejemplo de la utilización de la función g definida. Esta función se aplica a los símbolos procedentes de la representación del DR , pero no a aquellos pertenecientes a S .

$$g(121001200000100) = \\ 1\#1\ 2\#2\ 1\#3\ 0\#4\ 0\#5\ 1\#6\ 2\#7\ 0\#8\ 0\#9\ 0\#10\ 0\#11\ 0\#12\ 1\#13\ 0\#14\ 0\#15$$

Figura 6.5: Ejemplo de la función g definida en el MGGI

6.4.2.3. Clasificador mediante redes neuronales. El perceptrón multicapa

Una red neuronal es un procesador de información constituido por elementos de procesamiento simples conectados entre sí. Cada uno de esos elementos es un modelo simple de una neurona cerebral. Una neurona puede definirse como una unidad de proceso dotada de una serie de entradas y de una única salida. La neurona produce una combinación lineal de las entradas y utiliza una función de activación para activar su salida con un valor que depende de esta combinación de un modo en general no lineal.

Las Redes Neuronales Artificiales (RNA) son sistemas de procesamiento de la información cuya estructura y funcionamiento están inspirados en las redes neuronales biológicas. Consisten en un gran número neuronas organizadas de forma clásica en capas. Cada neurona está conectada con otras neuronas mediante enlaces de comunicación, cada uno de los cuales tiene asociado un peso. En los pesos se encuentra el conocimiento que tiene la RNA acerca de un determinado problema.

La estructura de una RNA está conformada por las entradas (valores que alimentan a la red neuronal), salidas (clases reconocidas con la red neuronal), capa de entrada (conjunto de neuronas que recogen directamente los valores de la entrada de la red), capa de salida (conjunto de neuronas que proporcionan los valores de salida de la red) y capas ocultas (capas de la red neuronal que procesan la información de tal manera que permiten una adecuada separación de las clases que se pretenden reconocer). La Figura 6.6 muestra la estructura descrita.

La característica que hace a las redes neuronales tan útiles para numerosas aplicaciones es su capacidad de mejorar con el tiempo conforme adquiere más conocimiento sobre el problema, su capacidad de generalizar y de manejar información difusa del mundo real. La red neuronal no se programa para realizar una cierta tarea, sino más bien se adapta y adquiere conocimiento mientras trabaja para completar una tarea específica.

El caso más sencillo de RNA es el que presenta una sola neurona de cómputo. A esta estructura se le denomina perceptrón. El comportamiento del perceptrón se basa en realizar las siguientes operaciones:

- Un producto escalar del vector de entradas, \vec{x} , por otro vector de pesos, \vec{w} . Además

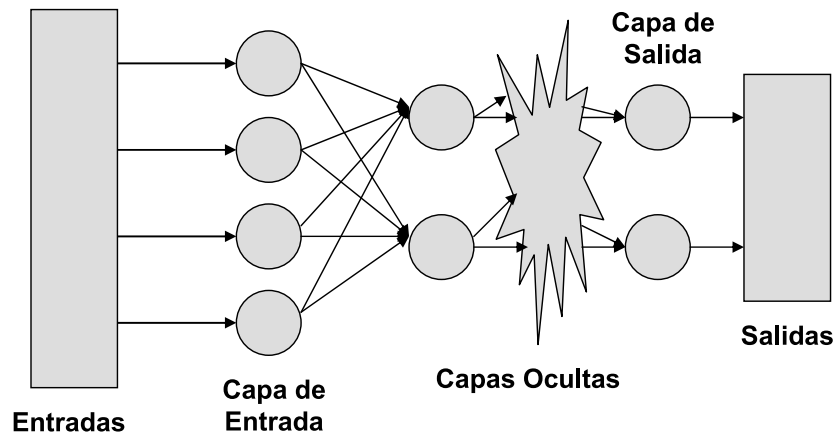


Figura 6.6: Red neuronal artificial

le resta un valor que se denomina umbral θ . Al resultado de esta operación se le denomina potencial: $\gamma = \vec{x} \cdot \vec{w} - \theta$

- Al resultado anterior le aplica una función de activación f , que suele ser no lineal: $f(\gamma) = z$. La Tabla 6.2 muestra diferentes ejemplos de funciones de activación comúnmente utilizadas.
- El resultado y es propagado por la salida de la neurona.

Función Lineal $f(x) = x$	Función Sigmoide $s_c(x) = \frac{1}{1+e^{-cx}}$
Tangente Hiperbólica $f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	Función Softmax $f(x_k) = e^{x_k} / \sum_{k'=1}^n e^{x_{k'}}$

Tabla 6.2: Ejemplos de funciones de activación de una red neuronal

El perceptrón multicapa es una RNA formada por múltiples neuronas simples (perceptrones) distribuidas en capas, de la forma indicada en la Figura 6.6. Se trata de RNA de propósito general y flexibles, cuya complejidad puede modificarse variando el número de capas y el número de unidades por cada capa. El perceptrón multicapa (MLP) puede estar totalmente o localmente conectado. En el primer caso, cada salida de una neurona de la capa i es entrada de todas las neuronas de la capa $i + 1$, mientras que el segundo, cada neurona de la capa i es entrada de una serie de neuronas (región) de la capa $i + 1$. El perceptrón computa los valores de su capa de salida calculando una combinación lineal de acuerdo con los pesos definidos para sus entradas y la aplicación de una función de activación no lineal al resultado obtenido. Matemáticamente este proceso puede expresarse para el perceptrón del siguiente modo:

$$y = \varphi\left(\sum_{i=1}^n \omega_i x_i + b\right) = \varphi(w^T x + b)$$

donde w denota al vector de pesos, x es el vector de entradas, b es un valor de sesgo (*bias*) y φ es la función de activación.

La Figura 6.7 muestra la arquitectura descrita para el perceptrón y el MLP. El perceptrón puede ajustarse modificando los valores de los pesos de sus conexiones. Para realizar este ajuste existen diversas técnicas, siendo las más comunes las basadas de descenso por gradiente. Un segundo tipo de técnicas son los métodos de segundo orden, que suelen ser variaciones de los métodos de descenso por gradiente.

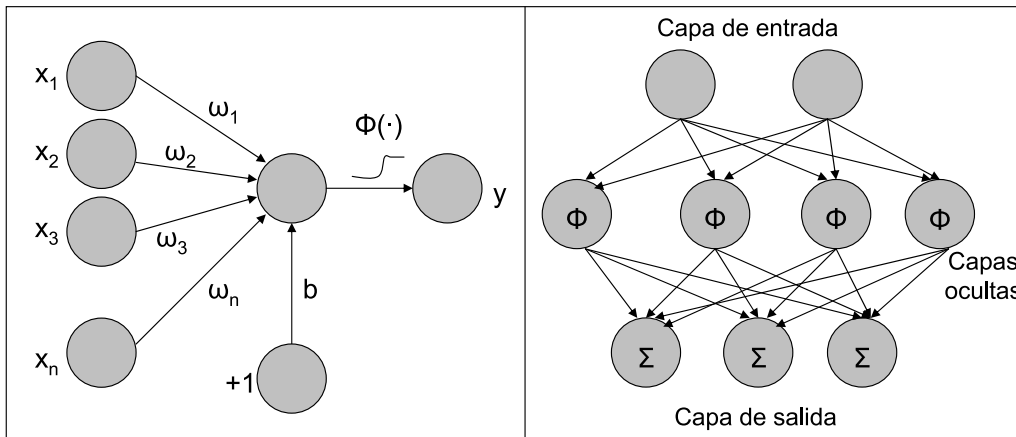


Figura 6.7: Esquema gráfico de un perceptrón (izquierda) y de un MLP (derecha)

Entre los métodos basados en descenso por gradiente, el más conocido es el Algoritmo de Retropropagación del Error o *Backpropagation* (BP) (Rumelhart, Hinton, y Williams, 1986) y todas sus variantes. Estos métodos consisten en definir una función error y calcular su gradiente respecto a los parámetros o pesos de la red. De esta forma, los pesos son modificados en la dirección del gradiente, buscando un mínimo en la función error. Se utiliza la siguiente técnica: dado un patrón de entrada, compara el resultado obtenido en las unidades de salida con la respuesta que se desea obtener. A continuación, reajusta los pesos de la red de manera que la siguiente vez que se presente el mismo patrón de entrada, la red produzca un resultado más cercano al deseado, es decir, que el error disminuya. A este método también se le denomina *regla delta generalizada*.

La regla delta generalizada fue creada para generalizar la regla delta sobre redes neuronales de múltiples capas y funciones de transferencia no lineales y diferenciables. Sus principales características son: entrenamiento supervisado por corrección de error, aprendizaje off line y capacidad de generalización. La regla delta puede expresarse de la siguiente forma:

$$\Delta w_{ji} = \alpha \cdot y_j \cdot (d_i - y_i)$$

donde:

d_i : valor de salida deseado para la neurona u_i .

α : factor de aprendizaje (regula la velocidad de aprendizaje).

$\Delta w_{ji} = w_{ji}^{actual} - w_{ji}^{anterior}$: modificación del peso w_{ji} .

$d_i - y_i = \delta_i$: error que se produce en la neurona u_i .

Regla delta generalizada, $\delta_i = (d_i - y_i) \cdot f'(Net_i)$

representando Net_i el valor conjunto de todas las señales que le llegan a la neurona u_i , $Net_i = \sum_j y_j \cdot w_{ji}$, y siendo $f(Net_i)$ la función de salida o transferencia de la neurona.

La expresión de la regla delta generalizada coincide con la de la regla delta, excepto en el término de la derivada de la función de salida. Este término representa la modificación que hay que realizar en la entrada que recibe la neurona u_i . En el caso de que u_i no sea una unidad de salida, el error que se produce está en función del error que se comete en las unidades que reciben como entrada la salida de u_i , de ahí el nombre de retropropagación. En el caso de que una neurona pertenezca a la capa de salida, se emplea:

$$\delta_i = f'(Net_i) \cdot (d_i - y_i)$$

En el caso de que una neurona no pertenezca a la capa de salida, se emplea:

$$\delta_i = f'(Net_i) \sum_k w_{ik} \cdot \delta_k$$

La figura 6.8 muestra un esquema gráfico de los términos definidos en la regla delta y en la regla delta generalizada.

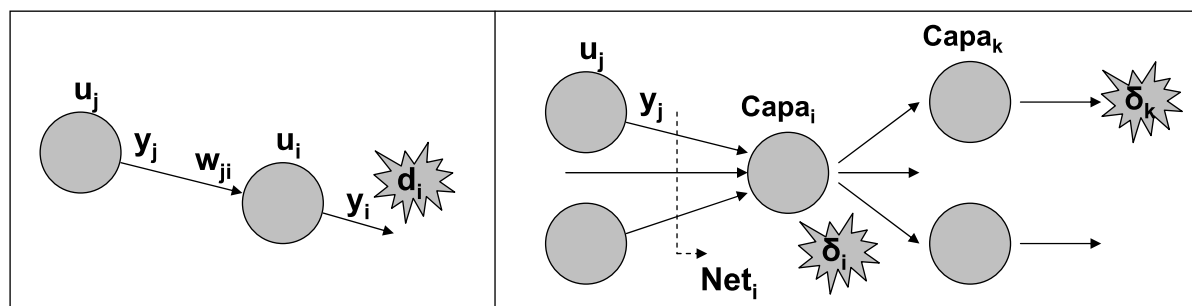


Figura 6.8: Esquema gráfico de la regla delta (izquierda) y de los términos involucrados en la regla delta generalizada (derecha)

En resumen, el algoritmo de aprendizaje BP se basa en la obtención de errores empezando en la capa de salida y retrocediendo sucesivamente hasta la capa de entrada. Una vez obtenido el error δ de una neurona, se puede calcular la variación que hay que ir asignando a los pesos w de entrada a dicha neurona. Este proceso se repite con sucesivos pares [*patrón de entrada, valor de salida esperado*], actuando sobre todas las neuronas, hasta que se considere que la red ha aprendido y proporciona valores de salida adecuados.

Establecida la forma de calcular los errores, la variación de los pesos que exige el aprendizaje se realiza aplicando la expresión:

$$\Delta w_{ji} = \alpha \cdot \delta_i \cdot y_j$$

La Figura 6.9 resume el algoritmo de aprendizaje BP. Entre las variantes de este método cabe destacar:

ALGORITMO DE BACKPROPAGATION

1. Inicializar los pesos y los umbrales iniciales de cada neurona. Hay varias posibilidades de inicialización siendo las mas comunes las que introducen valores aleatorios pequeños.
2. Para cada patrón del conjunto de los datos de entrenamiento:
 - a) Obtener la respuesta de la red ante ese patrón. Esta parte se consigue propagando la entrada hacia adelante. Las salidas de una capa sirven como entrada a las neuronas de la capa siguiente, procesándolas de acuerdo a la regla de propagación y la función de activación correspondientes.
 - b) Calcular los errores que se producen en la capa de salida, empleando $\delta_i = f'(Net_i) \cdot (d_i - y_i)$
 - c) Calcular los errores pertenecientes a la capa anterior, empleando $\delta_i = f'(Net_i) \sum_k w_{ik} \cdot \delta_k$
3. Volver al paso c) hasta que se alcance la capa de entrada.
4. Actualizar pesos y umbrales, empleando $\Delta w_{ji} = \alpha \cdot \delta_i \cdot y_j$
5. Calcular el error actual y volver al paso 2 si no es satisfactorio (iterar hasta que el error sea menor que un umbral establecido o hasta que se haya alcanzado un número fijado de iteraciones).

Figura 6.9: Algoritmo de Backpropagation

- BP “batch”: se trata del algoritmo clásico. El error de la red es la media de los errores producidos por cada muestra de entrenamiento. La actualización de los pesos se lleva a cabo cuando ya se han visitado todas las muestras.
- BP “on-line”: Difiere del anterior en el momento en que se actualizan los pesos, que en este caso se realiza con cada muestra de entrenamiento, calculando el error únicamente para esa muestra.
- BP con momentum: Puede ser tanto “batch” como “on-line”. Añade al BP clásico la definición de un factor, llamado momentum (μ), que pretende suavizar el entrenamiento introduciendo en el cálculo del incremento de los pesos un porcentaje del incremento que se produjo en la iteración anterior.

Entre los métodos de segundo orden, cabe destacar diferentes variantes del BP que tratan de acelerar la velocidad de convergencia:

- Algoritmo *Quickprop*: Sigue los mismos pasos que el BP, pero para calcular el incremento de los pesos ajusta una función cuadrática mediante la derivada de la función de error en la iteración anterior y la actual. De este modo, modifica los pesos buscando el mínimo de la función cuadrática, lo que origina un descenso mucho más rápido hacia el mínimo de la función de error.

- Algoritmo de *Newton*: Utiliza una matriz, denominada *matriz de Hessian*, que contiene la derivada segunda de la función de error respecto a cada uno de los pesos. A partir de la inversa de esta matriz puede calcularse la dirección exacta del mínimo de la función error.
- Algoritmos *quasi-Newton*: Estos métodos aceleran la velocidad del método anterior aproximando el cálculo de la inversa para que no sea tan costoso, introduciendo así un factor de imprecisión en los cálculos.

Tal y como se ha comentado previamente, los perceptrones multicapa (MLPs) son las redes neuronales artificiales más comúnmente utilizadas en tareas de clasificación. Para la aplicación de un MLP en la búsqueda de la respuesta del gestor del diálogo en nuestra tarea, la capa de entrada recibe la codificación del par de entrada (DR_{i-1}, S_{i-1}) , y la capa de salida se define de acuerdo al número de posibles respuestas del sistema (C) y representa la probabilidad de pertenencia de la muestra (x) a cada una de las clases $(p(c|x), c \in C)$. El resultado de esta clasificación nos da la correspondiente respuesta del sistema A_i asociada a dicha clase.

La representación definida para codificar el par de entrada (DR_{i-1}, S_{i-1}) coincide con la mostrada para el clasificador naive Bayes.

6.4.3. Implementación práctica del modelo

A la hora de implementar un gestor de diálogo que lleve a la práctica el modelo descrito, se ha realizado una división del modelo estadístico en dos procesos diferenciados.

Un primer proceso se encarga de realizar la actualización del registro del diálogo a partir de la información suministrada por el módulo de comprensión tras la intervención del usuario. Este proceso se realiza de forma automática, incorporándose al registro toda aquella información relevante para la tarea que proporcione comprensión y variando las confianzas almacenadas junto a los valores de los conceptos y atributos una vez el usuario los confirme positivamente. Este último caso implica un cambio en el registro de diálogo distinto a completar su contenido. Por ello, se separan de la representación semántica del turno de usuario los actos de diálogo independientes de la tarea *Afirmación*, *Negación* y *No-Entendido*.

Un segundo proceso se encarga de obtener los posibles etiquetados de diálogo concordantes con el estado actual del registro del diálogo (una vez se ha actualizado con la información suministrada por el usuario en el turno anterior), los conceptos independientes de la tarea y los dos primeros niveles de etiquetado de la última respuesta generada por el sistema.

La Figura 6.10 muestra el funcionamiento práctico del gestor desarrollado.

El gestor opera de acuerdo con el siguiente algoritmo:

- Recibe la representación semántica de la entrada del usuario generada por el módulo de comprensión.
 - Actualiza el registro del diálogo de acuerdo con la información anterior.
 - Realiza la codificación de la información suministrada por comprensión y del registro de diálogo, siguiendo el formato y orden definido para cada una de las estructuras.
-

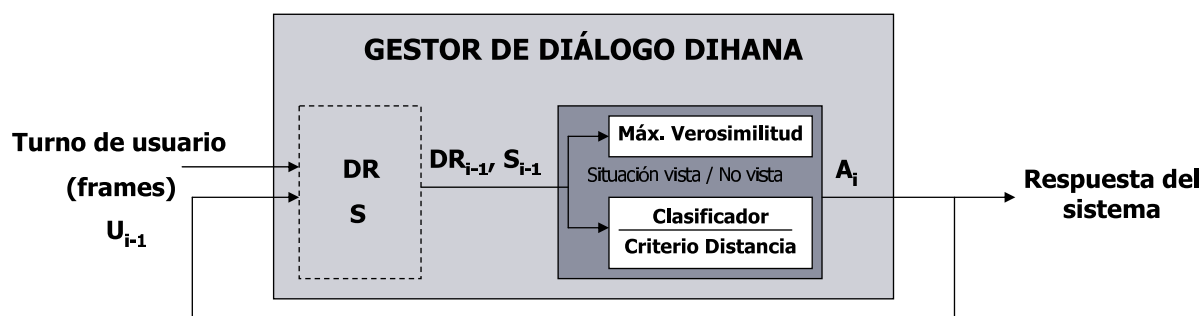


Figura 6.10: Gestor estadístico desarrollado con la aplicación de las técnicas de tratamiento de las situaciones no vistas

- Genera la secuencia resultante de unir los dos primeros niveles de etiquetado de la última respuesta del sistema, el contenido del registro de diálogo de acuerdo con la codificación definida, y la información independiente de la tarea aportada por el usuario en su última intervención.
- A partir de esta secuencia acude al fichero del corpus y selecciona uno de los posibles etiquetados de diálogo concordantes con dicha codificación.

En caso de no detectar ninguna correspondencia en el corpus para la salida generada por el gestor, se aplican los métodos descritos para el tratamiento del problema de la cobertura del modelo. Para evitar la aparición de bucles en las respuestas generadas por el gestor ante los casos de situaciones no vistas en el modelo, se utiliza un registro histórico de simplificaciones, en el que se almacenan las respuestas generadas por el gestor ante las diferentes situaciones no presentes en el corpus. De este modo, el gestor selecciona la secuencia de menor distancia siguiente a la última utilizada por el gestor como sustituto de la secuencia no presente en el corpus (criterio de la distancia) o aquella que sigue en probabilidad a la seleccionada por el clasificador en último lugar para dicha secuencia (criterio de clasificación).

Además, el gestor posee un componente adicional: el registro histórico de respuestas. Este registro almacena el conjunto de respuestas generadas por el gestor. Esta información es fundamental para actualizar el nivel de confianza asociado a la información tras una confirmación positiva del usuario, para lo cual es necesario disponer de la última respuesta suministrada por el sistema, es decir, de los atributos y/o conceptos que ha confirmado el usuario.

La Figura 6.11 muestra un ejemplo de diálogo adquirido siguiendo la metodología descrita. En el primer turno de diálogo puede observarse el etiquetado correspondiente al turno de bienvenida del sistema (*Apertura:Nil:Nil*) y el *DR* inicial definido para la tarea DIHANA, con el origen y la clase billete como información por defecto.

Tras cada intervención del usuario, se actualiza el *DR* con la información aportada en el mismo. Tal y como se ha comentado previamente, el gestor selecciona la nueva respuesta del sistema teniendo en cuenta la última respuesta del sistema (A_{i-1}), el registro del diálogo (DR_{i-1}) y los actos de diálogo *Afirmación*, *Negación* y *No-Entendido* que existan en la intervención del usuario (U_{i-1}).

S1: Bienvenido al sistema de información de trenes. ¿En qué puedo ayudarle? $A_1=(\text{Apertura:Nil:Nil})$ $DR_0=00000-1000001000$
U1: Quiero ir a Barcelona. $A_1=\text{Apertura:Nil}$ $U'_1=()$ $DR_1=00000-1100001000$ $A_2=(\text{Confirmación:Hora-Salida:Nil})$ S2: ¿Quiere saber los horarios?
U2: Sí, para el tren Euromed. $A_2=(\text{Confirmación:Hora-Salida})$ $U'_2=(\text{Afirmación})$ $DR_2=10000-1100001100$ $A_3=(\text{Pregunta:Fecha-Salida:Nil})$ S3: Dígame la fecha de salida.
U3: Mañana. $A_3=\text{Pregunta:Fecha-Salida}$ $U'_3=()$ $DR_3=10000-1120001100$ $A_4=(\text{Confirmación:Fecha-Salida:Fecha-Salida})$ S4: ¿Quiere salir mañana?
U4: Sí $A_4=(\text{Confirmación:Fecha-Salida})$ $U'_4=(\text{Afirmación})$ $DR_4=10000-1110001100$ $A_5=(\text{Respuesta:Fecha-Salida:Número-Trenes})(\text{Nueva-Consulta:Nil:Nil})$ S5: Hay varios trenes Euromed. El primero sale a 08:54 y el último a 23:45. ¿Desea algo más?
U5: No, gracias. $A_5=\text{Respuesta,Nueva-Consulta:Fecha-Salida,Nil}$ $U'_5=(\text{Negación})$ $DR_5=10000-1110001100$ $A_6=(\text{Cierre:Nil:Nil})$ S6: Gracias por utilizar el sistema de información de trenes. Adiós.

Figura 6.11: Ejemplo de diálogo obtenido con el gestor basado en el clasificador Naive Bayes

En el segundo turno del sistema se produce un par (DR, S) no visto durante el aprendizaje. La Figura 6.11 muestra la respuesta seleccionada por el gestor tras clasificar el par utilizando el clasificador bayesiano descrito en este capítulo.

6.5. Aproximación estadística mediante redes neuronales

El segundo gestor desarrollado se basa como principal idea en el buen funcionamiento del clasificador basado en redes neuronales a la hora de obtener una respuesta adecuada dada la situación del diálogo, ya sea una situación vista durante la fase de entrenamiento o se trate de una situación no vista. En ambos casos, puede asignarse la respuesta del sistema de acuerdo con el proceso de clasificación. De este modo, se utiliza un perceptrón multicapa para clasificar la variable definida para codificar el estado actual del diálogo y determinar así la clase (respuesta del sistema) más probable para la situación actual del diálogo, independientemente de que se trate de una situación vista o no vista.

Resumiendo esta aproximación, el espacio particionado de las secuencias posibles de actos del diálogo estimado durante la fase del entrenamiento se particiona una segunda vez en clases. Cada nueva clase agrupa todos los estados de diálogo que proporcionan

el mismo conjunto de acciones de sistema (respuestas). Después del entrenamiento se define un conjunto de clases \mathcal{C} . Durante el diálogo, tras cada intervención del usuario, el (DR_{i-1}, S_{i-1}) se clasifica en una clase de este conjunto $c \in \mathcal{C}$, y la respuesta del sistema en ese instante es la respuesta asociada a esta clase seleccionada.

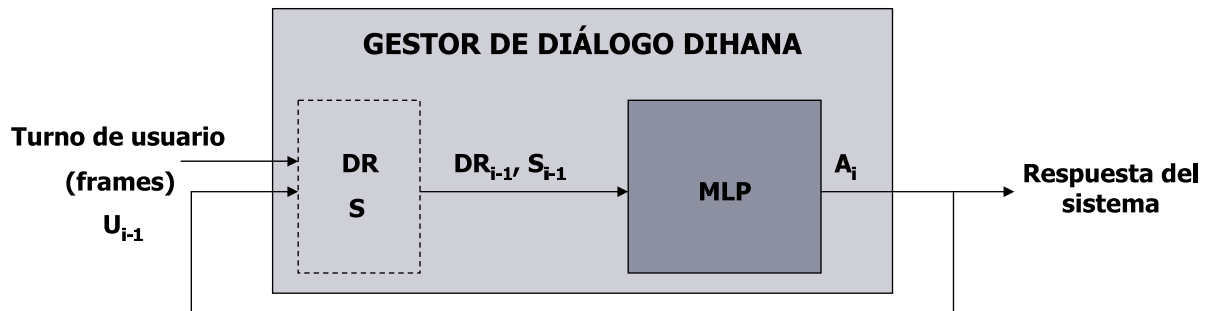


Figura 6.12: Esquema del gestor de diálogo desarrollado para el proyecto DIHANA

La Figura 6.12 muestra el funcionamiento práctico del gestor de diálogo desarrollado para DIHANA. Los frames generados por el módulo de comprensión tras cada intervención del usuario y la última respuesta proporcionada por el sistema se utilizan para generar el par (DR_{i-1}, S_{i-1}) . La codificación de este par constituye la entrada del perceptrón multi-capa que proporciona la probabilidad de seleccionar cada una de las respuestas definidas en DIHANA, dada la situación del diálogo representada por este par.

Seguidamente se muestra un ejemplo de un diálogo adquirido utilizando el gestor de diálogo descrito en esta sección. Las siglas TDI y TII hacen referencia respectivamente a la información dependiente e independiente de la tarea proporcionada por el módulo de comprensión. Se ha fijado un umbral de 0,5 para determinar la fiabilidad de la información. Las medidas de confianza proporcionadas por el módulo de comprensión se muestran entre paréntesis.

El diálogo comienza con un turno de bienvenida (S_1). El etiquetado a tres niveles de este turno es (*Apertura:Nil:Nil*). El DR inicial contiene como información por defecto el *origen* y la *clase del billete* (los dos “1” que pueden observarse en DR_0).

S_1 : Bienvenido al sistema de información de trenes. ¿En qué puedo ayudarle?
 A_1 : (*Apertura:Nil:Nil*)
 DR_0 : 00000-1000001000

En el primer turno, el usuario provee el concepto *Hora* y el atributo *Destino*. Esta información se utiliza para actualizar DR_0 y obtener DR_1 . La entrada del MLP se genera utilizando DR_1 , los dos últimos niveles del etiquetado de la última respuesta del sistema (A_1) y la información independiente de la tarea proporcionada por el usuario en el turno actual (ninguna en este caso). La salida seleccionada por el MLP consiste en requerir la fecha de salida.

En el siguiente turno, el usuario proporciona la fecha, a la que se le asigna una baja confianza. De esta forma, el DR_2 se obtiene añadiendo un “2” en la posición de la fecha-

U_1 : **Quisiera saber horarios para ir a Barcelona.**
 TDI: (*Hora*) [0,7] *Destino*:Barcelona [0,9]
 TII: ()
 DR_1 : 10000-1100001000
 10000-1100001000 + *Apertura*:Nil + () $\rightarrow A_2$: (*Pregunta*:Fecha-Salida:Nil)
 S_2 : **Dígame la fecha de salida.**

salida. La entrada del MLP se genera de la forma mencionada anteriormente. Como salida, se selecciona una confirmación de la *fecha-salida*.

U_2 : **Mañana.**
 TDI: *Fecha*:Mañana [0,3]
 TII: ()
 DR_2 : 10000-1120001000
 10000-1120001000 + *Pregunta*:Fecha-Salida + () $\rightarrow A_3$: (*Confirmación*:Fecha-Salida:Fecha-Salida)
 S_3 : **¿Desea salir mañana?**

En el tercer turno, el usuario confirma este valor. El módulo de comprensión proporciona el acto de diálogo *Afirmación*. Como resultado de la clasificación llevada a cabo por el MLP, se selecciona realizar una consulta a la base de datos. Como resultado, el sistema proporciona la información de horarios requerida por el usuario.

U_3 : **Sí.**
 TDI:
 TII: (*Afirmación*) [0,7]
 DR_3 : 10000-1120001000
 10000-1110001000 + *Confirmación*:Fecha-Salida + (*Afirmación*) $\rightarrow A_4$: (*Respuesta*:Hora-Salida:Número-Trenes,Tipo-Tren,Hora-Salida) (*Nueva-Consulta*:Nil:Nil)
 S_4 : **Hay varios trenes. El primero sale a las 08:55 y el último a las 23:45. ¿Desea algo más?**

Finalmente, el usuario menciona que no desea nada más. Como resultado de la clasificación, se selecciona el acto de diálogo *Cierre*.

U_4 : **No, gracias.**
 TDI: ()
 TII: (*Negación*) [0,8]
 DR_4 : 10000-1120001000
 10000-1110001100 + *Respuesta-Nueva-Consulta*:Hora-Salida-Nil + (*Negación*) $\rightarrow A_5$: (*Cierre*:Nil:Nil)
 S_5 : **Gracias por utilizar el sistema de información de trenes. Adiós.**

Por último, la Figura 6.13 muestra dos ejemplos adicionales de diálogos adquiridos con el gestor de diálogo descrito.

S1: Bienvenido al sistema de información de trenes. ¿En qué puedo ayudarle?
U1: Quiero saber horarios de Valencia a Madrid para el uno de julio.
S2: ¿Desea salir el 01/07/2007?
U2: Sí, por la noche.
S3: ¿Quiere salir entre las 21.00 y las 05.00?
U3: Sí.
S4: Le consulto horarios de Valencia a Madrid. No hay trenes que satisfagan los requisitos solicitados. ¿Desea algo más?
U4: ¿Hay trenes por la mañana?
S5: ¿Quiere salir entre las 05.00 y las 13.00?
U5: Sí.
S6: Le consulto horarios de Valencia a Madrid. Hay varios trenes. El primero sale a las 06.45 y llega a las 10.17 y el último sale a las 17.50 y llega a las 20.05. ¿Desea algo más?
U6: Me gustaría ir en Alaris.
S7: Le consulto horarios de valencia a madrid en Alaris. Hay varios trenes. El primero sale a las 06.45 y llega a las 10.17 y el último sale a las 11.20 y llega a las 14.55. ¿Desea algo más?
U7: Dígame el precio del primero.
S8: Le consulto precios de valencia a madrid. El resultado de su consulta es un tren alaris y su precio en clase preferente es 56,5 euros el billete de ida. ¿Desea algo más?
U8: No, gracias.
S9: Gracias por utilizar el sistema de información de trenes. Adiós.

S1: Bienvenido al sistema de información de trenes. ¿En qué puedo ayudarle?
U1: Quisiera ir de Valencia a Barcelona en Euromed.
S2: ¿Desea hacer el viaje en un Euromed?
U2: Sí.
S3: ¿Quiere saber horarios?
U3: Sí.
S4: ¿Qué día desea salir?
U4: Mañana.
S5: Le consulto horarios de trenes con salida en Valencia destino Barcelona saliendo 20/06/2007 en Euromed. Hay varios trenes. El primero sale a las 08.37 y llega a las 11.46 y el último sale a las 21.15 y llega a las 00.45. ¿Desea algo más?
U5: ¿Me dice el precio del primero?
S6: El resultado de su consulta es un tren Euromed y su precio en clase preferente es 47,5 euros el billete de ida. ¿Desea algo más?
U6: ¿Puede decirme la duración del trayecto?
S7: El resultado de su consulta es un tren Euromed con salida 20/06/2007 a las 08.37 y tarda tres horas y nueve minutos. ¿Desea algo más?
U7: No.
S8: Gracias por utilizar el sistema de información de trenes. Adiós.

Figura 6.13: Ejemplos de diálogos obtenidos con el gestor basado en la utilización del MLP

Capítulo 7

Evaluación de los gestores de diálogo desarrollados

7.1. Introducción

En la Sección 2.3 se presentaron las técnicas desarrolladas por diferentes grupos de investigación para la evaluación de los sistemas de diálogo. Aunque existe algún marco formal para la evaluación de los sistemas, como el paradigma PARADISE, su adaptación a la evaluación de un sistema concreto no resulta sencilla.

Se han definido dos tipos fundamentales de evaluación. La primera de ellas se basa en la evaluación del gestor de diálogo mediante usuarios reales y voz. Se ha optado por realizar una evaluación considerando las medidas cuantitativas de uso más extendido: tasas de éxito de los diálogos, duración de los diálogos (número de turnos), tipo de turnos y sus porcentajes, tasas de corrección de errores, etc. Este tipo de evaluación se ha aplicado tanto a los gestores basados en reglas como a los modelados estadísticamente.

La segunda técnica se ha aplicado a la evaluación de la metodología estadística para la gestión del diálogo, estableciendo particiones de entrenamiento y evaluación a partir del corpus DIHANA. Realizando el entrenamiento del gestor con la partición correspondiente, se evalúa turno a turno la respuesta generada por el gestor para cada una de las muestras de la partición de evaluación. Para realizar dicha evaluación, se han definido una serie de medidas por las que se estima la calidad de las respuestas proporcionadas por el gestor: coincidencia con la de referencia indicada en el corpus, coherente con la estrategia definida en DIHANA par al adquisición del corpus, etc.

La Sección 7.2 presenta los resultados obtenidos de la evaluación de los dos gestores de diálogo basados en reglas descritos en el Capítulo 5. Las secciones 7.3 y 7.4 se centran en la evaluación, mediante particiones de entrenamiento y evaluación, de la metodología de gestión estadística descrita en el capítulo anterior. En la Sección 7.3 se muestran los resultados de la evaluación de las dos técnicas propuestas para el tratamiento de las situaciones no vistas durante el entrenamiento: la función distancia definida en el Apartado 6.3.1 y los diferentes clasificadores propuestos en el Apartado 6.3.2. En la Sección 7.4 se evalúa el funcionamiento del gestor estadístico basado en redes neuronales, que se ha descrito en la Sección 6.4. Finalmente, la Sección 7.5 resume los resultados de la evaluación de la metodología de gestión estadística con usuarios reales y voz.

7.2. Evaluación de las metodologías basadas en reglas

Se ha evaluado el comportamiento de los gestores de diálogo presentados en el Capítulo 5 fijando un umbral de confianza (0,5) para valorar la fiabilidad de la información y utilizando un conjunto de 15 escenarios sobre consultas de horarios y precios en viajes de ida o de ida y vuelta. Se llevaron a cabo un total de 120 diálogos, 60 para cada una de las estrategias. En la adquisición participaron 6 usuarios familiarizados con el sistema. Para la evaluación de las estrategias se ha tenido en cuenta las siguientes medidas:

1. Porcentaje de diálogos en los que se alcanzó el objetivo (% éxito). Para cada escenario el usuario debe obtener información correspondiente a una o varias consultas, dependiendo el éxito del diálogo de que el gestor suministre correctamente la totalidad de la información.
2. Número medio de turnos (nT) de sistema por diálogo.
3. Ratio de confirmaciones (%confirm). Este valor se obtiene contando el número de turnos correspondientes a confirmaciones explícitas, nCT, con respecto al total de turnos de sistema, es decir, nCT/nT .
4. Número medio de errores corregidos por diálogo (nCE). Se trata de la media de errores detectados y corregidos por el gestor.
5. Número medio de errores no corregidos por diálogo (nNCE). Se consideran únicamente los errores que modifican los valores de los atributos.
6. Tasa de corrección de errores (%correct). Se trata del porcentaje de corrección de errores, es decir, $nCE / (nCE + nNCE)$.

Los resultados de la evaluación (Tabla 7.1) muestran en primer lugar la eficacia de los gestores de diálogo desarrollados, alcanzando un porcentaje de éxito del 93,45% para la primera de las estrategias y un 97,33% para la segunda de ellas. Para la obtención de esta medida se ha tenido en cuenta si el usuario ha obtenido o no la totalidad de los objetivos definidos para el escenario, considerándose un diálogo sin éxito aquel en el que el usuario no haya obtenido el total de información definida para el escenario.

	% éxito	nT	% confirm	nCE	nNCE	% correct
Gestor Completa-Confirma	93,45	6,61	21,39	0,62	0,46	57
Gestor Dihana-Oz	97,33	9,03	35,04	0,86	0,11	89

Tabla 7.1: Resultados de la evaluación realizada para los modelos de gestión basados en reglas

La primera de las estrategias, basada en la petición de los atributos necesarios para realizar una consulta determinada y en la confirmación de los atributos con menor fiabilidad una vez están disponibles todos ellos, posee una media de turnos de sistema inferior a la de la segunda de las estrategias, basada en la confirmación de los atributos en el instante en el que se incorporan a la pizarra del diálogo. Este es el coste a pagar de cara

a obtener una mayor tasa de detección y corrección de los errores introducidos durante el diálogo.

Estos factores hacen posible que la segunda de las estrategias, a costa de incrementar la media de turnos de sistema y de confirmaciones explícitas por diálogo, logre un porcentaje de éxito y de corrección de errores mayor que la primera de las estrategias.

Este hecho se ve favorecido por la utilización de confirmaciones mixtas, que además de dotar de mayor naturalidad al diálogo, posibilitan al usuario un mayor grado de conocimiento de los atributos y valores que se introducirán en la sentencia de consulta a la base de datos, y que marcarán la validez de la información obtenida con respecto a la consulta requerida por el usuario. La utilización de confirmaciones mixtas incluso cuando la fiabilidad de la información supera el umbral prefijado, permite eliminar o rectificar de la consulta aquellos valores marcados como fiables de forma errónea, y que son los principales causantes del fracaso del diálogo para la primera de las estrategias.

7.3. Tratamiento de las situaciones no vistas mediante las metodologías estadísticas

La evaluación y comparación de las diferentes aproximaciones se llevó a cabo mediante un proceso de validación cruzada. El corpus se dividió aleatoriamente en cinco subconjuntos de 1.232 muestras (20 % del corpus). La evaluación consistió en cinco experimentaciones. En cada una de ellas se utilizó un subconjunto diferente de los cinco definidos como test, y el 80 % del corpus restante se utilizó como partición de entrenamiento.

El número de clases en \mathcal{C} (es decir, el número de posibles respuestas de sistema) es 51. La media de pares (DR, S) diferentes en los conjuntos de entrenamiento es 1.126. La media de situaciones no vistas es el 14.6 % de las muestras de los conjuntos de evaluación.

La Tabla 7.2 resume las principales características de las particiones definidas en el corpus DIHANA (P1-P5) para la evaluación de los diferentes gestores estadísticos.

	P1	P2	P3	P4	P5
Nº muestras de entrenamiento	4.927	4.927	4.927	4.927	4.928
Nº muestras de entrenamiento diferentes	1.123	1.104	1.127	1.144	1.133
Media de muestras entrenamiento/caso	4,4	4,5	4,4	4,3	4,4
Nº respuestas del sistema diferentes	48	45	48	50	51
Nº muestras de evaluación	1.232	1.232	1.232	1.232	1.231
Muestras no vistas	175	215	176	152	179
Porcentaje de muestras no vistas	14,2 %	17,5 %	14,3 %	12,3 %	14,5 %

Tabla 7.2: Principales características de las particiones definidas para la evaluación de los gestores estadísticos

Cabe indicar que, seleccionando la respuesta más vista en el entrenamiento como respuesta del sistema, el 22,12 % de respuestas proporcionadas por el gestor coinciden con la de referencia en el corpus. Esta medida nos indica un punto de partida para la tarea DIHANA. Desarrollando un gestor de diálogo que seleccionase aleatoriamente la nueva respuesta del sistema teniendo únicamente en cuenta el número de respuestas de cada tipo,

el porcentaje obtenido tras 1.000 iteraciones fue del 12,11% de respuestas coincidentes con la de referencia.

Para evaluar el funcionamiento de las diferentes metodologías se definieron cuatro medidas:

- *%exacta*: Se trata del porcentaje de respuestas que coinciden con aquella generada por el Mago de Oz (respuesta de referencia en el corpus).
- *%estrategia*: Es el porcentaje de respuestas que siguen la estrategia del Mago de Oz definida para DIHANA, ya que dicha estrategia posibilita un conjunto de respuestas válidas dado un determinado estado del diálogo.
- *%correcta*: Se trata del porcentaje que son coherentes con el estado actual del diálogo, aunque no necesariamente sigan la estrategia fijada para el Mago de Oz.
- *%error*: Es el porcentaje de respuestas que no son compatibles con el estado actual del diálogo, provocando el fallo del diálogo.

Estas tres últimas medidas han sido obtenidas tras un proceso de revisión manual. Los resultados que son relevantes en el ámbito de la gestión del diálogo son *%estrategia*, *%correcta* y *%error*. La Figura 7.1 muestra las medidas definidas para evaluar el gestor.

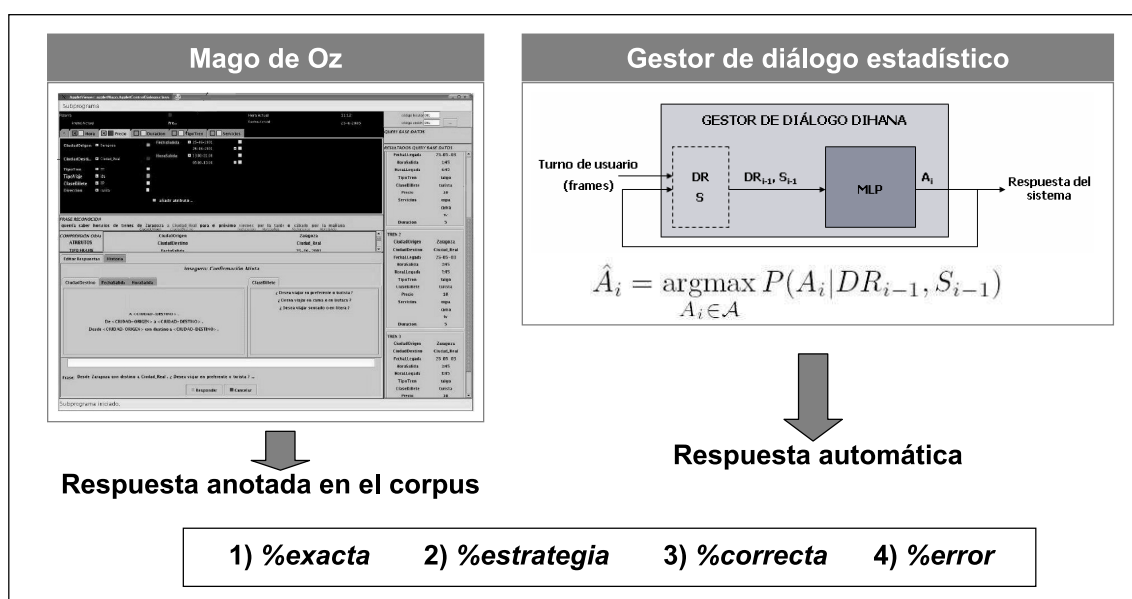


Figura 7.1: Medidas definidas para la evaluación de la metodología de gestión estadística

En el caso de los clasificadores mediante MLP, se utilizó el software **April** (*A Pattern Recognizer In Lua*) desarrollado en nuestro grupo de investigación (Zamora, 2005). El toolkit **April** intenta paliar las carencias que presentan muchas herramientas para el aprendizaje de redes neuronales. **April** incluye otras herramientas relacionadas con el reconocimiento de formas. En el campo de las RNA, la aplicación dispone de:

- Una implementación del algoritmo BP con momentum para poder entrenar cualquier red “hacia delante”, que no contenga ciclos, a una velocidad similar a otros algoritmos que, para tal fin, se limitan a topologías de tipo “capa a capa”.
- Una interfaz programable con el lenguaje de *script* Lua (Ierusalimschy, 2003; Ierusalimschy, de Figueiredo, y Celes, 2003).
- Una librería para describir conjuntos de datos.
- Objetos generadores de valores pseudo-aleatorios cuyo estado se puede guardar y recuperar.

Los MLP se entrenaron utilizando el algoritmo de Backpropagation con momentum. Se extrajo un subconjunto de validación (20%) de cada uno de los conjuntos de entrenamiento. La topología y parámetros del algoritmo (como la tasa de aprendizaje y el momento) se estimaron mediante una búsqueda exhaustiva, utilizando el error cuadrático medio (MSE) obtenido como criterio de parada para el conjunto de validación. La mejor topología fue dos capas ocultas con 100 y 10 neuronas respectivamente.

Los MLP se programan con el lenguaje de *script* Lua. La herramienta posee utilidades que permiten definir de forma fácil y flexible conjuntos de muestras mediante las clases *matrix* y *dataset*. Las Figuras B.2 y B.1 del Anexo B muestran respectivamente la programación realizada para generar y entrenar la red neuronal (Figura B.2), y para realizar la clasificación de las muestras del conjunto de evaluación (Figura B.1).

La Tabla 7.3 muestra los resultados obtenidos en la evaluación de la función distancia y de las diferentes funciones de clasificación para el tratamiento de las situaciones no vistas.

	<i>%exacta</i>	<i>%estrategia</i>	<i>%correcta</i>
Función Distancia	38,3 %	56,7 %	72,3 %
Clasificador Bigramas	30,4 %	37,7 %	48,7 %
Clasificador Trigramas	31,5 %	39,4 %	49,4 %
Clasificador MGGI	50,2 %	67,8 %	78,4 %
Clasificador Multinomial	56,3 %	75,2 %	88,5 %
Clasificador MLP	65,6 %	90,9 %	97,1 %

Tabla 7.3: Resultados de la evaluación del tratamiento de las situaciones no vistas

La Tabla 7.3 muestra que los mejores resultados se obtienen a través de la clasificación mediante el MLP. La codificación desarrollada para representar el estado actual del diálogo y el buen funcionamiento del clasificador basado en MLP hacen posible que la respuesta generada por el gestor coincida con una respuesta válida de acuerdo con la estrategia (*%estrategia*) en un porcentaje del 90,9 %.

Además, puede observarse que entre los clasificadores basados en modelos de estados finitos, los clasificadores de bigramas y trigramas tienen un peor comportamiento que el clasificador MGGI. Este comportamiento se debe a que estos modelos no pueden capturar dependencias a largo plazo y los modelos que estiman para las distintas clases no discriminan lo suficiente. La función definida para el clasificador MGGI proporciona modelos más

discriminantes que los anteriores pero, sin embargo, no consiguen alcanzar las prestaciones del clasificador multinomial y del MLP. Este comportamiento se debe a la dependencia de la metodología MGGI con respecto al orden de las variables, en este caso, con respecto a la variable anterior.

Los clasificadores basados en modelos de lenguaje capturan la información secuencial contenida en la codificación definida; en cambio el resto de los clasificadores tratan esta codificación sin tener en cuenta esta información secuencial, ya que en la codificación definida para el *DR* el orden de los diferentes elementos es irrelevante. Por los resultados obtenidos, parece que estas últimas aproximaciones resultan más adecuadas para el problema planteado.

La función Distancia presenta un mejor comportamiento que los clasificadores de bigramas y trigramas, obteniendo un porcentaje de respuestas correctas del 72,3 %, cercano al obtenido mediante el clasificador MGGI. La Tabla 7.4 muestra el resultado comparativo de la evaluación de la medida $\%error$ para el mejor de los clasificadores y para la función Distancia. Se observa la gran reducción alcanzada en el número de respuestas del gestor que provocarían un fallo crítico en el diálogo. Sin duda, ésta es la principal desventaja que presenta la función Distancia y que la utilización del MLP permite reducir.

	$\%error$
Función Distancia	22,8 %
Clasificador MLP	1,5 %

Tabla 7.4: Evaluación de la medida $\%error$ para la función distancia y el clasificador MLP

7.4. Evaluación del gestor de diálogo estadístico basado en redes neuronales

En este apartado se presentan los resultados de la evaluación del gestor de diálogo descrito en la Sección 6.5. Los resultados obtenidos para cada una de las medidas definidas aparecen en la Tabla 7.5. Dichos resultados muestran el funcionamiento satisfactorio del gestor de diálogo desarrollado. La codificación propuesta para representar el estado del diálogo y el buen funcionamiento del clasificador MLP hacen posible que la respuesta generada por el sistema coincida con una de las respuestas definidas en la estrategia en un porcentaje del 97,73 %. Dicha respuesta, además, coincide exactamente con la seleccionada por el Mago de Oz en un 78,62 % de los casos. Finalmente, el número de respuestas que pueden generar el fallo del sistema es sólo un 0,18 %. Teniendo en cuenta que la media de turnos de sistema por diálogo es aproximadamente siete, a partir del porcentaje $\%error$ obtenido puede estimarse que la probabilidad de que un diálogo sea fallido es del 1,25 %.

La Tabla 7.6 compara estos resultados con los que se obtienen con la utilización de la Función Distancia. De la observación de los resultados, cabe destacar los buenos resultados obtenidos por ambas aproximaciones con respecto al número de respuestas coherentes con el estado del diálogo. No obstante, de igual modo al comentado en el apartado anterior, la utilización del MLP permite reducir el porcentaje $\%error$ del 3,7 % al 0,18 %.

<i>%exacta</i>	78,62 %
<i>%estrategia</i>	97,73 %
<i>%correcta</i>	99,68 %
<i>%error</i>	0,18 %

Tabla 7.5: Evaluación del modelo estadístico basado en el MLP

	<i>%exacta</i>	<i>%estrategia</i>	<i>%correcta</i>	<i>%error</i>
Función Distancia	76,53 %	91,84 %	96,75 %	3,72 %
Clasificador MLP	78,64 %	97,75 %	99,72 %	0,18 %

Tabla 7.6: Comparativa entre la utilización de la función Distancia y el MLP

7.4.1. Evaluación de la influencia de la talla del corpus

Para realizar una evaluación del funcionamiento del modelo de diálogo frente a la talla del corpus de entrenamiento, se utilizaron las mismas particiones que en el apartado anterior, descartándose muestras de forma aleatoria en los conjuntos de entrenamiento para reducir el tamaño de esta partición. Se llevaron a cabo tres experimentaciones, utilizándose un 75 % de las muestras de entrenamiento (3.315 muestras), un 50 % de dicho conjunto (2.210 muestras) y un 25 % (1.105 muestras). Los conjuntos de test son idénticos a los del apartado anterior. La Tabla 7.7 muestra los resultados de las experimentaciones llevadas a cabo.

	100 %	75 %	50 %	25 %
<i>%exacta</i>	78,62 %	73,77 %	71,17 %	66,21 %
<i>%estrategia</i>	97,73 %	94,88 %	92,56 %	90,61 %
<i>%correcta</i>	99,68 %	98,62 %	97,33 %	96,10 %
<i>%error</i>	0,18 %	0,52 %	0,69 %	0,95 %

Tabla 7.7: Evaluación del modelo estadístico frente a diversas tallas del corpus de entrenamiento

Los resultados obtenidos muestran el correcto funcionamiento del gestor incluso si se utiliza únicamente un 25 % del corpus de entrenamiento para construir el modelo estadístico. De esta forma, los resultados obtenidos para la medida *%correcta* son muy similares en las tres experimentaciones. No obstante, si utilizamos únicamente un 25 % del corpus de entrenamiento, el porcentaje de respuestas del sistema que pueden causar el fallo del diálogo es del 0,95 %, lo que nos indica que aproximadamente una de cada cien respuestas dadas por el sistema ocasionará el fallo del diálogo (6,46 % de probabilidad de fallo del diálogo).

El alto porcentaje de acierto del gestor de diálogo incluso para un tamaño del corpus de entrenamiento reducido puede explicarse en parte porque existen muchos estados del diálogo que o bien son muy frecuentes en el corpus, o bien son similares a otros estados y, por tanto, fácilmente clasificables por el MLP.

7.4.2. Evaluación de la influencia del sexo

La influencia del sexo de los usuarios suele tenerse en cuenta en el reconocimiento automático del habla, pudiendo utilizarse modelos acústicos diferentes para cada caso. Para evaluar el comportamiento del gestor teniendo en cuenta el sexo de los usuarios, se realizaron particiones del corpus con igual número de muestras de mujeres y de hombres. Los resultados de esta experimentación aparecen en la Tabla 7.8, indicándose qué partición se utilizó como entrenamiento y como test (Entrenamiento / Test).

De la observación de los resultados, puede destacarse que prácticamente no existen diferencias remarcables en los resultados obtenidos si se realiza el aprendizaje del modelo utilizando únicamente muestras de hombres o de mujeres (primeras dos columnas de resultados en la Tabla 7.8). Mayores diferencias se observan en las experimentaciones llevadas a cabo evaluando el modelo teniendo en cuenta el sexo de los usuarios (en las particiones de test (tercera y cuarta columna de dicha figura). La diferencias obtenidas en esta última experimentación parecen indicar una mayor similitud en las muestras de hombres.

	Mujeres / Ambos	Hombres / Ambos	Ambos / Mujeres	Ambos / Hombres
<i>%exacta</i>	70,55 %	69,97 %	71,72 %	76,09 %
<i>%estrategia</i>	92,56 %	95,92 %	94,67 %	97,79 %
<i>%correcta</i>	96,93 %	97,96 %	97,75 %	99,71 %
<i>%error</i>	0,73 %	0,58 %	0,49 %	0,14 %

Tabla 7.8: Evaluación del modelo estadístico teniendo en cuenta el sexo de los usuarios

7.4.3. Evaluación de la procedencia de los diálogos

La evaluación del gestor de diálogo en este contexto se llevó a cabo partiendo de las mismas particiones que en el primer apartado de la evaluación, entrenando el modelo de diálogo únicamente con las muestras procedentes de la sede a evaluar (un tercio de los diálogos) y utilizando las mismas particiones de test que en dicho apartado (muestras procedentes de las tres sedes). Los resultados de la experimentación aparecen reflejados en la Tabla 7.9.

	Sede1	Sede2	Sede3
<i>%exacta</i>	71,94 %	77,24 %	70,39 %
<i>%estrategia</i>	95,55 %	95,51 %	90,03 %
<i>%correcta</i>	99,16 %	97,11 %	92,15 %
<i>%error</i>	0,42 %	1,44 %	3,92 %

Tabla 7.9: Evaluación del modelo estadístico teniendo en cuenta la procedencia de los diálogos

De la observación de los resultados de las diferentes experimentaciones, puede verificarse un mejor funcionamiento de la metodología teniendo en cuenta únicamente los

diálogos adquiridos en la Sede1. Utilizando sólo los diálogos de la Sede2, se obtiene un porcentaje de respuestas que siguen la estrategia ($\%estrategia$) equivalente al de la Sede1. Sin embargo, el porcentaje de respuestas erróneas es tres veces mayor. En cuanto al corpus de diálogos adquirido en la Sede3, a pesar de obtener un porcentaje de respuestas correctas del 92,15 %, el número de respuestas de sistema que pueden causar el fallo del diálogo es muy elevado en comparación con el obtenido en las dos sedes restantes (probabilidad del 24,42 % de que el diálogo sea fallido). Por tanto, teniendo en cuenta la medida $\%error$, existe una diferencia significativa entre los diálogos adquiridos en cada una de las sedes.

7.5. Evaluación utilizando la voz y usuarios reales

El funcionamiento del gestor del diálogo estadístico se ha evaluado con usuarios reales utilizando el conjunto de 15 escenarios relativos a consultas sobre horarios y/o precios descrito en la evaluación de los gestores basados en reglas. Se adquirieron un total de 150 diálogos, llevados a cabo por seis usuarios a través del sistema de diálogo desarrollado para el proyecto DIHANA. Se fijó un umbral de confianza de 0,5 para realizar la codificación de los datos del *DR*. Se consideraron las siguientes medidas:

1. Porcentaje de diálogos completados con éxito ($\% \text{ éxito}$). Para cada uno de los escenarios, el usuario debe completar uno o varios objetivos. El éxito del diálogo depende de que el sistema proporcione la información correcta para la totalidad de los objetivos establecidos en el escenario.
2. Número medio de turnos por diálogo (nST).
3. Porcentaje de situaciones no vistas durante el entrenamiento del modelo ($\% SNR$). Este valor representa el número de veces que el gestor debe buscar una respuesta aproximada en el modelo ($nSNR$), dividido por el número de turnos de sistema, ($nSNR/nST$).

Los resultados de la evaluación muestran (ver Tabla 7.10) que el modelo de diálogo aprendido automáticamente interactúa correctamente con el usuario, tanto con la utilización de la Función Distancia como con el MLP.

	Diálogos	nST	$\% \text{ éxito}$	SNR
Función Distancia	75	8,5	85,0 %	26,5 %
Clasificador MLP	75	8,2	93,0 %	23,7 %

Tabla 7.10: Evaluación del gestor de diálogo estadístico mediante pruebas con voz

Para el caso de la función Distancia, aunque los diálogos no finalizan correctamente en un 15,0 % de los casos, cabe destacar que el sistema proporciona la información correcta al usuario en un 94,7 % de los casos. El problema es, que en algunos casos, el sistema no detecta que el usuario desea finalizar el diálogo. Este problema se debe al hecho de que la distancia definida para tratar las situaciones no vistas se basa únicamente en el *DR*, con lo que puede seleccionarse una respuesta que no suponga una acción de *Cierre*.

En la Tabla 7.10 puede observarse también que el número de situaciones no representadas en el modelo es significativo (26,5% y 23,7%), tal y como era de esperar debido al limitado número de muestras de aprendizaje y el gran espacio de situaciones posibles. Sin embargo, los resultados confirman el buen funcionamiento de las aproximaciones propuestas para el tratamiento de las situaciones no vistas, dado que el sistema proporciona correctamente la información en el mayor número de casos.

Capítulo 8

Simulación de usuarios

8.1. Introducción

En este capítulo de la tesis se describen diferentes aproximaciones para la generación automática de diálogos. Gracias a estas técnicas, el sistema de diálogo desarrollado para DIHANA, además de poder interactuar con usuarios reales, permite la interacción con un módulo adicional capaz de desempeñar el papel del usuario.

Tal y como se ha comentado en la introducción y estado del arte de la tesis, una cuestión de gran interés en el campo de las Tecnologías del Habla es el desarrollo de herramientas y técnicas que faciliten la evaluación de los sistemas de diálogo. Una técnica en auge desde años recientes se basa en la generación automática de conversaciones entre el gestor de diálogo, y un módulo adicional llamado simulador del usuario. La finalidad de realizar modelos de usuario no reside en modelar las características precisas de cada uno de los usuarios, sino servir como herramienta en la fase de desarrollo de un nuevo sistema de diálogo y evaluar sus características.

La primera de las aproximaciones se basa en el desarrollo de un modelo estadístico para la simulación del usuario, aprendido a partir de un corpus de diálogos. Utilizando este modelo se ha construido un módulo de simulación de usuarios que amplía la formulación propuesta en la tesis para el desarrollo de un gestor de diálogo estadístico. En esta aproximación, el nuevo turno de usuario se selecciona teniendo en cuenta la información proporcionada por el usuario simulado a lo largo de la historia del diálogo, la última respuesta del usuario y el objetivo(s) prefijado para el diálogo. El turno de usuario, que se representa mediante actos de diálogo, se selecciona utilizando la distribución de probabilidad que proporciona un MLP. Esta metodología ha dado lugar a la publicación científica (Hurtado et al., 2007a)

Este módulo permite la adquisición de diálogos con un coste reducido, posibilitando la mejora del modelo inicial de diálogo aprendido a partir de un corpus adquirido con usuarios reales, pudiendo aumentar su variabilidad y detectar situaciones en las que el gestor no proporciona una respuesta apropiada.

Además, el modelado estadístico de usuarios reales se ha sugerido como solución a la falta de los datos necesarios para realizar el entrenamiento y la evaluación de los modelos de gestión del diálogo aprendidos de forma automática. El usuario simulado también permite explorar estrategias de diálogo no presentes en el corpus. De esta forma, el gestor del diálogo puede llegar a desviarse de las estrategias iniciales y aprender nuevas estrategias

potencialmente mejores. De este modo, puede entrenarse un modelo de usuario probabilístico que simule las intervenciones del usuario y utilizarlo así en el aprendizaje de la estrategia mediante la interacción entre el gestor del diálogo y el usuario simulado.

La segunda aproximación se basa en la adquisición de un nuevo corpus de diálogos mediante la definición de la semántica del usuario y del sistema, en forma de actos de diálogo, y la selección aleatoria de dichas respuestas, conjuntamente con la definición de un conjunto de condiciones de parada que posibiliten la evaluación automática de los diálogos obtenidos. Esta metodología ha posibilitado la publicación científica (Griol et al., 2007a).

8.2. Simulador de usuarios estadístico

Tal y como se ha descrito en el capítulo cuarto de la tesis, el sistema de diálogo desarrollado para DIHANA está compuesto por un total de siete módulos: un reconocedor automático del habla, un módulo de comprensión del habla, un gestor del diálogo, un módulo de consulta a la base de datos, un generador de respuestas en lenguaje natural, un sintetizador de texto a voz y finalmente por un gestor central de comunicaciones.

Mediante la incorporación del módulo de simulación de usuarios el sistema permite dos modos de funcionamiento. En primer lugar, la interacción entre el sistema y usuarios reales. Para ello, se hace uso del reconocedor automático del habla y el módulo de comprensión semántica. En segundo lugar, el sistema posibilita la adquisición automática de diálogos mediante la utilización del módulo simulador de usuarios. La Figura 8.1 muestra el diagrama de bloques del sistema de diálogo desarrollado, indicando los dos modos de funcionamiento descritos: (1) usuarios reales y (2) simulador de usuarios.

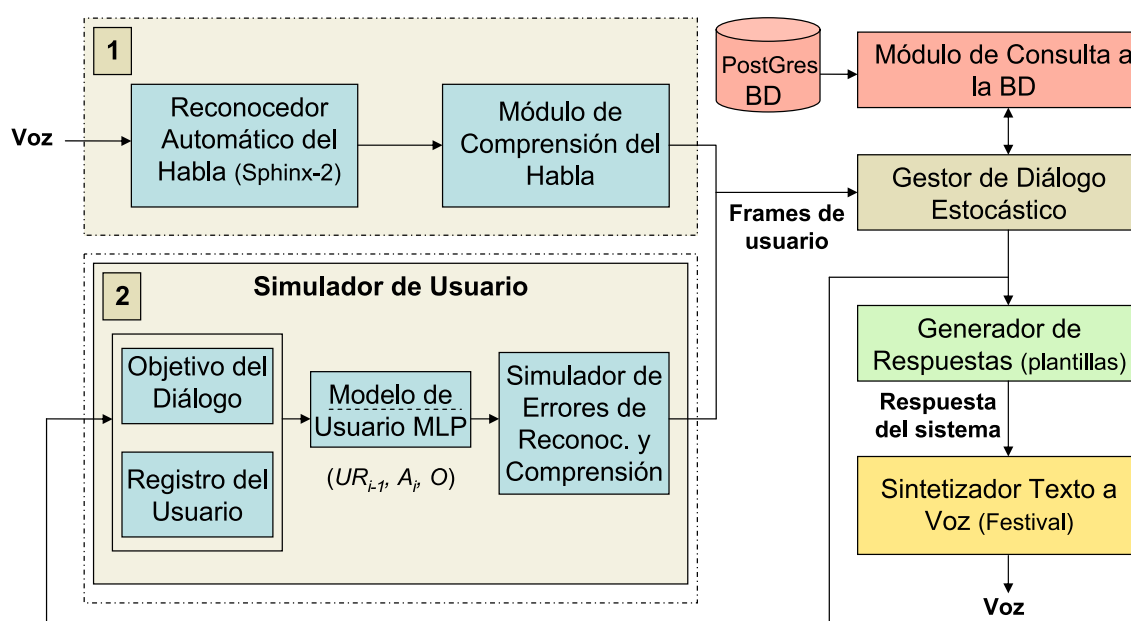


Figura 8.1: Diagrama de bloques del sistema de diálogo desarrollado para el proyecto DIHANA

Tal y como se observa en esta figura, en nuestro sistema, el simulador del usuario substituye las funciones realizadas por el reconocedor automático del habla y el módulo de comprensión del habla. Por tanto, genera frames en el mismo formato definido para la salida del módulo de comprensión, es decir, en el formato esperado por el gestor de diálogo.

La metodología que hemos desarrollado para la simulación del usuario se basa en la modelización estadística de las secuencias de actos de diálogo del usuario y del sistema. Tal y como se ha comentado en la introducción, en la generación de las respuestas del usuario se considera la información proporcionada por el simulador a través de la historia del diálogo, la última respuesta del sistema y los objetivos predefinidos para el diálogo. El modelo de usuario se ha aprendido a partir del etiquetado realizado a los turnos de usuario en el corpus DIHANA.

La descripción formal del modelo propuesto extiende la metodología estadística definida en DIHANA para la gestión del diálogo:

Sea A_i la salida generada por el gestor de diálogo (respuesta del sistema) en el instante i , expresada mediante actos de diálogo. Sea U_i la representación semántica del turno de usuario. De forma análoga a la mostrada en la descripción del gestor de diálogo, podemos representar el diálogo mediante una secuencia de pares (*turno-sistema, turno-usuario*):

$$(A_1, U_1), \dots, (A_i, U_i), \dots, (A_n, U_n)$$

donde A_1 es el turno de bienvenida del sistema (el primer turno del diálogo), y U_n el último turno de usuario. Denotamos al par (A_i, U_i) como S_i , el estado de la secuencia del diálogo en el instante i .

Dada esta representación, el objetivo del simulador de usuario en el instante i es encontrar una respuesta de usuario apropiada U_i . Esta selección, que es un proceso local para cada instante i , tiene en cuenta la secuencia de pares de turnos que precedieron al instante i , la respuesta proporcionada por el sistema en dicho instante y el objetivo del diálogo \mathcal{O} . En caso de seleccionar la respuesta de usuario más probable U_i para cada instante i , esta selección se realiza aplicando la siguiente maximización:

$$\hat{U}_i = \operatorname{argmax}_{U_i \in \mathcal{U}} P(U_i | S_1, \dots, S_{i-1}, A_i, \mathcal{O})$$

donde el conjunto \mathcal{U} contiene todas las posibles respuestas definidas para el usuario.

Dado que el número de posibles secuencias de estados es muy grande, hemos establecido una partición en dicho espacio, es decir, en la historia del diálogo que precede al instante i . Para ello, definimos el Registro del Usuario (*User Register, UR*). Este registro se define como una estructura de datos que contiene la información sobre los conceptos y atributos proporcionados por el usuario a lo largo del diálogo.

La Tabla 8.1 muestra la reducción del número de estados que se consigue para el corpus DIHANA con la introducción del *UR*.

Sea UR_i el registro del usuario en el instante i . La información contenida en UR_i es un resumen de la información proporcionada por el simulador de usuario a lo largo de la secuencia S_1, \dots, S_{i-1} . Diferentes secuencias de estados pueden conducir al mismo *UR*.

$$\begin{array}{ccccccc} & S_1, & & \dots, & & S_i, & \dots, & S_n \\ \uparrow & & \uparrow & & \uparrow & & \uparrow & \\ UR_0 & & UR_1 & & UR_{i-1} & & UR_{n-1} & \end{array}$$

UR distintos	232
Pares (UR, S) distintos	933
Secuencias S_1, \dots, S_{i-1}	4.290

Tabla 8.1: Reducción del espacio de estados tras la introducción del UR en DIHANA

donde UR_0 está vacío, y los siguientes UR_i se actualizan considerando la información suministrada por el simulador de usuario.

La partición que establecemos en este espacio se basa en considerar que *dos secuencias de estados diferentes son equivalentes si conducen al mismo UR* . Esta suposición provee una gran reducción en el número de historias diferentes en los diálogos a expensas de la pérdida del orden cronológico en el que el usuario suministró la información. Sin embargo, consideramos que ésta es una pérdida menor, dado que el orden en el que el usuario suministró la información no es un factor relevante para determinar la próxima respuesta del usuario U_i .

Tras aplicar estas consideraciones y establecer las relaciones de equivalencia en las historias de los diálogos, la selección de la mejor respuesta del usuario U_i viene dada por:

$$\hat{U}_i = \operatorname{argmax}_{U_i \in \mathcal{U}} P(U_i | UR_{i-1}, A_i, \mathcal{O}) \quad (8.1)$$

De forma análoga al trabajo llevado a cabo en el ámbito de la gestión de diálogo, proponemos la utilización de un MLP para seleccionar el turno de usuario. La capa de entrada recibe la codificación de la situación actual del diálogo, representada por el término $(UR_{i-1}, A_i, \mathcal{O})$ en la Ecuación 8.1. Los valores que proporciona el perceptrón en la capa de salida pueden verse como la probabilidad a posteriori de seleccionar cada una de las respuestas del usuario definidas para el usuario, dada la situación actual del diálogo. La selección de la respuesta del usuario más probable en esta distribución de probabilidades conduce a la Ecuación 8.1. En este caso, el simulador de usuarios generará siempre la misma respuesta para la misma situación del diálogo. Dado que deseamos que el simulador provea una variabilidad mayor de respuestas para poder modelar los posibles comportamientos de los usuarios, para la selección de la respuesta del usuario se considera la distribución de probabilidades proporcionada por el simulador para todo el conjunto de respuestas que se han definido para el usuario.

8.2.1. Codificación definida para el clasificador MLP

La representación definida para la entrada $(UR_{i-1}, A_i, \mathcal{O})$ consiste en un vector de 17 características:

$$(UR_{i-1}, A_i, \mathcal{O}) = (\vec{x}_1, \vec{x}_2, \vec{x}_3, \dots, \vec{x}_{17})$$

Los términos que componen este vector se enumeran a continuación:

- La última respuesta proporcionada por el sistema (A_i): Se han tenido en cuenta los dos primeros niveles del etiquetado de la última respuesta dada por el sistema, de forma idéntica a la propuesta en el marco de la gestión de diálogo. Esta información

se modela mediante una variable, que posee tantos bits como posibles combinaciones de estos dos niveles (51).

$$\vec{x}_1 = (x_{1_1}, x_{1_2}, x_{1_3}, \dots, x_{1_{51}}) \in \{0, 1\}^{51}$$

- El Registro del Usuario (UR_{i-1}): Para la tarea DIHANA, el UR es una secuencia de quince campos correspondientes a los cinco conceptos y diez atributos dependientes de la tarea. La definición, tal y como muestra la Figura 8.2 es equivalente a la del Registro del Diálogo (DR) definido para el gestor. Del mismo modo, para determinar la siguiente respuesta del simulador de usuario, hemos asumido que los valores exactos de los atributos no son significativos. Estos valores son necesarios para acceder a la base de datos y construir la respuesta final del sistema, sin embargo, la única información que hemos considerado necesaria para determinar la siguiente acción del usuario es la presencia o ausencia de conceptos y atributos. Por tanto, la información contenida en el UR es una codificación de esta información mediante dos valores:
 - **0** si el usuario no ha pedido información sobre el concepto no está activado o no ha proporcionado el valor del atributo.
 - **1** si el concepto o atributo está activado.

Conceptos	Atributos
Horario	Origen
Precio	Destino
Tipo-tren	Fecha-Salida
Tiempo-Recorrido	Fecha-Llegada
Servicios	Hora-Salida
	Hora-Llegada
	Clase
	Tipo-Tren
	Número-Orden
	Servicios

Figura 8.2: Estructura de campos definida para el Registro del Usuario (UR) en DIHANA

Cada uno de los atributos y conceptos del UR puede modelarse utilizando una variable con dos bits.

$$\vec{x}_i = (x_{i_1}, x_{i_2}) \in \{0, 1\}^2 \quad i = 2, \dots, 16$$

- El objetivo del diálogo (\mathcal{O}): En cuanto al número de objetivos considerado para codificar la variable objetivo del diálogo, se ha tenido en cuenta los diferentes tipos de escenarios definidos para realizar la adquisición del corpus DIHANA. De este modo, se han tenido en un cuenta un total de 20 objetivos, resumidos en la Figura 8.3. Esta información, por tanto, se modela mediante una variable con 20 bits.

$$\vec{x}_{17} = (x_{17_1}, x_{17_2}, x_{17_3}, \dots, x_{17_{20}}) \in \{0, 1\}^{20}$$

Código	Objetivo	Información que debe aportar el usuario
A0	Conocer Horarios	Destino, Fecha
A1	Conocer Horarios	Destino, Fecha, Hora
A2	Conocer Horarios y Tipos de Tren	Destino, Fecha
A3	Conocer Horarios	Destino, Fecha, Tipo-Tren
A4	Conocer Horarios	Destino, Fecha, Hora, Tipo-Tren
A5	Conocer Horarios	Origen, Destino, Fecha, Hora
A6	Conocer Horarios y Tipos de Tren	Origen, Destino, Fecha
A7	Conocer Horarios	Origen, Destino, Fecha, Tipo-Tren
A8	Conocer Horarios y Tipos de Tren	Origen, Destino, Fecha, Hora
A9	Conocer Horarios	Origen, Destino, Fecha, Hora, Tipo-Tren
B0	Conocer Horarios y Precios	Destino, Fecha
B1	Conocer Horarios y Precios	Destino, Fecha, Hora
B2	Conocer Horarios y Precios	Destino, Fecha, Tipo-Tren
B3	Conocer Horarios y Precios	Destino, Fecha, Hora, Tipo-Tren
B4	Conocer Horarios y Precios	Origen, Destino, Fecha, Hora
B5	Conocer Horarios y Precios	Origen, Destino, Fecha, Tipo-Tren
B6	Conocer Horarios y Precios	Origen, Destino, Fecha, Hora, Tipo-Tren
B7	Conocer Precios	Destino, Fecha, Tipo-Tren
B8	Conocer Precios	Origen, Destino, Fecha, Tipo-Tren
B9	Conocer Precios	Origen, Destino, Fecha, Hora, Tipo-Tren

Figura 8.3: Conjunto de objetivos definidos en los escenarios de la tarea DIHANA

8.2.2. Implementación práctica del simulador estadístico

Los componentes fundamentales que constituyen el simulador de usuario se muestran en la Figura 8.1. La aplicación del modelo de diálogo estadístico para la selección de la nueva respuesta del usuario, se acompaña de un conjunto de condiciones de parada que posibilitan la evaluación automática de los diálogos adquiridos. Las condiciones definidas son las siguientes:

- Teniendo en cuenta las características de los diálogos que componen el corpus DIHANA se ha establecido experimentalmente un número máximo de turnos de usuario por diálogo. De este modo, se consideran válidos los diálogos que alcanzan el objetivo prefijado en un número de turnos menor o igual al establecido.
- La segunda condición se aplica para generar una petición de cierre del diálogo por parte del usuario cuando el gestor haya proporcionado el conjunto de información prefijado en los objetivos del diálogo. En el caso de que el gestor del diálogo indique, tras suministrar la información referente a una determinada consulta, si el usuario desea realizar alguna consulta adicional o finalizar el diálogo, se realiza una consulta al registro de objetivos para seleccionar la respuesta generada por el modelo (caso de que el sistema no haya proporcionado la información correspondiente a todos los objetivos prefijados para el diálogo) o finalizar el diálogo.

Además se aplica una condición adicional para la confirmación de los datos requeridos por el gestor del diálogo. En caso de que el gestor del diálogo requiera la confirmación

de un concepto o atributo, el simulador de usuarios añadirá a la respuesta seleccionada por el MLP los actos de diálogo (*Afirmación*) o (*Negación*), en función de que el dato suministrado por el gestor coincida o no con el indicado en el registro objetivo del diálogo.

El algoritmo seguido por el simulador para generar una nueva respuesta de usuario es el siguiente:

- El Registro Objetivo del Diálogo almacena el objetivo establecido para los diálogos que se desean adquirir. Estos objetivos aparecen reseñados en la Figura 8.3.
- El simulador recibe la representación en actos de diálogo de la respuesta generada por el gestor de diálogo.
- A partir del objetivo del diálogo, la información almacenada en el UR y la respuesta proporcionada por el gestor de diálogo, se codifica la entrada $(UR_{i-1}, A_i, \mathcal{O})$ siguiendo el formato y orden mostrado en el apartado anterior.
- El MLP clasifica la entrada $(UR_{i-1}, A_i, \mathcal{O})$ y, en base a la distribución de probabilidades obtenida a su salida, se selecciona la nueva respuesta del usuario.
- Se actualiza el Registro del Usuario de acuerdo con la respuesta seleccionada por el MLP.
- Finalmente, se comprueban las condiciones de parada para determinar si el gestor de diálogo ha proporcionado la información correspondiente al objetivo definido o si el número máximo de turnos de usuario se ha superado.

La salida generada por el simulador constituye la entrada de un módulo encargado de introducir errores, modificando los frames generados por el simulador del usuario e incorporar las medidas de confianza asociadas a los conceptos y atributos. Este proceso se lleva a cabo de acuerdo con un análisis realizado al corpus DIHANA.

El corpus DIHANA incluye información sobre los errores que fueron introducidos durante la adquisición por el reconocedor automático del habla y el módulo de comprensión semántica. Esta información también incorpora las medidas de la confianza. Experimentalmente, hemos detectado 2,7 errores por diálogo. Este valor se puede modificar para adaptar el módulo del simulador de errores al funcionamiento de cualquier reconocedor y módulo de comprensión.

De esta forma, el UR se actualiza con la información facilitada a la salida del simulador de usuarios. Los valores de cada una de las posiciones de este registro pertenecen al conjunto $\{0, 1\}$. El simulador de errores se encarga, tal y como se ha comentado, de introducir errores en la salida generada por el simulador de usuarios e incorporar las medidas de confianza para concepto y atributo. Por tanto, la información contenida en el DR en un determinado instante del diálogo puede diferir de la almacenada en dicho instante en el UR . Con la incorporación de las medidas de confianza, la información contenida en el DR puede tomar los valores del conjunto $\{0, 1, 2\}$.

Las Figuras 8.4 y 8.5 muestran dos ejemplos de diálogos adquiridos mediante la interacción del simulador de usuarios descrito y el gestor de diálogo estadístico presentado en la Sección 6.4. El objetivo marcado en ambos diálogos es la obtención de horarios y precios para ir de Valencia a Barcelona el día 30-12-2006. En ambos ejemplos, las medidas de confianza generadas por el simulador de errores se muestran entre paréntesis.

<p>Respuesta generada por el gestor de diálogo: Respuesta del gestor: Apertura-[Nil]</p>
<p>Respuesta generada por el simulador de usuario: (Hora-Salida) [0,9] Origen: Alicante [0,1] Destino: Barcelona [0,9]</p> <p>Respuesta generada por el gestor de diálogo: Par (DR, S): Apertura-[Nil]*2100001000-10000*100 Respuesta del gestor: Confirmación-Origen</p>
<p>Respuesta generada por el simulador de usuario: (Negación)[0,9] Origen: Valencia [0,9]</p> <p>Respuesta generada por el gestor de diálogo: Par (DR, S): Confirmación-Origen*1100001000-10000*010 Respuesta del gestor: Pregunta-[Día]</p>
<p>Respuesta generada por el simulador de usuario: () [0,9] Fecha-Salida: 30-12-2006 [0,9] Hora-Salida: 08.00-23.00 [0,9]</p> <p>Respuesta generada por el gestor de diálogo: Par (DR, S): Pregunta-[Día]*1110101000-10000*000 Respuesta del gestor: Confirmación,Respuesta-Hora_salida,Hora_salida</p> <p>Consulta BD: (Hora-Salida): Origen:Valencia, Destino:Barcelona, Fecha-Salida:30-12-2006, Hora-Salida:08.00-23.00, Clase:preferente</p>
<p>Respuesta generada por el simulador de usuario: (Precio) [0,9]</p> <p>Respuesta generada por el gestor de diálogo: Par (DR, S):Confirmación,Respuesta-Hora_salida,Hora_salida*1110101000-01000*100 Respuesta del gestor:Respuesta-Precio</p> <p>Consulta BD: (Precio): Origen:Valencia, Destino:Barcelona, Fecha-Salida:30-12-2006, Hora-Salida:08.00-23.00, Clase:preferente</p>
<p>Respuesta generada por el simulador de usuario: (Negación) [0,9]</p> <p>Respuesta generada por el gestor de diálogo: Par (DR, S): Respuesta-Precio*1110101000-01000*010 Respuesta del gestor: Cierre-[Nil]</p>

Figura 8.4: Ejemplo de diálogo generado con el simulador de usuarios

En el diálogo mostrado en la Figura 8.4, el simulador indica en el primer turno de usuario que desea consultar horarios y aporta el origen y el destino. El simulador de errores introduce en dicho turno un error en la ciudad origen (cambia el valor Valencia por Alicante) y asigna medidas de confianza a los diferentes datos aportados. En dicha asignación, al valor de la ciudad origen se le asigna una baja fiabilidad. En el siguiente turno, el sistema solicita la confirmación del origen. El simulador de usuario aporta de nuevo en el siguiente turno la ciudad origen, asignándole esta vez el simulador de errores una fiabilidad alta.

En el siguiente turno, el sistema solicita la fecha de salida, necesaria para proporcionar la información sobre horarios. El simulador aporta en el siguiente turno dicha fecha y la hora de salida, marcada como dato opcional en el objetivo del diálogo. El simulador de errores asigna alta fiabilidad a ambos datos.

En el turno posterior, el sistema realiza la consulta sobre horarios y proporciona el resultado de la misma al simulador, indicándole si desea realizar una consulta adicional. Seguidamente, el simulador verifica en el Registro Objetivo del diálogo que dicho obje-

tivo todavía no se ha cumplido, seleccionando la respuesta que proporcione el modelo estadístico, en este caso, realizar una consulta sobre precios.

En el siguiente turno, el sistema proporciona la información sobre precios y consulta de nuevo al simulador si desea alguna consulta adicional. El simulador, tras verificar que el objetivo del diálogo se ha cumplido, selecciona como respuesta una petición de cierre del diálogo.

<p>Respuesta generada por el gestor de diálogo: Respuesta del gestor: Apertura-[Nil]</p>
<p>Respuesta generada por el simulador de usuario: (Afirmación) [0,1] Origen: Valencia [0,9] Destino: Barcelona [0,9] Fecha-Salida: 30-12-2006 [0,1]</p>
<p>Respuesta generada por el gestor de diálogo: Par (DR, S): Apertura-[Nil]*1120001000-00000*100 Respuesta del gestor: Confirmación-[Hora_salida]</p>
<p>Respuesta generada por el simulador de usuario: (Afirmación) [0,9]</p>
<p>Respuesta generada por el gestor de diálogo: Entrada GD: Confirmación-[Hora_salida]*1120001000-10000*100 Respuesta del gestor: Confirmación-Día</p>
<p>Respuesta generada por el simulador de usuario: (Afirmación) [0,9] Fecha-salida: 30-12-2006 [0,1]</p>
<p>Respuesta generada por el gestor de diálogo: (DR, S): Confirmación-Día*1120001000-10000*010 Respuesta del gestor: Confirmación-Día</p>
<p>Respuesta generada por el simulador de usuario: (Afirmación) [0,9]</p>
<p>Respuesta generada por el gestor de diálogo: (DR, S):Confirmación-Día*1110001000-10000*100 Respuesta del gestor: Confirmación,Respuesta-Hora_salida,Hora_salida</p> <p>Consulta BD:(Hora-Salida): Origen:Valencia, Destino:Barcelona, Salida:30-12-2006, Clase:preferente</p>
<p>Respuesta generada por el simulador de usuario: (Afirmación) [0,9] (Precio) [0,9]</p>
<p>Respuesta generada por el gestor de diálogo: (DR, S): Confirmación,Respuesta-Hora_salida,Hora_salida*1110001000-01000*000 Respuesta del gestor: Respuesta-Precio</p> <p>Consulta BD: (Precio): Origen:Valencia, Destino:Barcelona, Fecha-Salida:30-12-2006, Clase:preferente</p>
<p>Respuesta generada por el simulador de usuario: (Negación) [0,9]</p>
<p>Respuesta generada por el gestor de diálogo: (DR, S): Respuesta-Precio*1110001000-01000*010 Respuesta del gestor: Cierre-[Nil]</p>

Figura 8.5: Ejemplo de diálogo generado con el simulador de usuarios

En el diálogo mostrado en la Figura 8.5 el simulador proporciona en el primer turno el origen, el destino y la fecha de salida. El simulador de errores asigna una baja fiabilidad al valor de la fecha. En el siguiente turno, el sistema pregunta al usuario si desea llevar a cabo una consulta sobre horarios y el simulador contesta afirmativamente.

En el turno siguiente, el sistema requiere al simulador la confirmación de la fecha proporcionada. El simulador proporciona de nuevo la fecha y se le asigna nuevamente una confianza baja. En la siguiente respuesta, el sistema vuelve a pedir la confirmación de la fecha. El simulador genera un concepto (*Afirmación*).

Seguidamente, el sistema realiza la consulta a la base de datos sobre horarios y proporciona la información al simulador, requiriendo si desea llevar a cabo alguna consulta adicional. En el siguiente turno, el simulador consulta el Registro Objetivo del Diálogo y comprueba que el objetivo fijado todavía no se ha cumplido. La respuesta seleccionado por el modelo estadístico es un concepto (*Afirmación*) y la consulta de información sobre precios.

Finalmente, el sistema proporciona la información sobre precios y pregunta al usuario si desea una consulta adicional. Tras comprobar que el objetivo del diálogo se ha cumplido, el simulador genera una petición de cierre del diálogo.

8.2.3. Evaluación del simulador

Para realizar la evaluación del proceso de la simulación, se generaron un total de 100.000 diálogos. Tal y como se ha explicado en el apartado anterior, se definieron dos criterios para finalizar el diálogo. En primer lugar, el gestor de diálogo considera que el diálogo no es correcto y decide abortarlo cuando tienen lugar las siguientes condiciones:

- El diálogo excede de un máximo de número de turnos.
- La respuesta seleccionada por el gestor de diálogo se corresponde con una consulta no requerida por el simulador de usuarios.
- El módulo de consulta a la base de datos genera un mensaje de error debido a que el simulador del usuario no ha proporcionado la información obligatoria necesaria para llevar a cabo la consulta.
- El generador de la respuesta proporciona un mensaje de error cuando la respuesta seleccionada por el gestor implica el uso de datos no contenidos realmente en el *DR*, es decir, no proporcionados por el simulador del usuario.

El segundo criterio se aplica para generar una petición de usuario de cierre del diálogo cuando el gestor de diálogo ha proporcionado el conjunto de información definido como objetivo(s) del diálogo. Los diálogos correctos son los que concluyen cuando se aplica el segundo criterio.

Se definieron siete medidas para la evaluación de los diálogos simulados:

- Número de los diálogos correctos (SD),
 - Número medio de turnos por diálogo (NT),
 - Número de diálogos correctos diferentes (DD),
 - Número de repeticiones del diálogo más visto (MR),
 - Número de turnos del diálogo correcto más largo (TM),
-

- Número de turnos del diálogo correcto más corto(TS),
- Número de diálogos simulados que están contenidos en el corpus DIHANA, sin tener en cuenta el valor concreto de los atributos (CD).

Mediante el uso de estas medidas, se intenta evaluar tanto el éxito de los diálogos simulados como su eficacia y variabilidad con respecto a los diversos objetivos.

La Tabla 8.3 muestra los valores obtenidos para las diferentes medidas para el corpus simulado de 100.000 diálogos. Los objetivos del diálogo se han agrupado en cuatro conjuntos. Los tipos S1 y S2 engloban a los diálogos que poseen un único objetivo. En el tipo S1 la información marcada en el objetivo es la estrictamente necesaria para realizar el tipo de consulta requerido a la base de datos. En el tipo S2 el simulador debe proporcionar información adicional, no definida como necesaria para la consulta requerida. Los tipos S3 y S4 se corresponden con diálogos con dos objetivos prefijados. En el tipo S3, la información que debe aportar el simulador es la estrictamente necesaria para cumplir ambos objetivos. El tipo S4 incorpora atributos adicionales. La Tabla 8.2 muestra la agrupación de los tipos de escenarios definidos en DIHANA en los cuatro grupos descritos.

Tipo S1	A0, B7, B8
Tipo S2	A1, A3, A4, A5, A7, A9, B9
Tipo S3	A2, A6, B0, B2, B5
Tipo S4	A8, B1, B3, B4, B6

Tabla 8.2: Agrupación de escenarios definida para la evaluación del corpus simulado

Teniendo en cuenta los cuatro tipos de diálogos, el simulador generó un total de 22.614 diálogos que consiguieron alcanzar el objetivo marcado (22,6%). El número de diálogos simulados que están contenidos en el corpus inicial, CD, se corresponde sólo con una pequeña proporción de dichos diálogos (1,2%). Puede observarse cómo los diálogos del tipo S3 y S4 son más difíciles de simular debido a que los dos objetivos deben ser alcanzados.

Tipo	SD	NT	DD	MR	TM	TS	CD
S1	9.267	10,0	5.191	293	2	2	71
S2	9.133	10,8	6.359	41	3	2	153
S3	2.281	11,4	2.083	11	5	3	32
S4	1.933	12,6	1.750	14	3	3	10

Tabla 8.3: Evaluación del corpus adquirido mediante el simulador de usuarios

Además, se llevó a cabo una experimentación para determinar el comportamiento del gestor de diálogo desarrollado para DIHANA al añadir los diálogos simulados. En primer lugar, se evaluó el comportamiento del gestor de diálogo aprendido utilizando el corpus DIHANA. Seguidamente, se evaluó su evolución a medida que se incorporan los diálogos simulados correctos al corpus de entrenamiento. Se aprendió un nuevo modelo para el gestor de diálogo cada vez que se generaba un nuevo conjunto de diálogos simulados. Para llevar a cabo esta evaluación, se utilizó un subconjunto de evaluación extraído del corpus DIHANA (20% de las muestras).

Para realizar la evaluación se consideraron las siguientes medidas:

- *#no-vistas*: Se trata del número de situaciones no vistas, es decir, las situaciones del diálogo que están presentes en la partición de evaluación pero no en la corpus utilizado para aprender el gestor de diálogo;
- *#error*: Se trata del número de respuestas proporcionadas por el gestor que causaría el fallo del diálogo.
- *%estrategia*: Es el porcentaje de las respuestas proporcionada por el gestor que siguen exactamente la estrategia definida para que el Mago de Oz adquiriera el corpus DIHANA;
- *%correcta*: Es el porcentaje de respuestas proporcionadas por el gestor que son coherentes con el estado actual del diálogo, aunque no sigan necesariamente la estrategia original definida para el Mago de Oz.

Las Figuras 8.6 y 8.7 muestran respectivamente el decrecimiento de las medidas *#no-vistas* y *#error* conforme se van incorporando los diálogos simulados al corpus de entrenamiento, siendo el comportamiento esperado. Estas medidas decrecen hasta que se simularon 60.000 diálogos.

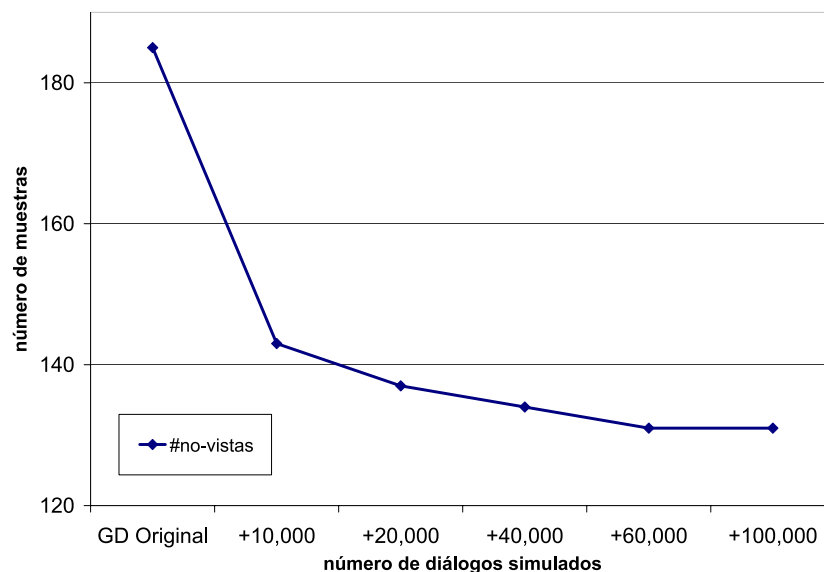


Figura 8.6: Evolución de la medida *#no-vistas* con respecto a la incorporación de nuevos diálogos simulados

La Figura 8.8 muestra la evolución de las medidas *%estrategia* y *%correcta*. Puede observarse que el gestor de diálogo mejora la generación de respuestas correctas conforme se incorporan los diálogos simulados. Además, se incrementa el número de respuestas correctas que son diferentes a aquellas definidas para la estrategia del Mago de Oz. En otras palabras, se modifica la estrategia inicial, de tal modo que se posibilita que el gestor de diálogo pueda abordar nuevas situaciones y generar nuevas respuestas coherentes con dichas situaciones.

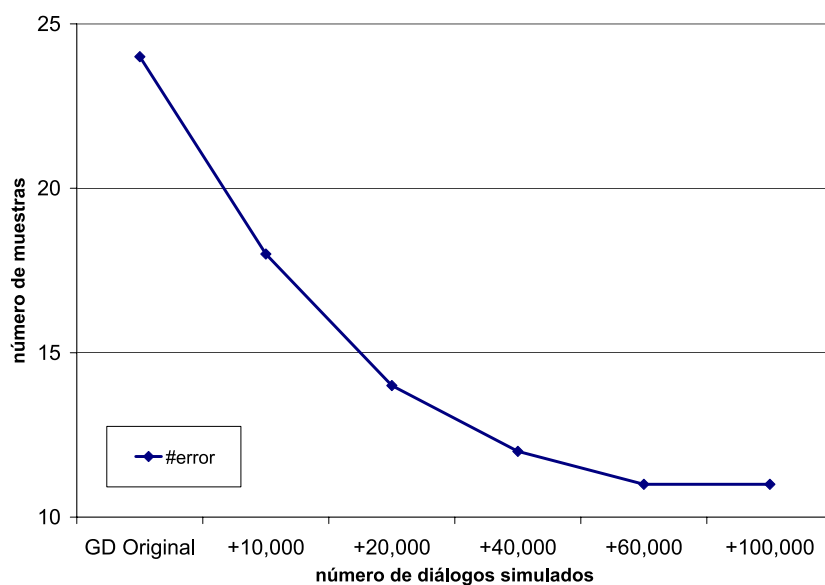


Figura 8.7: Evolución de la medida $\#error$ con respecto a la incorporación de nuevos diálogos simulados

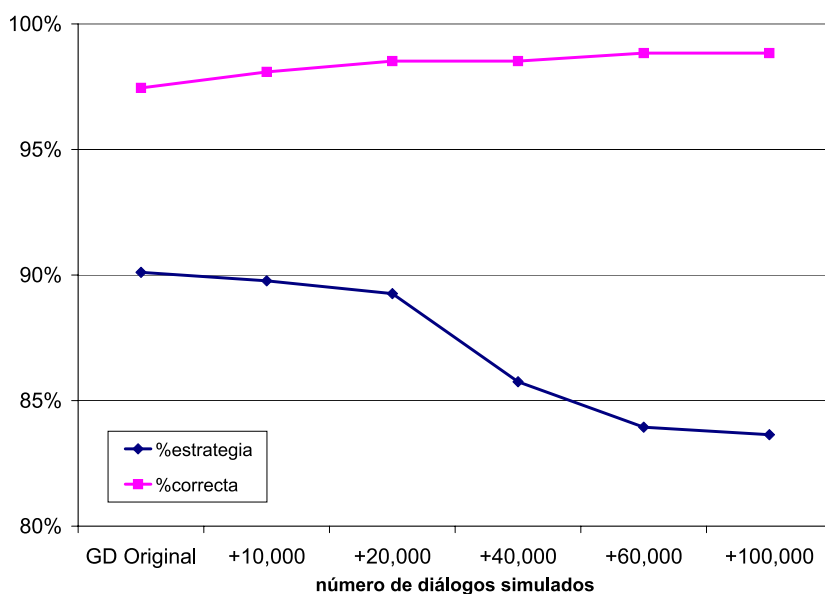


Figura 8.8: Evolución de las medidas $\%estrategia$ y $\%correcta$ con respecto a la incorporación de nuevos diálogos simulados

8.3. Definición de una técnica para la adquisición automática de diálogos

En la sección anterior se ha descrito una metodología estadística que permite la implementación de un módulo de simulación de usuarios. Mediante la interacción de este módulo con cualquiera de los gestores de diálogo descritos en la tesis, es posible realizar la adquisición de un corpus de diálogos. En dicho apartado se ha evaluado el corpus adquirido mediante la participación del simulador de usuarios descrito y el gestor de diálogo

presentado en la Sección 6.4.

En esta sección se presenta una metodología para simular tanto el comportamiento del usuario como el del sistema simultáneamente, de forma que pueda adquirirse un corpus de diálogos mediante la interacción de estos dos módulos. Esta aproximación se basa en una selección aleatoria de las respuestas del usuario y del sistema. Los únicos parámetros que se requieren para realizar esta adquisición son la definición de la semántica de la tarea (es decir, el conjunto de respuestas posibles para el usuario y el sistema), y un conjunto de condiciones que permitan descartar automáticamente los diálogos que no han alcanzado el objetivo predefinido.

Utilizando esta técnica se ha adquirido un nuevo corpus para la tarea DIHANA. Este corpus se ha utilizado para entrenar el gestor de diálogo estadístico descrito en la Sección 6.4. El gestor desarrollado se ha evaluado utilizando el corpus DIHANA como conjunto de evaluación, de forma que se posibilita la evaluación del gestor con un corpus de diálogos adquirido con usuarios reales.

La simulación se ha llevado a cabo en el nivel de la intención del usuario, es decir, se simulan los conceptos y atributos que proporciona el usuario en cada una de sus intervenciones. De este modo, el simulador de usuario realiza las funciones del reconocedor automático del habla y del módulo de comprensión del habla.

La semántica seleccionada para el gestor de diálogo se representa mediante las 51 posibles respuestas del sistema definidas para la tarea, contempladas en el corpus DIHANA. La selección de las posibles respuestas del usuario se llevó a cabo utilizando la semántica definida para el usuario en el módulo de comprensión.

Se ha utilizado el módulo de simulación de errores descrito en la sección anterior. Este módulo, tal y como se ha expuesto anteriormente, introduce errores en la información generada por el simulador de usuario e incorpora las medidas de confianza para los diferentes conceptos y atributos que aporte el simulador de usuario.

8.3.1. Adquisición de un corpus de diálogos

Para realizar la adquisición de un nuevo corpus de diálogos mediante esta técnica, se definió un número máximo de turnos por diálogo, determinado experimentalmente a partir del análisis del corpus DIHANA.

El gestor de diálogo considera que el diálogo no es correcto y decide abortarlo cuando el número de turnos de sistema excede un umbral obtenido experimentalmente o el módulo de generación de respuestas o de consulta a la base de datos proporcionan un error.

De forma análoga al simulador de usuario estadístico, se selecciona una petición de usuario para cerrar el diálogo una vez que el sistema haya proporcionado la información definida en los objetivos del diálogo. Los diálogos que cumplen esta condición antes del número máximo de turnos prefijado se consideran correctos.

Un total de dos millones de diálogos simulados fue necesario para adquirir un corpus de diálogo con las mismas características y distribuciones de objetivos definido para el corpus DIHANA, utilizando el mismo conjunto de escenarios definido para su adquisición. El PC utilizado para la adquisición (Pentium IV Dual Core) simulaba 1.000 diálogos cada 40 minutos.

La Tabla 8.4 resume las estadísticas de la adquisición para los cuatro tipos de escenarios mostrados en la Tabla 8.2.

	Tipos S1 y S2	Tipos S3 y S4
Nº de diálogos simulados	10 ⁶	10 ⁶
Diálogos con éxito	15.383	1.010
Diálogos diferentes	14.921	998
Media de turnos de usuario por diálogo	4,9	6,2

Tabla 8.4: Estadísticas de la adquisición del corpus mediante la técnica de simulación de diálogos

8.3.2. Ejemplos de diálogos adquiridos

Seguidamente se muestran dos ejemplos de los diálogos que componen el corpus adquirido. Los turnos con *S* hacen referencia a los turnos del sistema, y los turnos con *U* se refieren a las intervenciones del simulador de usuarios. Para clarificar la explicación, se ha incorporado una frase en lenguaje natural equivalente al turno etiquetado seleccionado por el gestor de diálogo.

La Figura 8.9 muestra un ejemplo correspondiente a un diálogo con un único objetivo (conocer horarios de Valencia a Barcelona viajando el 30 de mayo).

S1: (Opening:Nil:Nil) <i>Bienvenido al sistema de información de trenes. ¿En qué puedo ayudarle?</i>
U1: (Hora-Salida) [0,9] Destino: Barcelona [0,9] Fecha-Llegada: 31-05-2007 [0,1] Hora-Llegada: 08.00-23.00 [0,9]
S2: (Confirmación:Fecha:Fecha) <i>¿Quiere salir el 30 de Mayo?</i>
U2: Negación[0,9] Fecha-Llegada: 05-30-2007 [0,9]
S3: (Respuesta:Hora-Salida:Hora-Salida) (Nueva-Consulta:Nil:Nil) <i>{Información sobre horarios} ¿Desea algo más?</i>
U3: (Negación)
S4: (Cierre:Nil:Nil) <i>Gracias por utilizar el servicio de información de trenes. Adiós.</i>

Figura 8.9: Ejemplo de diálogo con un único objetivo adquirido mediante el simulador de diálogos

El diálogo comienza con un turno de bienvenida por parte del sistema (S1). En el primer turno, el usuario simulado proporciona los atributos *Destino* y *Fecha-Salida*. Además, facilita el atributo *Hora-Llegada* (fijado como dato opcional en el escenario).

El simulador de errores introduce en este primer turno un valor erróneo en el atributo *Fecha-Salida* (cambia el día 30 por 31) y asigna medidas de confianza a los diferentes valores. En este caso, se asigna una confianza baja al valor erróneo y confianza alta a los atributos correctos.

En el segundo turno del sistema, se selecciona una confirmación para el atributo *Fecha-Salida*. Teniendo en cuenta la información definida en el objetivo del escenario, el simulador del usuario selecciona el acto de diálogo del *Negación* y proporciona el valor correcto para la *Fecha-Salida* siguiendo el objetivo (U2). En este turno, el simulador de errores asigna un valor de confianza alto a la información proporcionada por el usuario. En el siguiente turno del sistema (S3), el gestor de diálogo selecciona realizar una consulta sobre horarios a la base de datos. Dado que está disponible la información necesaria, el módulo de consulta a la base de datos realiza correctamente la consulta y el gestor de diálogo proporciona la información definida como objetivo para el diálogo. Disponiendo de esta información, el simulador de usuarios selecciona una petición para cerrar el diálogo en el turno siguiente (U3).

La Figura 8.10 muestra un ejemplo de un diálogo correspondiente a un escenario con dos objetivos (saber horarios y precios de Valencia a Barcelona para el 30 de mayo).

S1: (Apertura:Nil:Nil)
Bienvenido al sistema de información de trenes. ¿En qué puedo ayudarle?

U1: (Hora-Salida) [0,9]
 Origen: Madrid [0,9]
 Destino: Barcelona [0,9]
 Fecha-salida: 30-05-2007 [0,9]
 Tipo-Tren: Euromed [0,9]

S2: (Confirmación:Hora-Salida:Nil)(Confirmación: Origen:Origen)
¿Desea horarios saliendo desde Madrid?

U2: (Negación) [0,9]
 Origen: Valencia [0,9]

S3: (Respuesta:Hora-Salida:Hora-Salida)(Nueva-Consulta:Nil:Nil)
{Información sobre horarios} ¿Desea algo más?

U3: (Precio) [0,1]
 Origen: Valencia [[0,9]
 Destino: Barcelona [0,9]
 Hora-Salida: 08.00-23.00 [0,9]

S4: (Confirmación:Precio:Nil)(Confirmación:Origen:Origen)
¿Desea conocer precios saliendo desde Valencia?

U4: (Afirmación) [0,9]

S5: (Confirmación:Fecha:Fecha)

U5: (Afirmación) [0,9]

S6: (Respuesta:Precio:Precio)(Nueva-consulta:Nil:Nil)
{Información sobre precios} ¿Desea algo más?

U6: (Negación)

S7: (Cierre:Nil:Nil)
Gracias por utilizar el servicio de información de trenes. Adiós.

Figura 8.10: Ejemplo de diálogo con dos objetivos adquirido mediante el simulador de diálogos

En el primer turno, el simulador de usuarios selecciona realizar una consulta sobre horarios y proporciona los atributos *Origen*, *Destino* y *Fecha-Salida*. Además proporciona, como información adicional definida en el escenario, el atributo *Tipo-Tren*). En este turno,

el simulador de errores introduce un error en el atributo *Origen* (substituye Valencia por Madrid) y asigna a este valor una confianza alta.

En el siguiente turno de sistema (S2), el gestor de diálogo solicita al usuario simulado si desea consultar horarios partiendo de Madrid. En el U2, el simulador de usuarios consulta el objetivo del escenario y proporciona otra vez el atributo *Origen*. Tras este turno, el sistema realiza una consulta sobre horarios a la base de datos (S3).

El simulador de usuarios verifica en el turno U3 que el objetivo del usuario no se ha completado todavía. En este turno, selecciona realizar una consulta sobre precios, proporcionando de nuevo los atributos *Origen* y *Destino*. Además, facilita el atributo *Hora-Salida* como información adicional. En el siguiente turno del sistema (S4), el gestor de diálogo solicita al usuario si desea información sobre precios saliendo desde Valencia. Verificado el objetivo del diálogo, el simulador de usuarios selecciona *Afirmación* (U4).

En el siguiente turno, el sistema selecciona confirmar el atributo *Fecha-Salida*. El simulador de usuarios confirma esta información conforme el objetivo del diálogo (U5). Seguidamente, el sistema selecciona realizar una consulta a la base de datos sobre precios (S6). Dado que está disponible la información necesaria, el módulo de consulta a la base de datos realiza la consulta correctamente y el gestor de diálogo completa los objetivos establecidos para el diálogo. Disponiendo de esta información, el simulador de usuarios selecciona una petición de cierre del diálogo en el turno siguiente (U6).

8.3.3. Evaluación del corpus simulado

Para realizar la evaluación de la metodología de simulación de diálogos, se aprendió un gestor de diálogo estadístico a partir del corpus simulado aplicando la técnica de simulación descrita (Gestor M1). Para evaluar este gestor se utilizó el corpus DIHANA, de forma que pudiese evaluarse el comportamiento de este gestor con un corpus de diálogo adquirido con usuarios reales.

Además, se llevo a cabo el aprendizaje de otro gestor de diálogo utilizando el corpus DIHANA como partición de aprendizaje (Gestor M2). Para evaluar este gestor, tal y como se ha explicado en la Sección 7.3, se utilizó un proceso de validación definiendo las cinco particiones descritas en dicha sección. De este modo, la totalidad del corpus DIHANA se utiliza para evaluar los gestores de diálogo M1 y M2.

La evaluación de los gestores de diálogo se llevo a cabo mediante las medidas definidas previamente: $\%estrategia$, $\%correcta$ y $\%error$.

La Tabla 8.5 muestra los resultados obtenidos para las diferentes medidas tras de la evaluación descrita.

	Gestor M1	Gestor M2
$\%estrategia$	54,57 %	97,34 %
$\%correcta$	88,83 %	99,33 %
$\%error$	11,17 %	0,67 %

Tabla 8.5: Resultado de la evaluación de los gestores de diálogo M1 y M2

De la observación de los resultados cabe destacar que el gestor proporciona un 88,83 % de respuestas coherentes con el estado actual del diálogo ($\%correcta$). Este resultado

posibilita que el modelo de diálogo aprendido pueda utilizarse para el desarrollo de un gestor de diálogo inicial aprendido con un esfuerzo reducido y capaz de proporcionar muy buenas prestaciones.

Utilizando el corpus DIHANA para aprender el modelo del diálogo (Gestor M2), el 97,34 % de las respuestas proporcionadas por este gestor de diálogo sigue la estrategia definida para el Mago de Oz. Con respecto al Gestor M1, únicamente el 54,57 % de las respuestas siguen esta estrategia. Por tanto, puede verse como el gestor de diálogo M1 se separa de la estrategia definida para el mago de Oz, según lo esperado.

Con respecto a la medida $\%error$, el gestor M1 proporciona un 11,17 % de respuestas que no son compatibles con el estado del diálogo. Este valor es considerable, pudiéndose reducir con la incorporación de nuevos diálogos y posterior aprendizaje del modelo.

8.4. Experimentación con las metodologías descritas

En esta sección se presentan las estadísticas de la adquisición de diversos corpus de diálogos mediante la interacción de las metodologías descritas. La Sección 8.4.1 muestra los resultados utilizando el corpus DIHANA para realizar el aprendizaje del simulador de usuarios y del gestor de diálogo, de la forma descrita en las secciones 8.2 y 6.4.

En la Sección 8.4.2 se presentan los resultados utilizando una selección aleatoria de respuestas para el simulador de usuarios (Sección 8.3) y el modelo estadístico para el gestor de diálogo.

Finalmente, las estadísticas mostradas en la Sección 8.4.3 se refieren a la interacción del simulador de usuarios mediante el modelo estadístico y la selección aleatoria de respuestas en el gestor de diálogo.

Cada una de estas secciones contiene un total de dos tablas, correspondientes a dos experimentaciones diferenciadas. En la primera de ellas, se ha utilizado la totalidad del corpus DIHANA para aprender el modelo estadístico para el simulador de usuario y/o el gestor de diálogo (modelo global). En la segunda, se han utilizado únicamente para el aprendizaje del modelo los diálogos correspondientes al objetivo (tipo de escenario) que se desea simular. De esta forma, se han generado modelos específicos para tratar la casuística de escenarios definida para la tarea DIHANA. Mediante estos modelos, el gestor de diálogo puede adaptarse específicamente al objetivo del diálogo, pudiendo transitar de un modelo específico a otro conforme el módulo de comprensión le indique que la consulta del usuario ha cambiado.

Las Tablas de cada sección muestran las estadísticas del corpus adquirido tras realizar 1.000 simulaciones para cada uno de los 20 objetivos diferentes definidos en la tarea DIHANA (Figura 8.3). Se han adquirido un total 40.000 diálogos en cada una de las secciones siguientes, 20.000 mediante el modelo global y 20.000 utilizando modelos específicos. Las estadísticas obtenidas se muestran a continuación.

8.4.1. Modelo estadístico tanto en el simulador de usuario como en el gestor de diálogo

Las Tablas 8.6 y 8.7 muestran los resultados de la interacción del simulador de usuarios y el gestor de diálogo, modelando ambos estadísticamente siguiendo las metodologías

expuestas en las secciones 6.4 y 8.2. Los resultados de la Tabla 8.6 se han obtenido aprendiendo ambos modelos con la totalidad del corpus DIHANA, mientras que en la experimentación mostrada en la Tabla 8.7 se han aprendido modelos específicos utilizando únicamente en el aprendizaje los diálogos del corpus que se corresponden con el objetivo de los diálogos a simular.

En ambas tablas se observa la dificultad de simular diálogos con dos objetivos, especialmente para los tipos A2, A6 y A8. Mediante la utilización de un único modelo general se observa que en general se obtiene un mayor número de diálogos exitosos utilizando los modelos específicos, dado que estos modelos se han entrenado con muestras procedentes de diálogos con el mismo objetivo que se desea simular. No obstante, el número de diálogos exitosos distintos que se obtiene utilizando un único modelo general es mayor que para el caso de los modelos específicos, ya que el modelo general posibilita que el gestor explore alternativas que permitan alcanzar el objetivo fijado para el diálogo y que no están contempladas en los diálogos correspondientes del corpus.

	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9
Nº diálogos correctos	771	547	25	475	493	562	25	459	28	487
Media turnos	4,3	5,2	5,4	5,4	5,5	5,3	5,8	5,4	6,2	5,6
Nº diálogos distintos	489	500	24	438	476	493	25	437	28	466
Nº repeticiones diálogo más visto	70	10	2	8	5	14	1	7	1	6
Nº turnos diálogo más visto	2	4	3	2	2	7	8	7	8	2
Nº turnos diálogo más corto	2	2	3	2	2	2	3	2	3	2
Nº turnos diálogo más largo	8	8	8	8	8	8	8	8	8	8
Nº diálogos iguales corpus	19	16	0	11	11	6	0	14	0	12
	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9
Nº diálogos correctos	290	215	185	167	210	184	194	247	273	263
Media turnos	5,6	6,2	6,2	6,2	6,3	6,0	6,3	5,6	5,6	5,8
Nº diálogos distintos	259	213	182	167	208	184	193	244	267	259
Nº repeticiones diálogo más visto	9	1	2	1	2	1	2	2	5	2
Nº turnos diálogo más visto	3	3	4	4	4	8	3	3	2	5
Nº turnos diálogo más corto	3	3	3	3	3	3	3	2	2	2
Nº turnos diálogo más largo	8	8	8	8	8	8	8	8	8	8
Nº diálogos iguales corpus	5	1	3	3	2	3	3	3	5	5

Tabla 8.6: Modelo estadístico en el simulador de usuario y en el gestor de diálogo. Modelo general

	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9
Nº diálogos correctos	831	484	96	279	353	749	403	760	391	636
Media turnos	3,7	4,6	5,4	3,8	4,3	3,8	5,2	4,7	4,9	4,3
Nº diálogos distintos	277	353	69	180	266	460	291	456	326	402
Nº repeticiones diálogo más visto	124	18	7	23	10	31	21	35	11	54
Nº turnos diálogo más visto	2	2	4	7	3	4	8	2	3	4
Nº turnos diálogo más corto	2	2	4	2	2	3	3	2	3	2
Nº turnos diálogo más largo	8	8	8	8	8	8	8	8	8	8
Nº diálogos iguales modelo	6	1	0	2	2	2	1	3	1	1
Nº diálogos iguales corpus	14	9	1	9	5	13	2	11	2	10
	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9
Nº diálogos correctos	403	195	120	207	429	357	291	445	627	664
Media turnos	4,9	5,6	5,5	5,9	5,3	5,7	5,8	4,9	4,4	4,5
Nº diálogos distintos	252	178	111	187	389	336	278	318	438	547
Nº repeticiones diálogo más visto	51	4	4	4	6	3	4	23	19	18
Nº turnos diálogo más visto	3	3	4	4	3	3	3	8	2	2
Nº turnos diálogo más corto	3	3	3	3	3	3	3	2	2	2
Nº turnos diálogo más largo	8	8	8	8	8	8	8	8	8	8
Nº diálogos iguales modelo	3	2	1	0	4	2	1	1	5	1
Nº diálogos iguales corpus	6	4	3	0	5	5	3	3	8	7

Tabla 8.7: Modelo estadístico en el simulador de usuario y en el gestor de diálogo. Modelos específicos

8.4.2. Selección aleatoria de respuestas en el simulador de usuarios y modelo estadístico en el gestor de diálogo

Las Tablas 8.8 y 8.9 muestran los resultados de la interacción del simulador de usuarios, utilizando la selección aleatoria de respuestas, y del modelo estadístico para el gestor de diálogo. Lógicamente, la no utilización del modelo estadístico para seleccionar las respuestas de usuario provoca que el número de diálogos exitosos sea menor que los obtenidos en el apartado anterior. Cabe destacar que la gran mayoría de los diálogos exitosos obtenidos para cada objetivo son distintos.

	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9
Nº diálogos correctos	378	324	20	315	280	316	19	313	12	258
Media turnos	4,6	5,1	5,6	5,2	5,6	5,1	5,9	5,3	5,5	5,8
Nº diálogos distintos	345	314	20	306	280	305	19	309	12	258
Nº repeticiones diálogo más visto	8	4	1	4	1	5	1	3	1	4
Nº turnos diálogo más visto	4	2	4	3	3	2	7	2	5	6
Nº turnos diálogo más corto	2	2	4	2	2	2	4	2	3	2
Nº turnos diálogo más largo	8	8	8	8	8	8	8	8	8	8
Nº diálogos iguales corpus	8	1	0	2	0	1	0	3	0	0
	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9
Nº diálogos correctos	168	162	136	120	159	126	146	208	169	218
Media turnos	6,0	6,0	6,0	6,2	6,1	5,9	6,2	5,4	5,4	5,8
Nº diálogos distintos	168	162	136	120	159	126	146	204	167	215
Nº repeticiones diálogo más visto	1	1	1	1	1	1	1	3	3	3
Nº turnos diálogo más visto	8	8	7	5	7	5	5	2	2	2
Nº turnos diálogo más corto	3	3	3	3	3	3	3	2	2	2
Nº turnos diálogo más largo	8	8	8	8	8	8	8	8	8	8
Nº diálogos iguales corpus	1	0	0	0	0	0	0	2	2	1

Tabla 8.8: Selección aleatoria de respuestas en el simulador de usuarios y modelo estadístico en el gestor de diálogo. Modelo general

	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9
Nº diálogos correctos	304	231	9	217	166	199	26	165	49	124
Media turnos	4,3	4,7	5,3	4,6	4,9	4,3	5,4	4,8	6,1	4,9
Nº diálogos distintos	273	224	9	210	165	192	26	162	49	121
Nº repeticiones diálogo más visto	8	3	1	5	2	3	1	2	1	4
Nº turnos diálogo más visto	4	2	6	2	3	2	8	2	4	2
Nº turnos diálogo más corto	2	2	3	2	2	2	4	2	3	2
Nº turnos diálogo más largo	8	8	7	8	8	8	8	8	8	8
Nº diálogos iguales modelo	2	0	0	0	0	0	0	0	0	0
Nº diálogos iguales corpus	6	0	0	1	1	0	0	1	0	0
	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9
Nº diálogos correctos	150	95	78	92	138	134	98	138	142	145
Media turnos	5,7	5,8	5,9	5,9	6,0	6,2	6,0	5,5	5,1	5,3
Nº diálogos distintos	150	95	78	92	138	134	98	137	141	145
Nº repeticiones diálogo más visto	1	1	1	1	1	1	1	2	2	1
Nº turnos diálogo más visto	8	5	5	6	7	3	6	2	2	6
Nº turnos diálogo más corto	3	3	3	3	3	3	3	2	2	2
Nº turnos diálogo más largo	8	8	8	8	8	8	8	8	8	8
Nº diálogos iguales modelo	0	0	0	0	0	0	0	0	1	0
Nº diálogos iguales corpus	1	0	0	0	0	0	0	0	2	0

Tabla 8.9: Selección aleatoria de respuestas en el simulador de usuarios y modelo estadístico en el gestor de diálogo. Modelos específicos

8.4.3. Modelo estadístico en el simulador de usuarios y selección aleatoria de respuestas en el gestor de diálogo

En las Tablas 8.10 y 8.11 se muestran las estadísticas de los diálogos simulados utilizando el modelo estadístico en el simulador de usuarios y la selección aleatoria de respuestas en el gestor de diálogo. De la comparativa con los resultados mostrados en el apartado anterior, se observa la mayor dificultad de obtener diálogos que alcancen el objetivo. Este comportamiento viene motivado por la mayor dependencia del éxito del diálogo frente a la respuesta que genere el sistema.

	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9
Nº diálogos correctos	58	35	2	35	32	34	1	26	1	26
Media turnos	3,3	4,0	0	3,7	3,7	3,7	5,0	3,8	5,5	4,0
Nº diálogos distintos	50	34	0	34	32	32	1	25	1	25
Nº repeticiones diálogo más visto	4	2	1	2	1	2	1	2	1	2
Nº turnos diálogo más visto	4	2	4	3	3	2	5	2	5	6
Nº turnos diálogo más corto	2	2	4	2	2	2	5	2	3	2
Nº turnos diálogo más largo	8	8	8	8	8	8	5	8	8	8
Nº diálogos iguales corpus	8	1	0	2	0	1	0	3	0	0
	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9
Nº diálogos correctos	1	1	0	0	0	1	0	6	7	6
Media turnos	8,0	5,0	0	0	0	7,0	0	4,3	4,7	4,5
Nº diálogos distintos	1	1	0	0	0	1	0	6	7	6
Nº repeticiones diálogo más visto	1	1	0	0	0	1	0	1	1	1
Nº turnos diálogo más visto	8	5	0	0	0	7	0	2	2	2
Nº turnos diálogo más corto	8	5	0	0	0	7	0	2	2	2
Nº turnos diálogo más largo	8	5	0	0	0	7	0	8	8	8
Nº diálogos iguales corpus	1	0	0	0	0	0	0	2	2	1

Tabla 8.10: Modelo estadístico en el simulador de usuarios y selección aleatoria de respuestas en el gestor de diálogo. Modelo general

	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9
Nº diálogos correctos	304	231	9	217	166	199	26	165	49	124
Media turnos	4,3	4,7	5,3	4,6	4,9	4,3	5,4	4,8	6,1	4,9
Nº diálogos distintos	273	224	9	210	165	192	26	162	49	121
Nº repeticiones diálogo más visto	8	3	1	5	2	3	1	2	1	4
Nº turnos diálogo más visto	4	2	6	2	3	2	8	2	4	2
Nº turnos diálogo más corto	2	2	3	2	2	2	4	2	3	2
Nº turnos diálogo más largo	8	8	7	8	8	8	8	8	8	8
Nº diálogos iguales modelo	2	0	0	0	0	0	0	0	0	0
Nº diálogos iguales corpus	6	0	0	1	1	0	0	1	0	0
	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9
Nº diálogos correctos	150	95	78	92	138	134	98	138	142	145
Media turnos	5,7	5,8	5,9	5,9	6,0	6,2	6,0	5,5	5,1	5,3
Nº diálogos distintos	150	95	78	92	138	134	98	137	141	145
Nº repeticiones diálogo más visto	1	1	1	1	1	1	1	2	2	1
Nº turnos diálogo más visto	8	5	5	6	7	3	6	2	2	6
Nº turnos diálogo más corto	3	3	3	3	3	3	3	2	2	2
Nº turnos diálogo más largo	8	8	8	8	8	8	8	8	8	8
Nº diálogos iguales modelo	0	0	0	0	0	0	0	0	1	0
Nº diálogos iguales corpus	1	0	0	0	0	0	0	0	2	0

Tabla 8.11: Modelo estadístico en el simulador de usuarios y selección aleatoria de respuestas en el gestor de diálogo. Modelos específicos

Capítulo 9

Adaptación del gestor del diálogo al usuario y a nuevas tareas

9.1. Introducción

Tal y como se ha visto en el capítulo de introducción de la tesis, una temática de especial interés actualmente es el desarrollo de sistemas de diálogo capaces de interactuar en diferentes dominios, diferentes entornos o con la posibilidad de adaptarse a los diferentes perfiles o preferencias del usuario. La robustez y la portabilidad de los sistemas de diálogo son necesarias para garantizar que el sistema sea flexible ante variaciones en la manera con la que los usuarios interactúan con el sistema.

La utilización de técnicas estadísticas aprendidas a partir de un corpus de datos facilita la adaptación del sistema a nuevas tareas desde el punto de vista que, manteniéndose las mismas funcionalidades para el sistema, el proceso de adaptación consiste básicamente en el aprendizaje del nuevo modelo de diálogo siguiendo la metodología propuesta. No obstante, es necesario definir la semántica de la nueva tarea y disponer de un corpus de diálogos etiquetado conforme a dicha semántica, a partir del cual pueda realizarse el aprendizaje.

En este capítulo se expone el trabajo que se ha llevado a cabo para adaptar la metodología de gestión de diálogo estadística propuesta en la tesis en el ámbito de una nueva tarea. Este trabajo se ha desarrollado en el marco del proyecto de investigación nacional EDECÁN y ha posibilitado las siguientes publicaciones (Hurtado et al., 2007b) (García et al., 2007) (Griol et al., 2007b) (Griol et al., 2007). La principal dificultad del proceso de adaptación radica en la incorporación de nuevas funcionalidades al sistema de diálogo, que hacen necesario que se tenga en cuenta el resultado de las consultas al módulo gestor de la aplicación, tal y como se expone a lo largo del capítulo.

9.2. El proyecto EDECÁN

El proyecto EDECÁN (financiado por el Ministerio de Educación y Ciencia TIN2005-08660-C04) tiene como objetivo aumentar la robustez de un sistema de diálogo de habla espontánea a través del desarrollo de tecnologías para la adaptación y personalización del mismo a los distintos contextos acústicos y de aplicación en los que pueda encontrarse. El

concepto de contexto acústico abarca todos los elementos, dependientes tanto del usuario como del entorno físico que lo rodea, que influyen en la señal que capta el/los micrófono/s que conforma/n la entrada del sistema de diálogo. Por otro lado, el contexto de aplicación hace referencia a la estructura semántica de los dominios en los que se desarrolla el diálogo.

En el proyecto, que tiene como duración de diciembre de 2005 a noviembre del 2008, participan el Grupo de Tecnologías de las Comunicaciones (GTC) del Instituto de Investigación en Ingeniería de Aragón (I3A) de la Universidad de Zaragoza, el Grupo de Reconocimiento de Formas y Tecnologías del Habla (GRFTH) del Departamento de Electricidad y Electrónica (DEE) de la Universidad del País Vasco/Euskal Herriko Unibertsitatea (EHU), el Grupo de Tecnología del Habla (GTH) del Departamento de Ingeniería Electrónica (DIE) de la Universidad Politécnica de Madrid (UPM) y el Grupo de Reconocimiento de Formas, Inteligencia Artificial (GRFIA) del Departamento de Sistemas Informáticos y Computación (DSIC) de la Universitat Politècnica de Valencia (UPV).

EDECÁN se subdivide en 4 subproyectos que participan de los objetivos tecnológicos del proyecto coordinado, en los apartados de adquisición de corpus, en la definición de la arquitectura del sistema de diálogo, en el desarrollo de la plataforma de trabajo y construcción de un prototipo que incorpore los resultados del proyecto, y en la evaluación y usabilidad de un sistema de diálogo multidominio, definiendo medidas y escenarios para analizar la usabilidad en entornos multidominio y la mejora obtenida al incorporar tecnologías de adaptación, personalización e información multimodal.

Los objetivos científicos definidos para el proyecto EDECÁN son:

- Optimización y mejora de los componentes de los sistemas de diálogo disponibles:
 - Mejora de la robustez y prestaciones de los sistemas de diálogo disponibles en cada grupo.
 - Diseño e implementación de nuevos componentes (generación de habla y síntesis con emociones e integración básica de información multimodal).
 - Búsqueda de mecanismos/técnicas eficientes que mantengan las prestaciones de los sistemas frente a cambios en el contexto acústico (en general: físico, del locutor, etc.).
 - Desarrollo de técnicas para la caracterización e identificación del contexto acústico.
 - Desarrollo de técnicas para mejorar la robustez de las fuentes de conocimiento de los sistemas de reconocimiento (modelos acústicos, modelos de lenguaje, múltiples pronunciaciones, etc.) frente a los cambios de contexto acústico.
 - Búsqueda de mecanismos/técnicas eficientes que minimicen el esfuerzo invertido para generar un nuevo sistema de diálogo.
 - Aprendizaje automático (uso de métodos supervisados y no supervisados) de modelos de lenguaje y de modelos de comprensión del habla.
 - Desarrollo de técnicas de adaptación a nuevas tareas.
-

- Estudio de metodologías de aprendizaje de la estrategia del diálogo para las diferentes tareas.
- Búsqueda de mecanismos eficientes para que los sistemas exploten la información disponible en un perfil de usuario, con el objetivo de adaptar el comportamiento del sistema a las características, capacidades y preferencias de los usuarios.
- Diseño de perfiles de usuario: elementos relevantes en aplicaciones de diálogo y comprensión.
- Estudio de las relaciones relevantes entre los elementos disponibles en el perfil de usuario y los distintos módulos del sistema de diálogo.
- Estudio de mecanismos contrastados para llevar a cabo la evaluación de las prestaciones y de la usabilidad.
 - Búsqueda de los parámetros adecuados para la evaluación de las tecnologías desarrolladas.
 - Estudio de metodologías de aprendizaje de la estrategia del diálogo para las diferentes tareas.
 - Evaluación comparativa de las estrategias de diseño y adaptación de los distintos componentes de los sistemas de diálogo y análisis de los resultados.

Como objetivos tecnológicos, se han definido:

- Sobre el corpus:
 - Adquisición y transcripción de un corpus específico para la investigación en la adaptación y personalización al contexto acústico y de aplicación.
 - Etiquetado del corpus anterior en términos de los elementos condicionantes del contexto acústico y de aplicación.
 - Sobre el prototipo:
 - Desarrollo de una plataforma para la integración de las diferentes unidades de los sistemas de diálogo.
 - Construcción de un prototipo de sistema de diálogo que integre los resultados científicos y tecnológicos sobre dos contextos de aplicación definidos por un servicio de información del transporte y un servicio de asistente en el hogar y en el coche.
-

9.3. La tarea EDECÁN

Una de las tareas que se han definido en el contexto del proyecto EDECÁN consiste en el diseño de un interfaz oral para informar y permitir las reservas de instalaciones deportivas en nuestra universidad. La principal diferencia entre esta tarea y la definida para el proyecto DIHANA radica en la incorporación de nuevas funcionalidades en el sistema que hacen necesario tener en cuenta el resultado de la consulta proporcionado por el gestor de la aplicación. En el dominio del sistema de diálogo desarrollado para DIHANA se proporcionaba únicamente información relativa a las consultas requeridas por el usuario, no modificándose en ningún instante la información almacenada en la base de datos del sistema. En la tarea EDECÁN se incorporan nuevas operaciones que suponen la modificación de la información almacenada en las bases de datos de la aplicación, por ejemplo, tras la reserva de una pista deportiva.

El módulo definido en la arquitectura del sistema EDECÁN para gestionar la información referente a la aplicación, que se ha denominado *Gestor de la Aplicación (Application Manager, AM)*, realiza dos funciones fundamentales. En primer lugar, gestiona las operaciones a la base de datos de la aplicación. En segundo lugar, verifica que las operaciones requeridas por el usuario cumplen la normativa definida por la universidad para la gestión de las pistas deportivas (por ejemplo: un usuario no puede reservar más de una pista deportiva al día, un usuario sancionado no puede realizar reservas, etc.).

De este modo, el resultado proporcionado por el AM debe tenerse en cuenta para generar la siguiente respuesta del sistema. Por ejemplo, a la hora de reservar una pista deportiva pueden ocurrir un conjunto de situaciones:

- Tras la consulta a la base de datos se detecta que el usuario está sancionado. El sistema debe informar al usuario que no podrá reservar pistas deportivas hasta que el periodo de sanción haya finalizado.
- Tras la consulta a la base de datos se comprueba que no existen pistas que cumplan los requerimientos expuestos por el usuario. En esta situación, el sistema puede comprobar si existen pistas disponibles cambiando uno o varios de estos requerimientos (por ejemplo, la hora), informando posteriormente al usuario sobre las opciones disponibles.
- Como resultado de la consulta a la base de datos se verifica que existe una única pista que cumple los requerimientos del usuario. El sistema debe confirmar que todo es correcto para proceder finalmente con la reserva.
- Si se comprueba que hay disponibles dos o más pistas que cumplen las exigencias del usuario, el sistema debe verificar cuál de ellas desea reservarse.

Para tener en cuenta la información proporcionada por el AM para la selección de la próxima respuesta del sistema, hemos considerado que se requieren dos etapas. En la primera etapa, la información contenida en el DR y el último estado S_{i-1} se tienen en cuenta para seleccionar la mejor consulta a realizar al AM (\hat{A}_{1_i}):

$$\hat{A}_{1_i} = \operatorname{argmax}_{A_{1_i} \in \mathcal{A}_1} P(A_i | DR_{i-1}, S_{i-1})$$

donde \mathcal{A}_1 es el conjunto de posibles consultas al AM.

En la segunda fase, se genera la respuesta final del sistema (\hat{A}_{2_i}) teniendo en cuenta \hat{A}_{1_i} y la información proporcionada por el AM (AM_i):

$$\hat{A}_{2_i} = \operatorname{argmax}_{A_{2_i} \in \mathcal{A}_2} P(A_i | AM_i, A_{1_i})$$

donde \mathcal{A}_2 es el conjunto de posibles respuestas del sistema.

La Figura 9.1 muestra el esquema propuesto para el desarrollo del gestor de diálogo para el proyecto EDECÁN, detallándose las dos etapas descritas para la generación de la respuesta final del sistema.

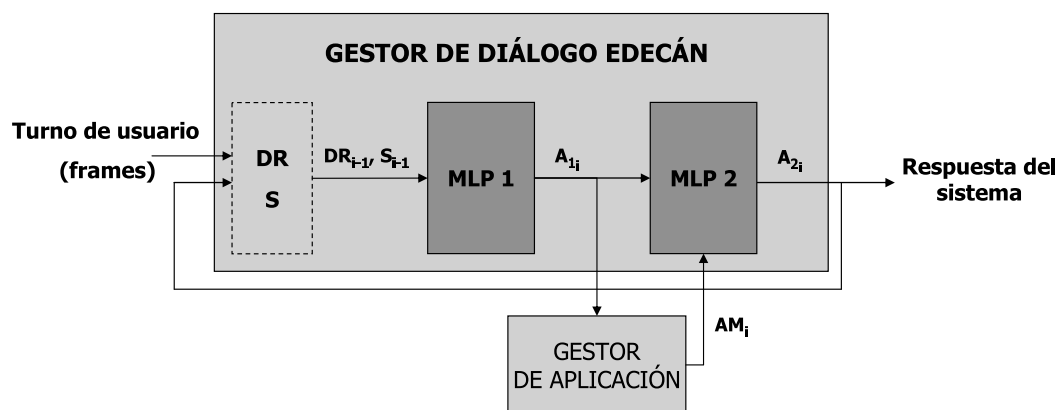


Figura 9.1: Esquema del gestor de diálogo desarrollado para el proyecto EDECÁN

9.4. Representación semántica de la tarea

La determinación de la semántica de la tarea EDECÁN se ha llevado a cabo teniendo en cuenta las diferentes funcionalidades con las que se desea dotar al sistema de reservas y la información que se requiere para completarlas. Para realizar esta definición se ha utilizado un conjunto de diálogos persona-persona proporcionados por el personal del Área de Deportes de la Universidad. De este modo, en estos diálogos han participado usuarios que deseaban realmente realizar las diferentes consultas que proporcionará el sistema automático.

Este conjunto de diálogos se ha ampliado con nuevos diálogos generados por parte del personal de nuestro grupo de investigación. Para la generación de estos diálogos, se ha llevado a cabo la simulación del comportamiento del sistema, introduciendo errores y medidas de confianza en la información suministrada por el usuario. En total se dispone de un corpus de 150 diálogos (873 turnos de usuario). La Figura 9.2 muestra un ejemplo de uno de los diálogos que conforman el corpus descrito. El conjunto de diálogos se ha etiquetado mediante una representación en forma de actos de diálogo, que definen la semántica de la tarea.

S1: Bienvenido al sistema de información del Área de Deportes. ¿En qué puedo ayudarle?
 U1: Quiero reservar una pista de pádel exterior para el miércoles que viene.
 S2: ¿A qué hora desea jugar?
 U2: A las 3 de la tarde.
 S3: El día 23 de mayo a las 15:00 dispone de las pistas 1 y 4 de pádel. ¿Cuál de ellas le reservo?
 U3: La pista 1 por favor.
 S4: Le reservo la pista 1 de pádel para el miércoles 23 de mayo a las 15:00. ¿Desea algo más?
 U4: Sí, quería liberar una pista que tenía para mañana.
 S5: ¿Desea anular la pista de pádel del día 21 de mayo a las siete de la tarde?
 U5: Sí.
 S6: Le anulo la pista solicitada. ¿Desea algo más?
 U6: No gracias.
 S7: Gracias por utilizar el sistema de información del Área de Deportes. Adiós.

Figura 9.2: Ejemplo de diálogo para la tarea EDECÁN

9.4.1. Etiquetado de los turnos de usuario

Para el caso de los turnos de usuario, los actos de diálogo se corresponden con la interpretación semántica de la intervención del usuario mediante frames (atributos y conceptos). Para la tarea EDECÁN se han definido un total de siete conceptos:

- Cuatro conceptos dependientes de la tarea, correspondientes a los tipos de consultas que puede solicitar el usuario: conocer la disponibilidad de pistas (*Availability*), realizar una reserva (*Booking*), conocer las reservas que tiene vigentes (*Booked*) o cancelar alguna de ellas (*Cancellation*).
- Tres conceptos independientes de la tarea, análogos a los utilizados en la tarea DIHANA (*Acceptance*, *Rejection* y *Not-Understood*).

Se han definido un total de seis atributos, relativos a la información que debe aportar el usuario para completar las diferentes consultas contempladas por el sistema. Los atributos definidos son el deporte que se desea practicar (*Sport*), el horario para el que se desea la consulta (*Hour*), la fecha (*Date*), el tipo de pista polideportiva (*Court-Type*), el identificador de pista (*Court-ID*) y el número de orden correspondiente a la pista que se desea reservar (*Order-Number*).

A continuación se muestra un ejemplo de la interpretación semántica de una intervención del usuario:

Turno de usuario:

Quiero reservar una pista de squash para mañana por la tarde

Interpretación semántica:

(Booking)

Sport: squash

Date: mañana

Hour: tarde

La Figura 9.3 muestra el conjunto de conceptos y atributos definidos para la representación semántica de los turnos de usuario.

Conceptos dependientes de la tarea
<i>(Availability)</i>
<i>(Booking)</i>
<i>(Booked)</i>
<i>(Cancellation)</i>
Conceptos independientes de la tarea
<i>(Acceptance)</i>
<i>(Rejection)</i>
<i>(Not-Understood)</i>
Conjunto de atributos
<i>Sport</i>
<i>Hour</i>
<i>Date</i>
<i>Court-Type</i>
<i>Court-ID</i>
<i>Order-Number</i>

Figura 9.3: Conjunto de conceptos y atributos definidos para el etiquetado de los turnos de usuario

9.4.2. Etiquetado de los turnos de sistema

El etiquetado de los turnos de sistema se ha realizado de forma similar al de las intervenciones del usuario. Los conceptos definidos pueden clasificarse igualmente en dependientes de la tarea e independientes de la misma. Se han detallado un total de 18 conceptos dependientes de la tarea:

- Conceptos utilizados para informar al usuario del resultado de una determinada consulta: sobre disponibilidad de pistas (*Availability*), sobre la realización de una reserva (*Booking*), sobre las reservas actuales del usuario (*Booked*) o sobre la anulación de una reserva (*Cancellation*).
- Conceptos definidos para requerir al usuario los atributos necesarios para una determinada consulta: deporte (*Sport*), fecha (*Date*), hora (*Hour*) y tipo de pista (*Court-Type*).
- Conceptos utilizados para la confirmación de conceptos (*Confirmation-Availability*, *Confirmation-Booking*, *Confirmation-Booked*, *Confirmation-Cancellation*) y de atributos (*Confirmation-Sport*, *Confirmation-Date*, *Confirmation-Hour*, *Confirmation-CourtType*).
- Conceptos relativos al AM: infracción de la normativa de reservas (*Rule-Info*) o indicación de la necesidad de seleccionar alguna de las pistas disponibles (*Booking-Choice*).

Se han definido un total de seis atributos, correspondientes a los cinco detallados para el etiquetado de los turnos de usuario (*Sport*, *Court-Type*, *Court-ID*, *Date*, *Hour*) y un atributo relativo al número de pistas que satisfacen los requerimientos del usuario (*Num-Courts*).

La Figura 9.4 muestra el conjunto de conceptos y atributos definidos para el etiquetado de los turnos de sistema.

Conceptos	Atributos
Consultas dependientes de la tarea <i>Availability</i> <i>Booking</i> <i>Booked</i> <i>Cancellation</i> <i>Booking-Choice</i> <i>Rule-Info</i>	<i>Sport</i> <i>Court-Type</i> <i>Court-ID</i> <i>Date</i> <i>Hour</i>
Confirmaciones conceptos y atributos <i>Confirmation-Availability</i> <i>Confirmation-Booking</i> <i>Confirmation-Booked</i> <i>Confirmation-Cancellation</i> <i>Confirmation-Sport</i> <i>Confirmation-Date</i> <i>Confirmation-Hour</i> <i>Confirmation-CourtType</i>	
Petición de atributos <i>Sport</i> <i>Date</i> <i>Hour</i> <i>Court-Type</i>	
Conceptos no dependientes de la tarea <i>Opening</i> <i>Closing</i> <i>New-Query</i> <i>Undefined</i>	

Figura 9.4: Conjunto de conceptos y atributos definidos para el etiquetado de los turnos de sistema

Seguidamente se muestra un ejemplo del etiquetado de una respuesta del sistema:

Turno de Sistema:

¿Le reservo la pista de squash 1 del pabellón para el 25 de junio de 20:00 a 20:30?

Etiquetado:

(Confirmation-Booking)

Sport: squash

Date: 25-06-2007

Hour: 20:00-20:30

Court-Type: pabellón

Court-ID:1

9.5. Representación de las fuentes de información

La representación definida para el par de entrada (DR, S) es la siguiente:

- La codificación de los actos de diálogos correspondientes a la última respuesta generada por el sistema (A_i): Esta información se modela mediante una variable, que posee tantos bits como posibles respuestas diferentes se han detallado para el sistema.

$$\vec{x}_1 = (x_{11}, x_{12}, x_{13}, \dots, x_{122}) \in \{0, 1\}^{22}$$

- Registro del diálogo (DR): El DR definido para la tarea EDECÁN almacena un total de diez características, correspondientes a los cuatro conceptos y seis atributos dependientes de la tarea que se han detallado para realizar el etiquetado de las intervenciones del usuario (Figura 9.5). Análogamente a la tarea DIHANA, cada una de estas características pueden tomar los valores $\{0, 1, 2\}$. De este modo, cada uno de los conceptos y atributos del DR puede modelarse utilizando una variable con tres bits.

$$\vec{x}_i = (x_{i1}, x_{i2}, x_{i3}) \in \{0, 1\}^3 \quad i = 2, \dots, 11$$

Conceptos	Atributos
<i>Availability</i>	<i>Sport</i>
<i>Booking</i>	<i>Court-Type</i>
<i>Booked</i>	<i>Court-ID</i>
<i>Cancellation</i>	<i>Date</i>
	<i>Hour</i>
	<i>Order-Number</i>

Figura 9.5: Registro del diálogo definido para la tarea EDECÁN

- Información independiente de la tarea (actos de diálogo *Acceptance*, *Rejection* y *Not-Understood*): Estos tres actos de diálogo se han codificado de forma idéntica a las características almacenadas en el DR . De esta forma, cada uno de estos tres actos de diálogo puede tomar los valores $\{0, 1, 2\}$ y modelarse utilizando una variable con tres bits.

$$\vec{x}_i = (x_{i1}, x_{i2}, x_{i3}) \in \{0, 1\}^3 \quad i = 11, \dots, 13$$

De este modo, la variable (DR, S) puede representarse mediante el vector de 13 características:

$$(DR, S) = (\vec{x}_1, \vec{x}_2, \vec{x}_3, \dots, \vec{x}_{13})$$

La respuesta generada por el AM se ha codificado teniendo en cuenta el conjunto de posibles respuestas existentes en el corpus tras llevar a cabo una consulta al AM. Este conjunto engloba las diferentes situaciones que puede comportar una consulta al AM desarrollado para EDECÁN y contempladas en el corpus persona-persona:

- Caso 1: El AM no ha intervenido en la generación de la respuesta final del sistema, por ejemplo, cuando se selecciona la confirmación de un atributo, la determinación del cierre del diálogo, etc.
- Casos 2-4: Tras una consulta a la base de datos, el AM proporciona como respuesta que no existen pistas que cumplan los requerimientos del usuario (caso 2), existe una única pista (caso 3) o existe más de una pista (caso 4).
- Caso 5: El AM advierte que la operación del usuario no puede efectuarse por incumplir la normativa establecida en la Universidad.

De este modo, la respuesta generada por el AM se ha modelado con una variable de tres bits, con las que se representan estas cinco situaciones:

$$AM = (x_1, x_2, x_3) \in \{0, 1\}^3$$

9.6. Gestor de diálogo desarrollado

A partir del etiquetado del corpus de diálogos persona-persona, y aplicando la adaptación expuesta en este capítulo, se ha desarrollado un gestor de diálogo en el contexto del proyecto EDECÁN.

Para realizar el entrenamiento de los MLP, se utilizó el software **April** desarrollado en nuestro grupo de investigación. Se extrajo un subconjunto de validación (20 %) de cada uno de los conjuntos de test. Los MLP se entrenaron utilizando el algoritmo de Backpropagation con momentum. La mejor topología fue dos capas ocultas con 100 y 10 neuronas respectivamente.

La evaluación se llevó a cabo mediante un proceso de validación cruzada. El corpus se dividió aleatoriamente en cinco subconjuntos. Cada evaluación, de este modo, consistió en cinco experimentaciones. En cada una de ellas se utilizó un subconjunto diferente de los cinco definidos como muestras de evaluación, y el 80 % del corpus restante se utilizó como partición de entrenamiento. Para evaluar el funcionamiento del gestor desarrollado se han utilizado las tres medidas definidas para la evaluación del gestor de diálogo estadístico en DIHANA:

- Porcentaje de respuestas que coinciden con la respuesta de referencia anotada en el corpus (*%exacta*).

- Porcentaje de respuestas que son coherentes con el estado actual del diálogo (*%correcta*).
- Porcentaje de respuestas que no son compatibles con el estado actual del diálogo (*%error*), provocando el fallo del diálogo.

Estas dos últimas medidas se han obtenido tras una revisión manual de las respuestas proporcionadas por el gestor. La Tabla 9.1 muestra los resultados obtenidos de la evaluación del gestor.

<i>%exacta</i>	72,9 %
<i>%correcta</i>	86,7 %
<i>%error</i>	4,5 %

Tabla 9.1: Resultados de la evaluación del gestor de diálogo desarrollado para el proyecto EDECÁN

De los resultados obtenidos tras la experimentación realizada puede verse que el gestor de diálogo se adapta correctamente a los requerimientos de la nueva tarea, proporcionando un 86,7% de respuestas que son coherentes con el estado actual del diálogo, coincidiendo un 72,9 % con la respuesta de referencia anotada en el corpus.

El porcentaje de respuestas proporcionadas por el gestor que puede causar el fallo del diálogo es considerable (4.5%). Asimismo, existe un 8.8 % de respuestas que suponen que el diálogo pueda continuar, pero no son coherentes con el estado actual del diálogo (como por ejemplo, solicitar información de la que ya se dispone actualmente). Mediante la ampliación del corpus inicial de diálogos se espera poder reducir ambos porcentajes. Para ello, se están utilizando las técnicas de simulación automática de diálogos propuestas en la tesis. Además, utilizando el gestor de diálogo propuesto conjuntamente con el resto de módulos del sistema puede realizarse la adquisición de un corpus de manera supervisada. Con la incorporación de estos diálogos adquiridos podrá realizarse la mejora del modelo de diálogo inicial.

Capítulo 10

Conclusiones

A lo largo de esta tesis se han descrito una serie de trabajos que han dado lugar al desarrollo de diferentes módulos de un sistema de diálogo para el proyecto DIHANA. La integración de estos trabajos ha posibilitado que actualmente exista un sistema de diálogo completo capaz de interactuar en el dominio de la tarea definida (acceder a un sistema de información de trayectos en tren de recorrido nacional).

El capítulo primero de la tesis ha expuesto el contexto en el que se enmarca la tesis, definiendo qué se entiende por sistema de diálogo hablado, qué componentes fundamentales lo constituyen y cuáles son las características principales de los mismos. Sin duda, el área de investigación de los sistemas de diálogo es muy compleja, englobando un gran número de disciplinas y temáticas que deben coordinarse para posibilitar que el sistema reconozca las palabras pronunciadas, comprenda su significado, gestione el diálogo (incluyendo la información contextual, manejo de errores y acceso a la aplicación final) y genere la respuesta oral.

El segundo capítulo muestra con mayor precisión el marco de la tesis, mostrando un estado del arte detallado del conjunto de metodologías más relevantes propuestas por los diferentes grupos de investigación para desarrollar gestores de diálogo, prestando una especial atención a las metodologías basadas en el aprendizaje de un modelo a partir de un corpus de datos. Asimismo, se concretan las metodologías utilizadas para el desarrollo y evaluación de sistemas de diálogo, describiendo las técnicas existentes para la implementación de simuladores de usuario. Finalmente, se citan métodos propuestos por diferentes autores para facilitar la adaptación de los sistemas de diálogo a tareas diferentes a las definidas inicialmente.

El orden utilizado en el resto de capítulos para detallar la estructura de los diferentes trabajos que conforman la tesis coincide con el seguido temporalmente para posibilitar su consecución. En primer lugar se ha descrito la tarea DIHANA. En el capítulo tercero de la tesis se precisa el conjunto de consultas que se definieron para el sistema, así como la información que el sistema debía tener en cuenta para llevar a cabo dichas consultas. Este proceso es fundamental para garantizar el diseño correcto del sistema. En DIHANA, este diseño se ha llevado a cabo mediante la adquisición de un corpus de diálogos mediante la técnica del Mago de Oz. La utilización de esta técnica hizo necesario el desarrollo de un conjunto de escenarios, cada uno con unos objetivos predefinidos, y la definición de una estrategia para la interacción del Mago con el usuario.

El elemento fundamental que rige dicha estrategia es la utilización de un registro del

diálogo, en el que se almacenan el conjunto de atributos y conceptos detectados por el módulo de comprensión a lo largo del diálogo, así como las medidas de confianza generadas por el reconocedor y el módulo de comprensión para cada uno de los datos anteriores. La utilización de este registro es una de las bases fundamentales en las que se apoyan las diferentes metodologías propuestas en la tesis para gestionar el diálogo.

Por último, en el capítulo tercero, se describe el proceso de etiquetado del corpus DIHANA. Este etiquetado se llevó a cabo en términos de actos de diálogos. Los actos de diálogos de usuario se corresponden con la interpretación semántica de la intervención del usuario mediante frames. El etiquetado de actos de diálogo se definió a tres niveles, que representan respectivamente intencionalidad del segmento, el conjunto de frames mencionados en el turno y los valores concretos de los atributos presentes en el mismo. El trabajo presentado en este capítulo ha posibilitado que se disponga actualmente de un gran corpus de diálogos, que aumenta las prestaciones del corpus adquirido en el proyecto BASURDE, predecesor de DIHANA, en cuanto a número de diálogos, usuarios participantes en la adquisición, vocabulario, etc.

De la evaluación del proceso de adquisición recibida por parte de los usuarios que participaron en la misma, se puede concluir el correcto funcionamiento del proceso, en cuanto a la satisfacción de los usuarios con el ritmo de interacción con el sistema y la consecución de los objetivos definidos para los distintos escenarios. Cabe destacar que gran parte de los usuarios entendieron que estaban utilizando un sistema completamente automático.

El capítulo cuarto ha descrito el sistema de diálogo desarrollado para DIHANA. Los módulos más relevantes del sistema se rigen por modelos aprendidos automáticamente a partir del corpus DIHANA. La estructura definida para el sistema se basa en una arquitectura en estrella, existiendo un módulo central que gestiona las comunicaciones entre los diferentes módulos que completan el sistema de diálogo. Para realizar dicha comunicación, se ha definido un protocolo de paquetes XML, que transmiten tanto la información de control como la salida generada por los diferentes módulos. Este sistema incorpora los trabajos desarrollados por Sergio Grau para la implementación de un reconocedor automático del habla mediante la herramienta Sphinx-II y Lluís Hurtado, para el desarrollo del módulo de comprensión, el diseño de la base de datos PostGres que almacena la información del sistema, la integración del sintetizador Festival, y la programación de la estructura de comunicaciones entre los diferentes módulos, así como del módulo central gestor de dichas comunicaciones. Sin este trabajo no hubiese sido posible integrar de una forma tan sencilla los diferentes gestores de diálogo y generadores de respuestas presentados en la tesis.

Los capítulos quinto y sexto de la tesis describen la temática central de la tesis: la gestión del diálogo. Sin duda alguna, el gestor de diálogo constituye el núcleo central en un sistema de diálogo, desde el punto de vista de las múltiples fuentes de información con las que debe tratar (respuesta generada por el módulo de comprensión, historia del diálogo, resultado de las consultas a las bases de datos de la aplicación, etc.), la supervisión que debe realizar del curso del diálogo y su influencia en la determinación del comportamiento del sistema. Para el desarrollo de los gestores descritos se han seguido las premisas de estandarización y portabilidad, así como los esfuerzos por realizar la separación de los aspectos relativos a la tarea del resto de código necesario para realizar la gestión del diálogo.

En el capítulo quinto, se han relatado dos modelos de gestión basados en la utilización de reglas. Estos modelos presentan las ventajas de simplicidad y robustez, así como de rapidez de desarrollo. Para tareas en las que la estrategia que desee aplicarse para la gestión del diálogo esté claramente definida, la utilización de este tipo de modelos es claramente aprovechable. No obstante, su principal inconveniente radica en el crecimiento del número de reglas conforme la complejidad de la tarea aumenta. Asimismo, esta solución tradicional para el desarrollo de gestores de diálogo presenta como desventaja adicional la dificultad de adaptar el gestor implementado para llevar a cabo nuevas tareas diferentes a aquella para la cual se desarrolló el sistema.

El capítulo sexto constituye el núcleo central de la tesis y la principal novedad introducida en la misma: la definición de una metodología completamente estadística para gestionar el diálogo. Sin duda alguna, éste fue el principal reto definido para la tesis, dado que actualmente existe únicamente un conjunto reducido de metodologías que se basen en este tipo de técnicas para gestionar el diálogo, tal y como se ha descrito en el estado del arte de la tesis, la mayoría de las cuales se apoyan en la utilización de heurística para gestionar correctamente el diálogo. El principal objetivo fue definir una metodología que permitiese modelizar estadísticamente la gestión de diálogo, aprendiendo su comportamiento a partir de un corpus de datos etiquetado.

Tal y como se ha descrito en el capítulo sexto, la metodología estadística desarrollada para esta tesis se basa en la modelización de las secuencias de actos de diálogos de usuario y de sistema, y la introducción de una partición en el espacio de todas las posibles secuencias de actos de diálogo, de forma que el proceso de aprendizaje del modelo estadístico a partir de un corpus de diálogos sea manejable. Esta partición se define teniendo en cuenta la información suministrada por el usuario a lo largo del diálogo, considerando además las medidas de confianza proporcionadas por el reconocedor automático del habla y módulo de comprensión. Dicha partición se basa en la utilización del registro del diálogo. En la formulación propuesta, la nueva respuesta del sistema se genera teniendo en cuenta la información disponible en el registro del diálogo y la última respuesta generada por el sistema.

Una idea fundamental del modelo se basa en la codificación definida para representar las fuentes de información presentes en el mismo. Especialmente, cabe destacar la definición de únicamente tres estados para representar cada uno de los valores presentes en el registro del diálogo. Estos tres valores codifican si existe o no un valor para un determinado concepto o atributo y que confianza le han asignado al valor correspondiente el reconocedor y el módulo de comprensión. Estos son los únicos estados que consideramos necesarios para decidir la próxima acción que debe realizar el sistema.

Las aproximaciones basadas en la estimación de un gestor a partir de un corpus de datos deben tratar el problema de modelar todas las posibles situaciones que pueden ocurrir durante el diálogo, utilizando únicamente un corpus de aprendizaje. En la tesis se han propuesto dos metodologías para tratar el problema de la cobertura del modelo. La primera de ellas se basa en la definición de una función distancia que posibilite la selección de la situación presente en el modelo más cercana a aquella no vista, de forma que el diálogo pueda continuar el diálogo de la forma más adecuada.

La segunda metodología se basa en la sustitución de esta función distancia por un clasificador, cuya entrada se corresponde con la situación no vista y su salida puede interpretarse como la situación presente en el modelo más cercana a la no vista. Para la

tesis se han evaluado un conjunto de clasificadores: clasificador naive Bayes multinomial, clasificadores de n-gramas, metodología MGGI y el perceptrón multicapa.

El buen funcionamiento del clasificador basado en redes neuronales, tanto para tratar las situaciones no vistas en el modelo como para decidir la mejor respuesta del sistema para las situaciones vistas, ha posibilitado la implementación de un gestor de diálogo basado en la utilización de un perceptrón para el tratamiento de todas las situaciones del diálogo. De esta forma, este gestor de diálogo se basa en la actualización del registro del diálogo y la clasificación del par (DR, S) que representa dicho registro y la situación actual del diálogo.

La evaluación de las diferentes metodologías que se ha llevado a cabo muestra el funcionamiento satisfactorio de los gestores de diálogo desarrollados. Para realizar esta evaluación se definieron dos técnicas fundamentales. En la primera de ellas se evalúa el comportamiento del gestor de diálogo mediante usuarios reales utilizando el conjunto de módulos que componen el sistema completo desarrollado. Para realizar esta evaluación se han definido una serie de medidas: porcentaje de diálogos que finalizaron con éxito, número de errores detectados y corregidos por el gestor, porcentaje de confirmaciones, número medio de turnos por diálogo, etc. La evaluación de los gestores implementados muestra la validez de los diferentes modelos utilizados. No obstante, una evaluación más detallada es necesaria con el objetivo de probar la validez de su funcionamiento frente a mayores tasas de errores y poder discernir problemas que no han sido detectados debido al poco número de diálogos y usuarios utilizados en esta evaluación.

La segunda evaluación se ha llevado a cabo mediante la definición de particiones de aprendizaje y evaluación en el corpus DIHANA y la utilización de técnicas de validación cruzada. Para llevar a cabo esta evaluación se definieron cuatro medidas: el porcentaje de respuestas proporcionadas por el gestor que coinciden con la de referencia en el corpus, el porcentaje de respuestas que siguen la estrategia definida para el Mago de Oz en DIHANA, el porcentaje de respuestas que son coherentes con el estado actual del diálogo y el porcentaje de respuestas proporcionadas por el gestor que ocasionarían el fallo del diálogo. La evaluación del gestor de diálogo basado en el perceptrón multicapa muestra que un 99,7% de las respuestas proporcionadas por el gestor son coherentes con el estado del diálogo, existiendo únicamente un 0,18% de respuestas que ocasionarían el fallo del diálogo. El clasificador multinomial es el que posee un comportamiento más similar al alcanzado con las redes neuronales, alcanzando un porcentaje de 88,5% de respuestas coherentes.

De las diferentes evaluaciones del gestor basado en redes neuronales mostradas en la tesis, cabe destacar el estudio realizado para evaluar el comportamiento del gestor con respecto a la talla del corpus utilizado para su aprendizaje. Del estudio realizado puede observarse el buen funcionamiento del gestor incluso con la utilización de únicamente un 25% de las muestras disponibles inicialmente en el corpus DIHANA para realizar el aprendizaje del modelo. Este estudio permite concretar de una manera más precisa el número de diálogos que serían necesarios para construir un buen modelo de gestión.

La tarea necesaria para llevar a cabo el procesamiento del corpus y solventar los errores producidos en la adquisición ha sido, sin ningún tipo de dudas, ardua y poco gratificante. Este ha sido la principal causa que ha motivado el desarrollo de técnicas de simulación de usuario que posibilitasen la adquisición automática de un corpus de diálogos, de forma que este esfuerzo no fuese necesario.

En el capítulo octavo se extiende la metodología estadística desarrollada para gestionar el diálogo dentro del ámbito de la simulación estadística de usuarios. De este modo, se modeliza de forma estadística el proceso completo del diálogo, desde la determinación de la nueva intervención del usuario a la determinación de la respuesta que proporciona el sistema. Sin duda, ésta es una de las principales aportaciones resultante del trabajo desarrollado para la tesis.

El simulador estadístico se ha evaluado mediante la interacción con el gestor de diálogo basado en la utilización del perceptrón multicapa, simulando un total de 100.000 diálogos, que se incorporaron para el aprendizaje de un nuevo gestor de diálogo. tal y como se ha comentado, la incorporación de estos diálogos posibilita que el número de situaciones no vistas durante el aprendizaje se reduzca y que el gestor proporcione un menor número de respuestas que ocasionarían el fallo del diálogo. Asimismo, se incrementa el porcentaje de respuestas del gestor que son coherentes con el estado actual del diálogo, posibilitándose que el gestor explore nuevos caminos para la gestión de diálogo, distanciándose de la estrategia definida inicialmente e incorporando al corpus nuevas respuestas para el sistema no contempladas inicialmente.

El capítulo octavo también describe una metodología, más ambiciosa, que permite simular simultáneamente el comportamiento del usuario y del sistema. Esta aproximación se basa en la definición del conjunto de respuestas del usuario y del sistema (en forma de actos de diálogo), la selección aleatoria de las respuestas y la definición de un conjunto de condiciones estrictas que posibiliten la evaluación automática de los diálogos obtenidos.

Mediante esta metodología es posible adquirir de forma automática un corpus de diálogos etiquetado mediante actos de diálogo, tanto para las intervenciones del usuario, como las del sistema. La aplicación de esta metodología para la tarea DIHANA ha permitido la adquisición de un corpus utilizado para el aprendizaje de un gestor de diálogo con unas prestaciones de funcionamiento satisfactorias y con un bajo esfuerzo para su realización. Se simularon un total de dos millones de diálogos.

En último lugar, el capítulo noveno de la tesis completa las temáticas descritas en el estado del arte de la tesis, describiendo la adaptación de las metodologías de gestión estadística del diálogo definidas en DIHANA para posibilitar su utilización en nuevas tareas. Esta adaptación se ha llevado a cabo en el ámbito del proyecto EDECÁN, siendo una de las nuevas tareas seleccionadas la gestión de las instalaciones deportivas disponibles en nuestra universidad. La complejidad de esta adaptación radica en la adición de nuevas funcionalidades que debe llevar a cabo el sistema y que suponen la modificación de la información disponible en la base de datos de la aplicación.

La incorporación de estas funcionalidades supone la aplicación de la metodología estadística propuesta para gestionar el diálogo en dos pasos. En primer lugar, para la determinación de la mejor consulta que debe realizarse al gestor de la información de la aplicación. Seguidamente, se utiliza la consulta realizada a dicho gestor y el resultado de la misma, para determinar la respuesta final del sistema. De esta forma, utilizando un corpus de diálogos etiquetado manualmente, se ha realizado el aprendizaje de un gestor inicial para la tarea. La evaluación de dicho gestor muestra que el 72,9 % de las respuestas generadas por el gestor son coherentes con el estado actual del diálogo.

En la fecha en la que termina la redacción de esta tesis, se encuentra en curso el proyecto EDECÁN. En el marco de este proyecto, está prevista la adquisición de un corpus para la tarea descrita mediante un nuevo esquema en el que participarán dos Magos

de Oz. El primero de ellos se encargará de realizar las funciones propias del módulo de reconocimiento automático del habla y del módulo de comprensión. La representación semántica generada por este Mago será la entrada de un módulo encargado de introducir errores y medidas de confianza en dicha representación. La salida de este módulo constituirá la entrada del gestor de diálogo descrito, de tal forma, que un segundo Mago pueda supervisar la respuesta generada por el gestor, modificándola en el caso que no sea correcta.

Las líneas de trabajo futuro tienen como principal objetivo la realización y evaluación de un sistema de diálogo completo para el proyecto EDECÁN. Para llevar a cabo este objetivo, se están desarrollando actualmente diferentes trabajos para la adaptación de las metodologías descritas en DIHANA para la gestión de diálogo, simulación de usuarios y generación de respuestas. Entre los principales retos que se desean conseguir con la realización de este sistema, cabe destacar la posibilidad de adaptar el funcionamiento del sistema a diferentes perfiles de usuario, de tal forma que el gestor sea capaz de incorporar sugerencias al usuario de acuerdo con sus preferencias anteriores e incluso modificar la estrategia del diálogo durante el transcurso del mismo tras comprobar la evolución de las respuestas del usuario y del sistema.

Anexo A

Adquisición Corpus DIHANA

A.1. Ejemplo de ficha definida para la adquisición del corpus DIHANA

UNIVERSIDAD POLITÉCNICA DE VALENCIA
COLABORACIÓN EN UN PROYECTO DE INVESTIGACIÓN

Código: V101

INFORMACIÓN

1. El Departamento de Sistemas Informáticos y Computación de la Universidad Politécnica de Valencia está desarrollando un proyecto de investigación en el campo del reconocimiento de voz por ordenador. La investigación que se está realizando está orientada al desarrollo de sistemas de atención telefónica que puedan, por ejemplo, darle información sobre medios de transporte o datos meteorológicos.
2. Para llevar a cabo este proyecto, se precisa el registro de diálogos entre usuarios y el sistema de información. Se solicita su colaboración para realizar cuatro registros con consultas sobre información de trenes de largo recorrido.

A fin de que las grabaciones sean útiles en el desarrollo del proyecto, es preciso que los diálogos correspondan a situaciones controladas. Por ello le suministramos la descripción de cuatro “escenarios”, que especifican la información que se supone usted desea obtener al llamar al servicio de información. Las grabaciones se realizarán en castellano.
3. Cuando termine con las grabaciones de los diálogos, lea las frases que aparecen a continuación de los escenarios.

INSTRUCCIONES

1. Las llamadas telefónicas al sistema de atención telefónica se realizarán en nuestras instalaciones, el día de de 2003 a las horas.
2. Sobre los escenarios:
 - a) Antes de acudir a la cita lea atentamente la descripción de los escenarios, familiarizándose con la información (**Objetivo**) que se desea obtener del sistema de atención telefónica, la situación establecida y el tipo de viaje.
 - b) Prepare cuidadosamente los escenarios utilizando su experiencia personal de viajes. Establezca ciudades origen y destino cuando sea necesario, las fechas del viaje, márgenes horarios para realizar el viaje, tipo de tren, viaje de ida y vuelta, etc. Puede escribir notas en la ficha que le hemos suministrado si ello le facilita la tarea.
 - c) Trate de cumplir el objetivo comportándose como lo haría en la situación establecida por el escenario. Elija libremente la información que el sistema le requiera y no esté incluida en el escenario.
 - d) Cuando el origen no está especificado (i.e., *Punto de salida: -*) no debe indicarlo, el sistema entiende que el punto de salida corresponde al origen de la llamada.
 - e) Procure ajustarse a las restricciones impuestas para cumplir su objetivo.
 - f) Las preferencias se pueden cumplir o no.
3. Cuando realice la llamada, hable *normalmente*, con su tono y velocidad habituales, aunque procurando hacer *frases cortas*.
4. Observará que el sistema es muy lento en generar sus respuestas. Ello es normal.
5. Empiece a hablar cuando el sistema haya terminado, después de oír la señal (un pequeño pitido).
6. No cuelgue hasta que no oiga el mensaje de despedida.

CONDICIONES

La colaboración que se presta tiene carácter voluntario y desinteresado, por lo que no supone la adquisición de derecho alguno respecto al proyecto de investigación y sus posteriores usos, entendiéndose que el registro que se realice no constituye "obra" o "ejecución de obra" a tenor de lo dispuesto en el Real Decreto Legislativo 1/1996 de 12 de abril, por el que se aprueba el texto refundido de la Ley de Propiedad Intelectual.

D./Dña.:
con DNI acepta las condiciones anteriores.

Firmado en Valencia, a dede 2003.

Código: V101

Lea las frases en orden (directamente, sin leer el número asociado).

1. Francia, Suiza y Hungría ya hicieron causa común.
2. Estuve en Guernica dando clase de lengua y literatura.
3. Los automóviles circulan por el pueblo.
4. Yo iba hasta la ermita de San Gabriel con este cacharro.
5. Es la riqueza ganadera y la riqueza agrícola.
6. Me vine aquí y me admitieron en su organización.
7. Esto se hace cuando ya se saben los resultados del veterinario.
8. Uno de esos caciques era amigo de Azaña.
9. No, lo que quería decir es de Pontevedra a Guadalajara, cualquier día durante el mes de noviembre, qué horarios tenéis.
10. Sí, quiero saber horarios de trenes de Lérida a Alicante el cuatro de diciembre por la tarde.
11. No, a ver, seguro que hay algún tren en dirección a Jaén.
12. Al cabo de una semana, o sea, salida en viernes y vuelta en viernes.
13. No, llamaba para hacer un trayecto de ida y vuelta Albacete Cuenca, Cuenca Albacete.
14. No, deseo viajar el próximo veinticuatro de abril.
15. Sí, el ocho de octubre puedo salir de Huelva.
16. No, el martes cuatro de septiembre por la tarde.

Código: V101**Escenario 1**

Objetivo:	obtener horario
<i>Tipo de viaje:</i>	viaje de ida
<i>Situación:</i>	visita a unos familiares
<i>Punto de salida:</i>	-
<i>Punto de llegada:</i>	Cuenca
<i>Restricciones:</i>	salir el día 30 de junio

Escenario 2

Objetivo:	obtener horario; obtener precio
<i>Tipo de viaje:</i>	viaje de ida
<i>Situación:</i>	inicia sus vacaciones y ha de coger un avión en Barcelona con destino a Londres
<i>Punto de salida:</i>	-
<i>Punto de llegada:</i>	Barcelona
<i>Restricciones:</i>	salir el día 30 de julio; el avión sale a las 20:00h; que sea un tren rápido

Escenario 3

Objetivo:	obtener horario; obtener precio
<i>Tipo de viaje:</i>	viaje de ida
<i>Situación:</i>	asistir a un concierto
<i>Punto de salida:</i>	Castellón
<i>Punto de llegada:</i>	Barcelona
<i>Restricciones:</i>	viajar el sábado próximo; en Euromed, si es posible

Escenario 4

Objetivo:	obtener horarios; obtener precio
<i>Tipo de viaje:</i>	viaje de ida; viaje de vuelta
<i>Situación:</i>	visitar una ciudad monumental
<i>Punto de salida:</i>	-
<i>Punto de llegada:</i>	elegir ciudad destino
<i>Restricciones:</i>	el viaje durará una semana máximo

Muchas gracias por su colaboración.

A.2. Ficheros almacenados durante la adquisición de cada diálogo

El formato utilizado para almacenar las diferentes fuentes de información generadas en el diálogo fue XML (eXtensible Markup Language) (Bray et al., 2006). Se trata de un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C) (*www.w3.org*) Es una simplificación y adaptación del lenguaje SGML que permite definir la gramática de lenguajes específicos. De este modo, puede considerarse como una manera de definir lenguajes para diferentes necesidades creado no únicamente para su aplicación en Internet, sino como un estándar para el intercambio de información estructurada entre diferentes plataformas (bases de datos, editores de texto, hojas de cálculo, etc.).

A.2.1. Fichero `.fra.xml`. Frases acústicamente balanceadas

Este fichero se definió para almacenar las frases acústicamente balanceadas y de la tarea que debía pronunciar el usuario previamente a su interacción con el sistema. Las etiquetas definidas para este tipo de fichero se enumeran seguidamente:

- **<frases>**: Indica que se trata de un fichero de frases. A continuación hay una pequeña cabecera con las siguientes etiquetas:
 - **<fecha>**: Fecha de la realización de la adquisición del diálogo;
 - **<codigo>**: Código del diálogo, que coincide con el nombre del archivo.
- **<sesion>**: Indica que lo que viene a continuación son las frases a adquirir. Esta información se estructura en turnos numerados secuencialmente empezando por el 1. Cada turno contiene, delimitada entre las etiquetas `<mago>...</mago>`, la acción realizada por el sistema (bienvenida, requerir la repetición de la frase mencionada en último lugar o solicitar la pronunciación de la siguiente) y la frase que debía mencionar el usuario (delimitada entre las etiquetas `<user>...</user>`).

```
<?xml version="1.0" encoding="ISO-8859-15" ?>
<frases>
<fecha>04-03-2004</fecha>
  <hora>18:2</hora>
  <codigo>P143</codigo>
  <sesion>
    <turno numero="1">
      <mago>Bienvenido a la sesión de grabación de frases . A continuación diga la frase numero 1.</mago>
      <user>1: Mi mujer es profesora de Lengua y Literatura.</user>
    </turno>
    <turno numero="2">
      <mago>REPETIR</mago>
      <user>Mi mujer es profesora de Lengua y Literatura.</user>
    </turno>
    <turno numero="3">
      <mago>SIGUIENTE</mago>
      <user>Antes de primero de bachiller ya te traumatizaban.</user>
    </turno>
    <turno numero="4">
      <mago>SIGUIENTE</mago>
      <user>De repente se giró una cosa con gafas y era Carlos.</user>
    </turno>
```

```

<turno numero="5">
  <mago>SIGUIENTE</mago>
  <user>Conseguimos que se hiciese un pueblo.</user>
</turno>
.
.
.
<turno numero="18">
  <mago>SIGUIENTE</mago>
  <user>Quisiera saber si hay trenes desde Lugo a Valladolid.</user>
</turno>
<turno numero="19">
  <mago>SIGUIENTE</mago>
  <user>Sí, el segundo de estos trenes ¿a qué hora llega a Córdoba?</user>
  <mago>FIN</mago>
</turno>
</sesion>
<locutor-info>
  <sexo>mujer</sexo>
  <acento>
    <tipo>Bilbao</tipo>
    <comentario />
  </acento>
</locutor-info>
</frases>

```

A.2.2. Fichero .rec.xml. Módulo de reconocimiento automático del habla

Este fichero almacena las secuencias de palabras proporcionadas por el reconocedor tras cada intervención del usuario. Las etiquetas definidas para este tipo de fichero se enumeran seguidamente:

- **<reconocedor>**: Indica que se trata de un fichero de la salida del reconocedor. A continuación hay una pequeña cabecera con las siguientes etiquetas:
 - **<fecha>**: Fecha de la realización de la adquisición del diálogo;
 - **<codigo>**: Código del diálogo, que coincide con el nombre del archivo.
- **<sesion>**: Indica que lo que viene a continuación son los datos generados en el reconocimiento. Esta información se estructura en turnos numerados secuencialmente empezando por el 1.
 - **<turno numero=N>**: Enésimo turno del diálogo. Cada turno está compuesto a su vez por la siguiente etiqueta (única y optativa):
 - **<frase reconocida>**: Contiene la secuencia de palabras reconocida. Seguidamente aparece la etiqueta **<confianza>**, que se mostrará para cada una de las palabras de la frase reconocida anteriormente. Indica la medida de confianza calculada por el reconocedor para esa palabra, siguiéndose el formato **<palabra valor=PALABRA confianza=CONF>**. CONF es un número entre uno y cero con dos decimales de precisión. En caso de que no hubiese intervención por parte del usuario no aparece esta etiqueta (**<frase reconocida>**).

A continuación se muestra un ejemplo del fichero definido para almacenar la información generada por el reconocedor automático del habla.

```

<?xml version="1.0" encoding="ISO-8859-15"?>
<reconocedor>
  <fecha>17-03-2004</fecha>
  <hora>10:23</hora>
  <codigo>B401-BA3a3</codigo>
<sesion>
  <turno numero="1">
    <frase_reconocida>quería saber el horario de trenes para ir a Madrid el día veinte de marzo
    <confianzas>
      <palabra valor="quer'ia" confianza="0.62"/>
      <palabra valor="saber" confianza="0.33"/>
      <palabra valor="el" confianza="0.34"/>
      <palabra valor="horario" confianza="0.73"/>
      <palabra valor="de" confianza="0.38"/>
      <palabra valor="trenes" confianza="0.62"/>
      <palabra valor="para" confianza="0.70"/>
      <palabra valor="ir" confianza="0.23"/>
      <palabra valor="a" confianza="0.25"/>
      <palabra valor="Madrid" confianza="0.62"/>
      <palabra valor="el" confianza="0.37"/>
      <palabra valor="d'ia" confianza="0.49"/>
      <palabra valor="veinte" confianza="0.75"/>
      <palabra valor="de" confianza="0.0"/>
      <palabra valor="marzo" confianza="0.67"/>
    </confianzas>
  </frase_reconocida>
</turno>
  <turno numero="2">
    <frase_reconocida>no nada más muchas gracias es
    <confianzas>
      <palabra valor="no" confianza="0.47"/>
      <palabra valor="nada" confianza="0.43"/>
      <palabra valor="m'as" confianza="0.50"/>
      <palabra valor="muchas" confianza="0.68"/>
      <palabra valor="gracias" confianza="0.55"/>
      <palabra valor="es" confianza="0.61"/>
    </confianzas>
  </frase_reconocida>
</turno>
  <turno numero="3">
</turno>
</sesion>
<locutor_info>
  <sexo>hombre</sexo>
  <acento>
    <tipo> Bilbao</tipo>
    <comentario> </comentario>
  </acento>
</locutor_info>
</reconocedor>

```

A.2.3. Fichero .com.xml. Módulo de comprensión semántica

Este fichero almacena la representación semántica generada por el módulo de comprensión para cada una de las intervenciones del usuario (conjunto de palabras proporcionadas por el reconocedor). Las etiquetas correspondientes al formato de este fichero son:

- **<comprension>**: Indica que se trata de un fichero de comprensión semántica. A continuación hay una pequeña cabecera con las siguientes etiquetas:
 - **<fecha>**: Fecha de la realización de la adquisición del diálogo;
 - **<codigo>**: Código del diálogo, que coincide con el nombre del archivo.

- **<sesion>**: Indica que lo que viene a continuación son los datos proporcionados por el módulo de comprensión para el diálogo correspondiente. Esta información está estructurada en turnos numerados secuencialmente empezando por el 1.
 - **<turno numero=N>**: Se trata del enésimo turno del diálogo. Cada turno está compuesto a su vez por la siguiente etiqueta (única y optativa).
 - **<frame_comprehension>**: Indica la salida del módulo de comprensión generada a partir de la intervención del usuario en ese turno. El formato de esa salida es el siguiente: `<!-- interpretación semántica -->` En caso de que no hubiese intervención por parte del usuario no aparece esta etiqueta.

A continuación se muestra un ejemplo de uno del fichero definido para almacenar la información generada por el módulo de comprensión semántica.

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<comprension>
  <fecha>17-03-2004</fecha>
  <hora>10:23</hora>
  <codigo>B401-BA3a3</codigo>
  <sesion>
    <turno numero="1">
      <frame_comprehension>
        <!--
          quer'ia saber:consulta
          el horario de trenes para ir:<hora_s>
          a:m_destino
          madrid:ciudad_destino
          el d'ia veinte:dia_mes_s
          de marzo:mes_s

          (HORA-SALIDA) *[1.0,0.5]
          CIUDAD-DESTINO:madrid *[1.0,0.5]
          INTERVALO-FECHA-SALIDA:20-03-2004/20-03-2004 *[1.0,0.5]
          -->
        </frame_comprehension>
      </turno>
    <turno numero="2">
      <frame_comprehension>
        <!--
          no nada m'as muchas gracias es:<cierre>

          (CIERRE) *[1.0,0.5]
          -->
        </frame_comprehension>
      </turno>
    <turno numero="3">
      </turno>
    </sesion>
  <locutor_info>
    <sexo>hombre</sexo>
    <acento>
      <tipo> Bilbao</tipo>
      <comentario> </comentario>
    </acento>
  </locutor_info>
</comprension>
```

A.2.4. Fichero .dia.xml. Módulo gestor de diálogo

Este fichero almacena las intervenciones del usuario (salida generada por el reconocedor automático del habla tras cada una de sus intervenciones) y la respuesta correspondiente

del sistema (respuesta seleccionada por el Mago de Oz). Las etiquetas definidas en este fichero son:

- **<dialogo>**: Sirve para indicar que es un fichero de diálogo. A continuación le sigue una pequeña cabecera con las siguientes etiquetas:
 - **<fecha>**: Fecha de la realización de la adquisición del diálogo;
 - **<codigo>**: Código del diálogo, que coincide con el nombre del archivo.
- **<sesion>**: Indica que los datos que vienen a continuación son ya propiamente los adquiridos en el diálogo correspondiente. Estos están estructurados en turnos numerados secuencialmente empezando por el 1.
- **<turno numero=N>**: Enésimo turno del diálogo. Cada turno está compuesto al principio por una o varias intervenciones del mago (obligatorio) y una al final por parte del usuario (optativo).
 - **<mago>**: Corresponde a la frase que ha dicho el mago en ese turno (transcripción correcta, ya que proviene del archivo de respuestas).
 - **<user>**: Muestra las palabras reconocidas tras la intervención del usuario en ese turno. En caso de que no hubiese intervención por parte del usuario no aparece esta etiqueta.

A continuación se muestra un ejemplo de los ficheros definidos para almacenar la secuencia de turnos sistema-usuario generados en el diálogo.

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<dialogo>
  <fecha>17-03-2004</fecha>
  <hora>10:23</hora>
  <codigo>B401-BA3a3</codigo>
<sesion>
  <turno numero="1">
    <mago>Bienvenido al servicio de información de trenes.¿En qué puedo ayudarle?</mago>
    <user>Quería saber el horario de trenes para ir a Madrid el día veinte de marzo</user>
  </turno>
  <turno numero="2">
    <mago>Le consulto horarios de trenes con salida en bilbao destino madrid saliendo el sábado,
      día 20 de marzo de 2004. Un momento por favor.</mago>
    <mago>El único tren que realiza el trayecto es un talgo que sale a las 9 y 50 de la mañana .
      ¿Desea algo más?</mago>
    <user>No nada más muchas gracias es</user>
  </turno>
  <turno numero="3">
    <mago>Gracias por utilizar este servicio. Feliz viaje.</mago>
  </turno>
</sesion>
<locutor_info>
  <sexo>hombre</sexo>
  <acento>
    <tipo> Bilbao</tipo>
    <comentario> </comentario>
  </acento>
</locutor_info>
</dialogo>
```

A.2.5. Fichero .que.xml. Consultas a la base de datos

Este fichero almacena las consultas realizadas a la base de datos que contiene la información de la tarea y el resultado de ejecutar las mismas. Las etiquetas definidas son:

- **<query>**: Sirve para indicar que es el archivo donde se almacena las consultas a la base de datos y los resultados de las mismas. A continuación hay una pequeña cabecera con las etiquetas:
 - **<fecha>**: Fecha de la realización de la adquisición del diálogo;
 - **<codigo>**: Código del diálogo, que coincide con el nombre del archivo.
- **<sesion>**: Indica que la información que le sigue se corresponde a las consultas a la base de datos. La información se estructura en turnos, numerados secuencialmente empezando por el 1.
- **<turno numero=N>**: Enésimo turno del diálogo. Cada turno está compuesto a su vez por las siguientes etiquetas (pueden aparecer varias veces y son optativas, aunque siempre aparecen juntas):
 - **<query>**: Muestra los campos (conceptos y atributos) sobre los que se ha realizado la consulta. El formato es el siguiente: `<!-- datos de la consulta según el formato establecido de la aplicación -->`. Este formato consta de la siguiente información: número de campos de la consulta, secuencia de pares etiqueta del campo (e.g. `<CIUDAD-ORIGEN>`) y número de valores correspondientes al mismo y, por último, los valores concretos del campo.
 - **<resultados_query>**: Muestra el resultado de realizar la consulta anterior. El formato es el siguiente: `<!-- -- resultados de la consulta según el formato establecido de la aplicación -->`. Este formato es idéntico al de la consulta, incorporándose en primer lugar un número que indica el número de trenes, es decir, aquellos que satisfacen las condiciones de la consulta. Posteriormente a ese número, se citan las características de los trenes obtenidos.

A continuación se muestra un ejemplo de los ficheros definidos para almacenar las consultas a la base de datos del sistema.

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<query>
  <fecha>17-03-2004</fecha>
  <hora>10:23</hora>
  <codigo>B401-BA3a3</codigo>
<sesion>
  <turno numero="1"></turno>
  <turno numero="2">
    <query>
      <!--
      <CIUDAD-ORIGEN> bilbao
      <CIUDAD-DESTINO> madrid
      <DIRECCION> salida
      <FECHA-SALIDA> 20-03-2004/20-03-2004
      <CLASE-BILLETE> turista
      -->
    </query>
  <resultados_query>
```

```

<!--
<CIUDAD-ORIGEN> bilbao
<CIUDAD-DESTINO> madrid
<FECHA-SALIDA> 20-03-04
<FECHA-LLEGADA> 20-03-04
<HORA-SALIDA> 09:50
<HORA-LLEGADA> 16:53
<TIPO-TREN> talgo
<CLASE-BILLETE> preferente turista
<PRECIO> 39.00 29.50
<SERVICIOS> música reserva_telefónica vídeo prensa
-->
</resultados_query>
</turno>
<turno numero="3"></turno>
</sesion>
<locutor_info>
  <sexo>hombre</sexo>
  <acento>
    <tipo> Bilbao</tipo>
    <comentario> </comentario>
  </acento>
</locutor_info>
</query>

```

A.2.6. Fichero .piz.xml. Registro del diálogo

Este fichero almacena el estado del registro del diálogo en el momento de cada una de las intervenciones del sistema. Las etiquetas definidas para este tipo de ficheros son:

- **<pizarra>**: Sirve para indicar que es el archivo donde se almacena el registro del diálogo en cada turno del sistema. A continuación hay una pequeña cabecera con las etiquetas:
 - **<fecha>**: Fecha de la realización de la adquisición del diálogo;
 - **<codigo>**: Código del diálogo, que coincide con el nombre del archivo.
- **<sesion>**: Indica que los datos que se muestran a continuación se corresponden con el estado del registro del diálogo. Esta información se estructura en turnos numerados secuencialmente empezando por el 1.
 - **<turno numero=N>**: Enésimo turno del diálogo. Cada turno está compuesto a su vez por una o varias de las siguientes etiquetas (optativas):
 - **<evento_eliminado_frame tipo=FRA/>**: El Mago eliminó en ese turno del sistema el frame FRA del registro.
 - **<evento_agnadido_frame tipo=FRA/>**: El Mago incorporó en ese turno el frame FRA en el registro;
 - **<evento_nuevo_valor tipo=ATR valor=VAL estado=EST/>**: El Mago incorporó en la pizarra el atributo ATR con el valor VAL y con un estado de confianza de ese atributo de EST.
 - **<evento_eliminado_valor tipo=ATR valor=VAL/>**: El Mago eliminó del registro el atributo ATR que contiene el dato VAL.
 - **<evento_cambio_valor tipo=ATR valor_anterior=ANT valor_nuevo=NUE/>**: El Mago cambió el valor ANT del atributo ATR por el de NUE en el registro.

- **<evento_cambio_estado tipo=ATR valor=VALOR estado_anterior=ANT estado_nuevo=NUE/>**: El Mago cambió en el atributo ATR que contiene el valor VALOR, el estado ANT por el estado NUE.

A continuación se muestra un ejemplo de los ficheros definidos para almacenar el registro del diálogo.

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<pizarra>
  <fecha>17-03-2004</fecha>
  <hora>10:23</hora>
  <codigo>B401-BA3a3</codigo>
<sesion>
  <turno numero="1">
    <evento_nuevo_valor tipo="CiudadDestino" valor="madrid" estado="Indefinido"/>
    <evento_nuevo_valor tipo="FechaSalida" valor="20-03-2004/20-03-2004" estado="Indefinido"/>
    <evento_agnadado_frame tipo="Hora"/>
    <pizarra>
      <atributo tipo="CiudadOrigen"><valor="bilbao" estado="Indefinido"/></atributo>
      <atributo tipo="CiudadDestino"><valor="madrid" estado="Indefinido"/></atributo>
      <atributo tipo="Direccion"><valor="salida" estado="Indefinido"/></atributo>
      <atributo tipo="FechaSalida"><valor="20-03-2004/20-03-2004" estado="Indefinido"/></atributo>
      <atributo tipo="ClaseBillete"><valor="turista" estado="Indefinido"/></atributo>
    </pizarra>
  </turno>
  <turno numero="2">
    <pizarra>
      <atributo tipo="CiudadOrigen"><valor="bilbao" estado="Indefinido"/></atributo>
      <atributo tipo="CiudadDestino"><valor="madrid" estado="Indefinido"/></atributo>
      <atributo tipo="Direccion"><valor="salida" estado="Indefinido"/></atributo>
      <atributo tipo="FechaSalida"><valor="20-03-2004/20-03-2004" estado="Indefinido"/></atributo>
      <atributo tipo="ClaseBillete"><valor="turista" estado="Indefinido"/></atributo>
    </pizarra>
  </turno>
  <turno numero="3">
    <pizarra>
      <atributo tipo="CiudadOrigen"><valor="bilbao" estado="Indefinido"/></atributo>
      <atributo tipo="CiudadDestino"><valor="madrid" estado="Indefinido"/>
      </atributo><atributo tipo="Direccion"><valor="salida" estado="Indefinido"/></atributo>
      <atributo tipo="FechaSalida"><valor="20-03-2004/20-03-2004" estado="Indefinido"/></atributo>
      <atributo tipo="ClaseBillete"><valor="turista" estado="Indefinido"/></atributo>
    </pizarra>
  </turno>
</sesion>
<locutor_info>
  <sexo>hombre</sexo>
  <acento>
    <tipo> Bilbao</tipo>
    <comentario> </comentario>
  </acento>
</locutor_info>
</pizarra>
```

A.3. Resultados de la encuestas

A.3.1. Encuesta que rellenaron los solicitantes

Encuesta informante

Código: V101

Fecha adquisición: Edad: Sexo: M H

1. ¿Entendió al sistema cuando éste hablaba?
Siempre Habitualmente La mayor parte de las veces A veces Nunca

2. ¿El sistema comprendió lo que usted le decía?
Siempre Habitualmente La mayor parte de las veces A veces Nunca

3. ¿Fue fácil conseguir el objetivo del escenario 1?
Muy fácil Fácil Así así Difícil No pude
Si ha contestado "No pude", por favor, indique la información no obtenida y las causas por las que no pudo obtenerla.
.....
.....
.....

4. ¿Fue fácil conseguir el objetivo del escenario 2?
Muy fácil Fácil Así así Difícil No pude
Si ha contestado "No pude", por favor, indique la información no obtenida y las causas por las que no pudo obtenerla.
.....
.....
.....

5. ¿Fue fácil conseguir el objetivo del escenario 3?
Muy fácil Fácil Así así Difícil No pude
Si ha contestado "No pude", por favor, indique la información no obtenida y las causas por las que no pudo obtenerla.
.....
.....
.....

6. ¿Fue fácil conseguir el objetivo del escenario 4?

Muy fácil Fácil Así así Difícil No pude

Si ha contestado "No pude", por favor, indique la información no obtenida y las causas por las que no pudo obtenerla.

.....

7. ¿Fue adecuado el ritmo de interacción?

Siempre Habitualmente La mayor parte de las veces A veces Nunca

8. ¿Supo usted cómo actuar en cada momento del diálogo?

Siempre Habitualmente La mayor parte de las veces A veces Nunca

9. ¿Con qué frecuencia el sistema fue lento en su respuesta?

Siempre Habitualmente La mayor parte de las veces A veces Nunca

10. ¿El sistema se comportó del modo esperado durante la conversación?

Siempre Habitualmente La mayor parte de las veces A veces Nunca

Por favor, realice un comentario general sobre la experiencia.

.....

A.3.2. Estadísticas generales de las encuestas y evaluación de los escenarios

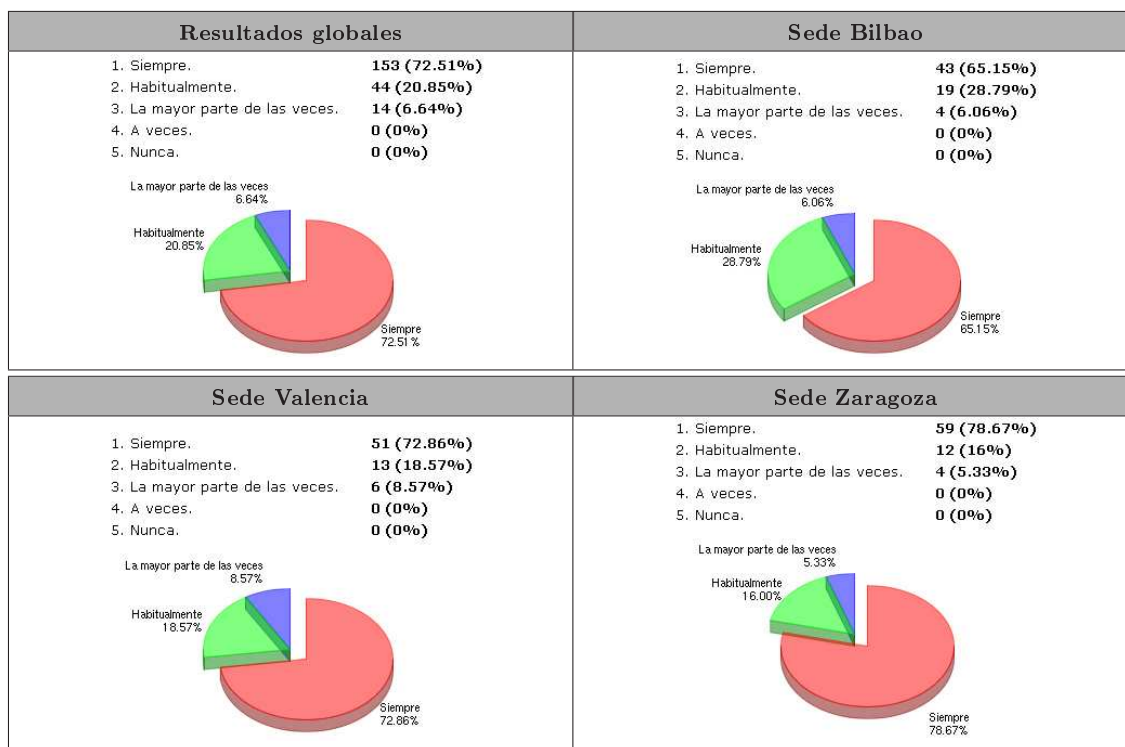


Figura A.1: ¿Entendió al sistema cuando éste le hablaba?

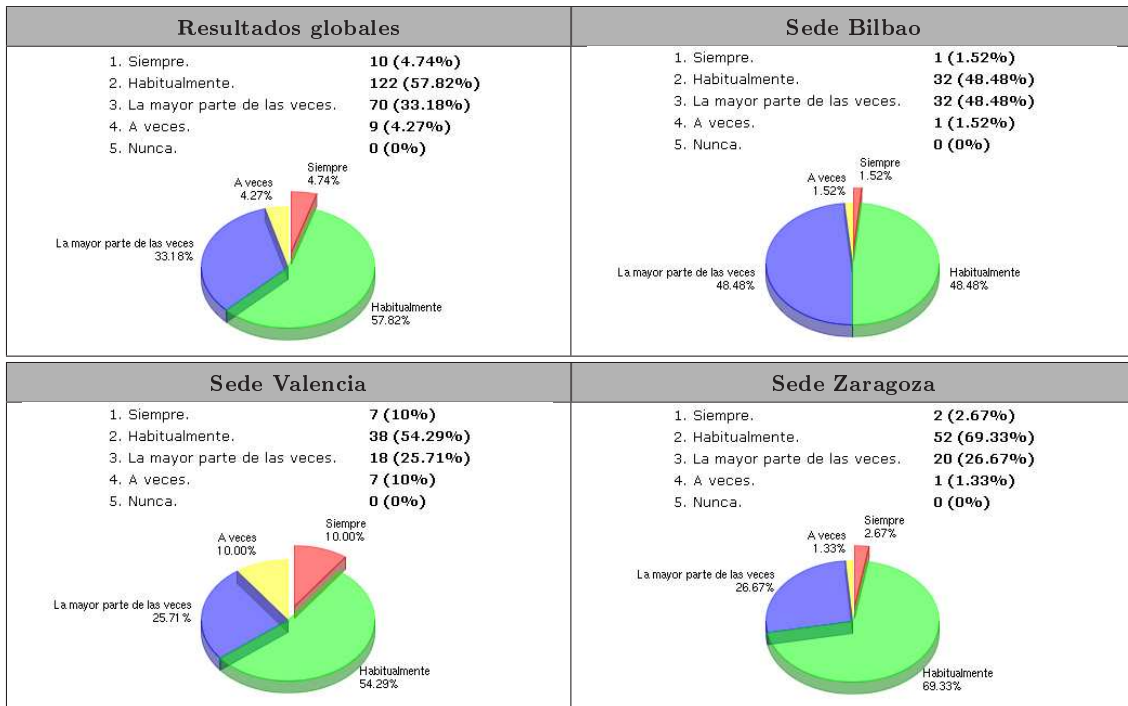


Figura A.2: ¿El sistema comprendió lo que usted le decía?

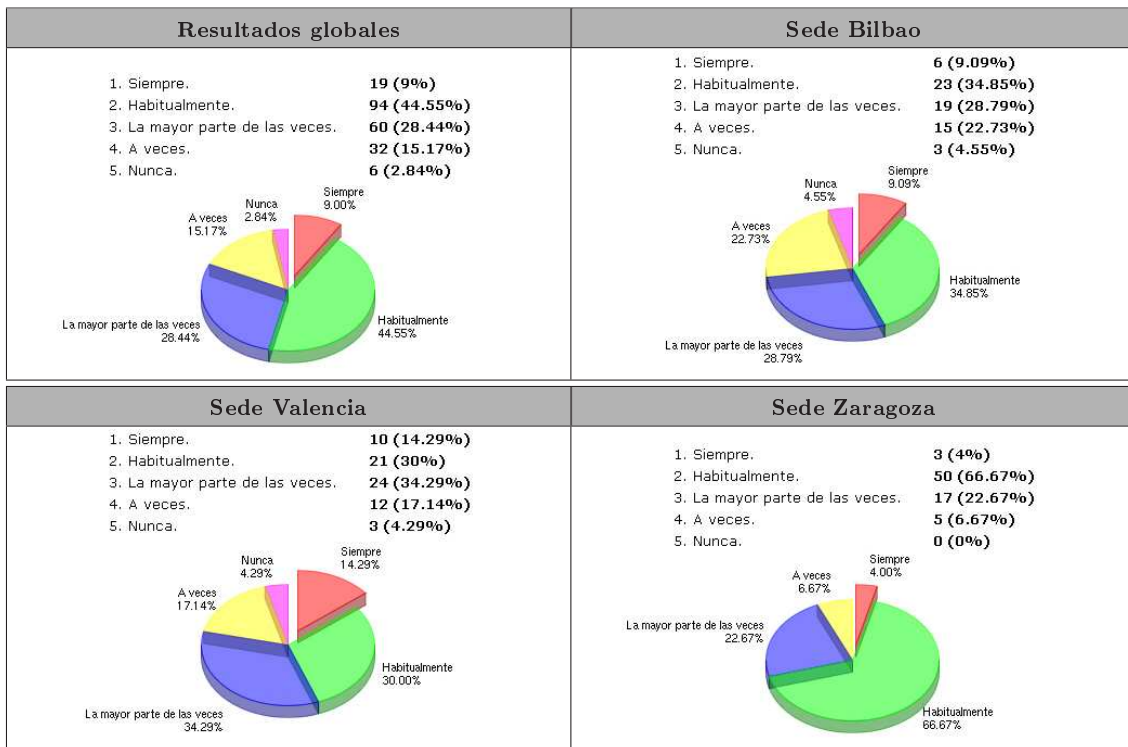


Figura A.3: ¿Fue adecuado el ritmo de interacción?

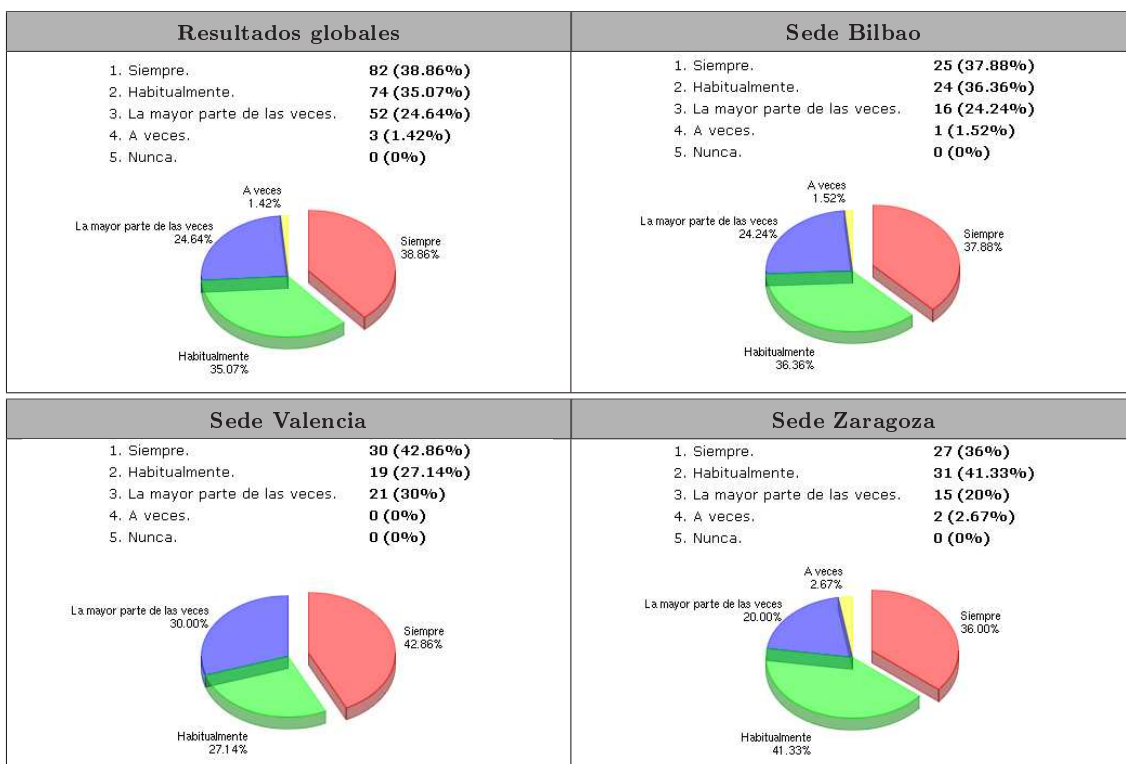


Figura A.4: ¿Supo usted cómo actuar en cada momento del diálogo?

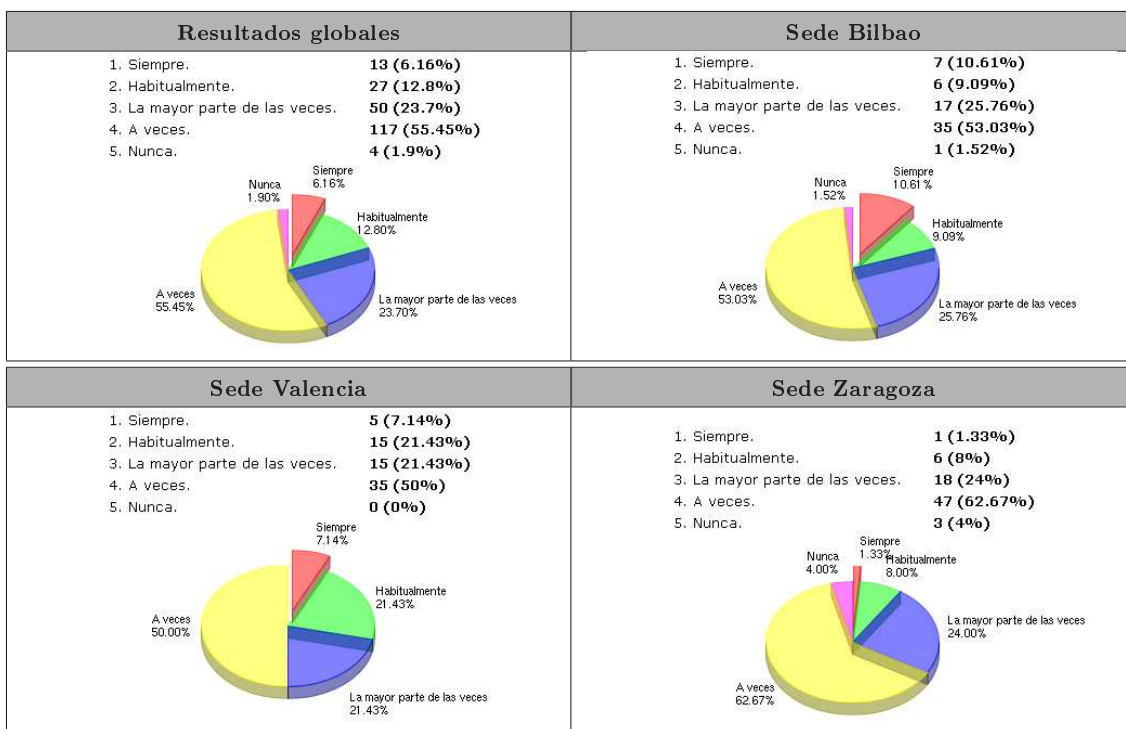


Figura A.5: ¿Con qué frecuencia el sistema fue lento en su respuesta?

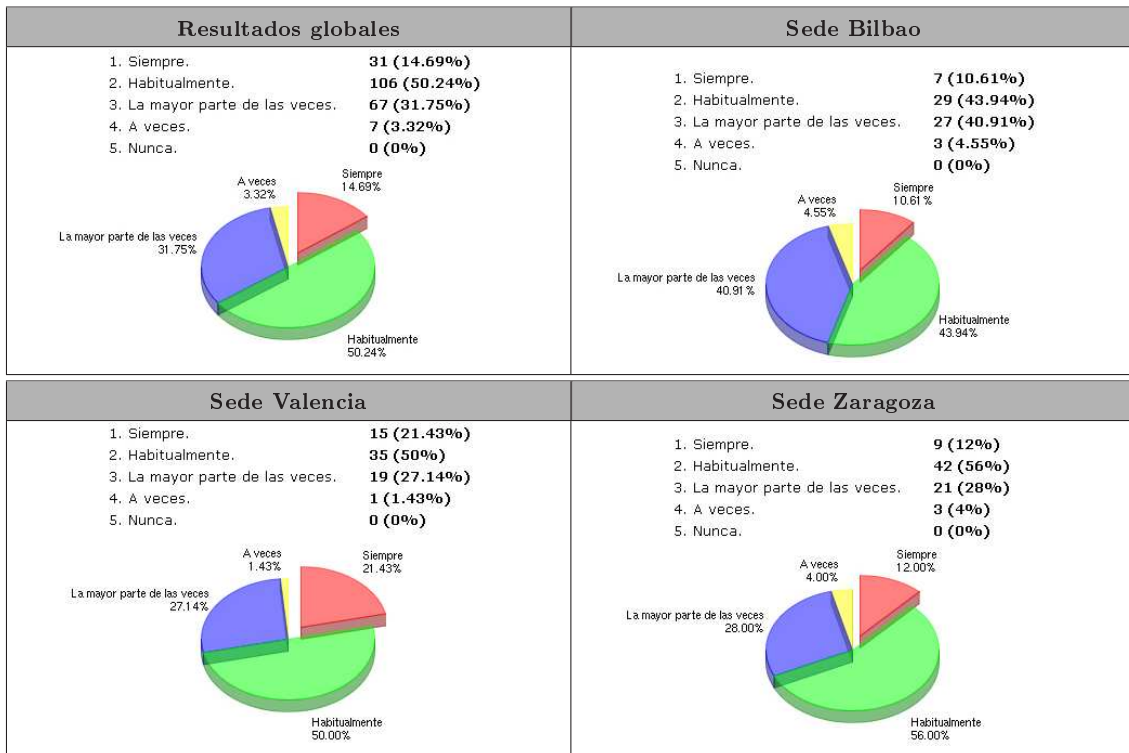


Figura A.6: ¿El sistema se comportó del modo esperado durante la conversación?

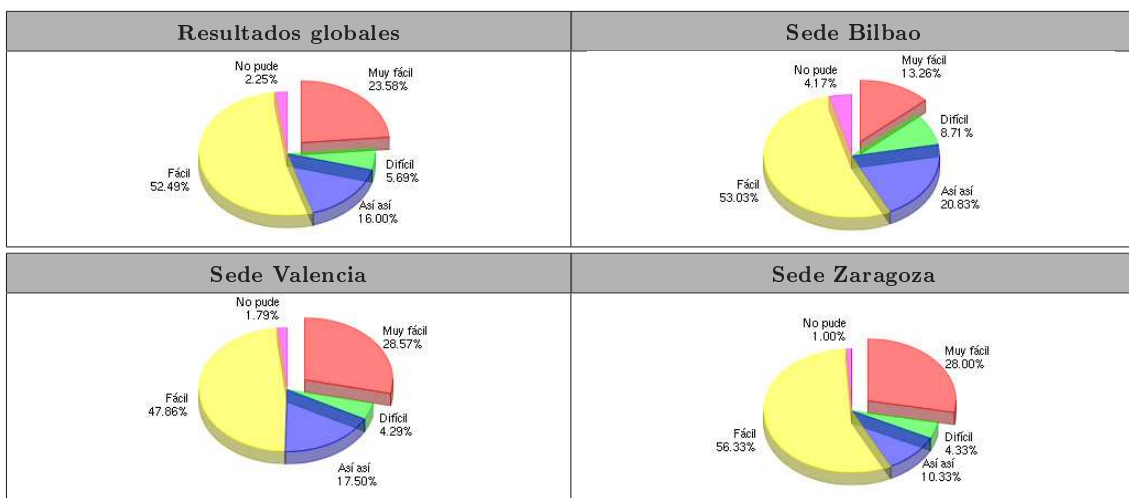


Figura A.7: Resultados globales de los escenarios

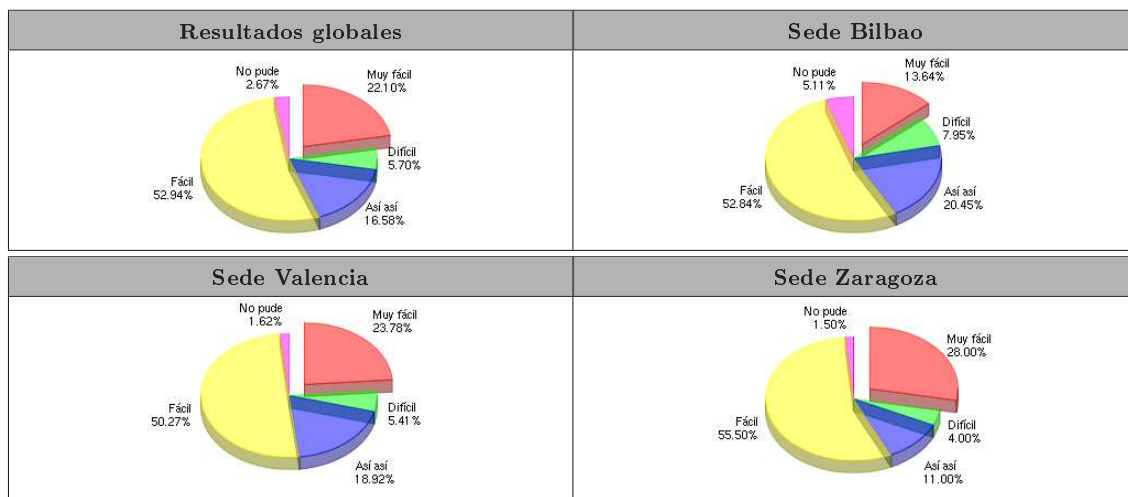


Figura A.8: Resultados de los escenarios cerrados

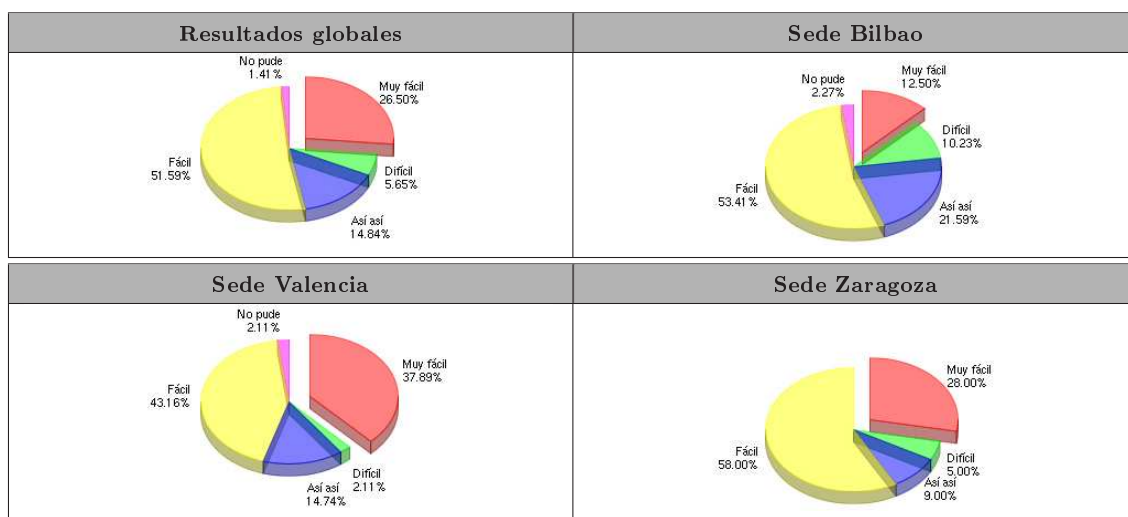


Figura A.9: Resultados de los escenarios abiertos

Anexo B

Sistema de diálogo DIHANA

B.1. Formato de los paquetes XML definidos

B.1.1. Módulo de entrada (RAH)

Recibe paquetes de control procedentes del módulo de síntesis de texto a voz indicando el inicio de la adquisición, la finalización de la misma o el término de la interacción del usuario con el sistema. Seguidamente se muestra un ejemplo del paquete XML de inicio de la adquisición:

```
<?xml version="1.0" encoding="ISO-8859-15"?>
  <paquete destino="RAH" origen="CTV" version="1.0">
    <ini_adqui/>
  </paquete>
```

La frase reconocida se remite al módulo de comprensión. A continuación se muestra el paquete definido para realizar esta comunicación:

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<paquete destino="CH" origen="RAH" version="1.0">
  <frase_completa>
    <frase_reconocida>
      <str_info>
        horarios para Madrid
      </str_info>
      <confianzas>
        <palabra confianza="0.5" valor="horarios"/>
        <palabra confianza="0.4" valor="para"/>
        <palabra confianza="0.9" valor="Madrid"/>
      </confianzas>
    </frase_reconocida>
    <gramática nombre="dihana.jsgf"/>
  </frase_completa>
</paquete>
```

B.1.2. Módulo de comprensión (CH)

Recibe la frase reconocida siguiendo el formato del paquete mostrado en la sección anterior. El resultado del proceso de comprensión semántica se remite al gestor de diálogo siguiendo el formato siguiente:

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<paquete destino="GD" origen="CH" version="1.0">
```

```

<comprension>
  <segmentacion_semantica>
    <segmento etiqueta="consulta" valor="quiero">
      <confianza valor="0.0219018941131"/>
      <confianza valor="0.126746506986"/>
    </segmento>
    <segmento etiqueta="hora_s" valor="horarios">
      <confianza valor="0.14695464419"/>
      <confianza valor="0.0763993948563"/>
    </segmento>
    <segmento etiqueta="hora_s" valor="a las cinco o a las seis">
      <confianza valor="0.111408972611"/>
      <confianza valor="0.142503536068"/>
    </segmento>
  </segmentacion_semantica>
  <frames>
    <frame tipo="(HORA-SALIDA)">
      <confianza_frame valor="0.14695464419"/>
      <confianza_frame valor="0.0763993948563"/>
      <atributo tipo="INTERVALO-HORA-SALIDA">
        <valor_atributo valor="16.06-18.06">
          <confianza valor="0.111408972611"/>
          <confianza valor="0.142503536068"/>
        </valor_atributo>
      </atributo>
    </frame>
    <frame tipo="(PRECIO)">
      <confianza_frame valor="0.00052060226672"/>
      <confianza_frame valor="0.0490797546012"/>
      <atributo tipo="INTERVALO-HORA-SALIDA">
        <valor_atributo valor="16.06-18.06">
          <confianza valor="0.111408972611"/>
          <confianza valor="0.142503536068"/>
        </valor_atributo>
      </atributo>
    </frame>
  </frames>
</comprension>
</paquete>

```

B.1.3. Gestor de diálogo (GD)

Recibe la salida generada por el módulo de comprensión siguiendo el formato mostrado en la sección anterior. Así mismo, puede recibir paquetes XML procedentes del módulo de consulta a la base de datos (BD). El formato definido para que el módulo BD transmita el resultado de la consulta se muestra a continuación:

```

<?xml version="1.0" encoding="ISO-8859-15"?>
<paquete destino="GD" origen="BD" version="1.0">
  <respuesta_bd>
    <info_tren>
      <str_info> Valencia </str_info>
      <str_info> Bilbao </str_info>
      <str_info> 21:50 </str_info>
      <str_info> 09:55 </str_info>
      <str_info> 24-11-2006 </str_info>
      <str_info> 24-11-2006 </str_info>
      <str_info> X </str_info>
      <str_info> Sol de Levante </str_info>
    </info_tren>
  </respuesta_bd>
</paquete>

```

Envía paquetes al generador de respuestas siguiendo el formato indicado a continuación:

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<paquete destino="GR" origen="GD" version="1.0">
  <dialogo>
    <frames>
      <frame tipo="(CONFIRMACION)">
        <confianza_frame valor="1.00"/>
        <atributo tipo="CIUDAD-ORIGEN">
          <valor_atributo valor="valencia">
            <confianza valor="0.63"/>
          </valor_atributo>
        </atributo>
        <atributo tipo="CIUDAD-DESTINO-ACTUAL">
          <valor_atributo valor="madrid">
            <confianza valor="0.94"/>
          </valor_atributo>
        </atributo>
      </frame>
    </frames>
  </dialogo>
</paquete>
```

Remite paquetes al módulo de consulta a la base de datos con el siguiente formato:

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<paquete destino="BD" origen="GD" version="1.0">
  <consulta_bd>
    <frames>
      <frame tipo="(HORA-SALIDA)">
        <atributo tipo="CIUDAD-ORIGEN">
          <valor_atributo valor="valencia"/>
        </atributo>
        <atributo tipo="CIUDAD-DESTINO">
          <valor_atributo valor="bilbao"/>
        </atributo>
        <atributo tipo="INTERVALO-FECHA-SALIDA">
          <valor_atributo valor="24-11-2004/24-11-2004"/>
        </atributo>
      </frame>
    </frames>
  </consulta_bd>
</paquete>
```

B.1.4. Módulo generador de respuestas (GR)

Recibe paquetes XML procedentes del gestor de diálogo siguiendo el formato indicado en la sección anterior. Remite la frase en lenguaje natural al módulo de síntesis de texto a voz siguiendo el formato indicando a continuación:

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<paquete destino="CTV" origen="GR" version="1.0">
  <frase>
    <str_info>
      hay 8 trenes , y le informo del primero . el tren sale a las diez
      menos diez de la noche . desea algo más ?
    </str_info>
  </frase>
</paquete>
```

B.1.5. Módulo de síntesis de voz (CTV)

Recibe la frase en lenguaje natural proporcionada por el generador de respuestas siguiendo el formato indicado anteriormente. Remite paquetes XML de control al reconocedor automático del habla (Sección B.1.1).

B.1.6. Módulo de base de datos (BD)

Recibe del gestor de diálogo una petición de consulta a la base de datos siguiendo el formato previamente definido. El resultado de la consulta se transmite al gestor mediante el paquete XML mostrado en la Sección B.1.3.

B.2. Categorías definidas en el corpus DIHANA

B.2.1. Ciudades

Albacete	Granada	Ponferrada
Alcázar de San Juan	Guadalajara	Pontevedra
Algeciras	Hendaya	Puertollano
Alicante	Huelva	Reus
Almería	Huesca	Sahagún
Ávila	Irún	Salamanca
Badajoz	Iruña/Pamplona	San Sebastián
Barcelona	Jaén	Santander
Bilbao	Jerez de la Frontera	Santiago
Bobadilla	La Coruña	Santiago de Compostela
Burgos	León	Segovia
Cáceres	Lérida/Lleida	Sevilla
Cádiz	Linares - Baeza	Soria
Castelló/Castellón	Logroño	Sueca
Cartagena	Lorca	Tarragona
Ciudad Real	Lugo	Teruel
Córdoba	Madrid	Toledo
Cuenca	Málaga	Tudela
Cullera	Medina del Campo	Valencia
Elda - Petrer	Mérida	Valladolid
El Ferrol	Miranda del Ebro	Vigo
Ermua	Monzón	Villena
Figueras	Murcia	Zamora
Gandia	Navalmoral de la Mata	Zaragoza
Gasteiz/Vitoria	Orense	Cualquier
Gerona/Girona	Oviedo	
Gijón	Palencia	

B.2.2. Servicios

Música	Vídeo	Autoexpreso
Checking	Teléfono	Motoexpreso
Compartimento	Prensa	Cualquier
Facturación equipajes	Copa bienvenida	
Reserva telefónica	Restaurante	
Adaptación minusválidos	Cafetería	

B.2.3. Tipo-viaje

Nil	Ida y vuelta
Ida	Vuelta

B.2.4. Tipo-tren

Alaris	Delta	Rápido
Altaria	Estrecho	Más rápido
Andalucía Exprés	Estrella	Río Aragón
Arco	Euromed	Regional
Atlántico	Extremadura	Regional Exprés
Ave	Finisterre	Sol de Levante
Castilla y León Exprés	Galicia	Surex
Catalan Talgo	García Lorca	Talgo
Cataluña Exprés	Intercity	Talgo 200
Costa Brava	Lanzadera	TRD
Costa de la Mañana	Mare Nostrum	Trenhotel
Costa Vasca	Miguel de Unamuno	Triana
Costa Verde	Pío Baroja	Cualquier
Covadonga	Picasso	

B.2.5. Precio

Barato	Caro	Cualquier
Más barato	Más caro	

B.2.6. Número-relativo-orden

Primero	Siguiente	Otro
Segundo	Anterior	TREN-ACTUAL
Tercero	Antepenúltimo	
Cuarto	Penúltimo	
Quinto	Último	

B.2.7. Dirección

Salida	Salida y llegada
Llegada	Cualquier

B.2.8. Clase billete

Turista	Litera
Preferente	Cama
Club	Cualquier

B.3. Gestión del diálogo: Implementación en LUA de las redes neuronales

```

clases = {}
dic_clases = {}
numclases = 0
for line in io.lines("clases.txt") do
    numclases = numclases+1
    clases[line] = numclases
    table.insert(dic_clases,line)
end

casiuno = 0.9
casicero = (1-casiuno)/(numclases-1)
ds_ident = dataset.identity(numclases,casicero,casiuno)

function leer_fichero(filename)
    local f = io.open(filename,"r")
    local fichero = f:read("*a")
    f:close()
    local tsalida = {}
    for etiqueta in string.gfind(fichero,"[01%*]+([~%c]+)%c") do
        table.insert(tsalida,clases[etiqueta])
    end
    local m_salida = matrix(table.getn(tsalida),tsalida)
    fichero = string.gsub(fichero,"[~01%c)","")
    local numfeatures
    _,numfeatures = string.find(fichero,"[01]+")
    local m_entrada = matrix.fromString(string.format("%d\nascii\n%s",
        numfeatures,
        string.gsub(fichero,"([01])","%1 ")))

    local ds_entrada = dataset.matrix(m_entrada)
    local d_salida = dataset.matrix(m_salida)
    local ds_salida = dataset.indexed(d_salida, {ds_ident})
    return ds_entrada,ds_salida,d_salida
end

test_input, test_output, d_salida = leer_fichero("tu1.txt")
datostest = {
    input_dataset = test_input,
    output_dataset = test_output,
}

lared=Mlp.load("lared.txt")
for indice,entrada in test_input:patterns() do
    resultado = lared:classify(entrada)
    printf("%s\n", dic_clases[resultado])
end

```

Figura B.1: Script en LUA para la clasificación de muestras mediante un MLP

```

clases = {}
dic_clases = {}
numclases = 0
for line in io.lines("clases.txt") do
    numclases = numclases+1
    clases[line] = numclases
    table.insert(dic_clases,line)
end
-- creamos un dataset para numclases:
casiuno = 0.9
casicero = (1-casiuno)/(numclases-1)
ds_ident = dataset.identity(numclases,casicero,casiuno)
function leer_fichero(filename)
    local f = io.open(filename,"r")
    local fichero = f:read("*a")
    f:close()
    local tsalida = {}
    for etiqueta in string.gfind(fichero,"[01%*]+([~%c]+)%c") do
        table.insert(tsalida,clases[etiqueta])
    end
    local m_salida = matrix(table.getn(tsalida),tsalida)
    fichero = string.gsub(fichero,"[~01%*]+","")
    local numfeatures
    _,numfeatures = string.find(fichero,"[01]+")
    local m_entrada = matrix.fromString(string.format("* %d\nascii\n%s",
numfeatures, string.gsub(fichero,"([01])","%1 ")))
    local ds_entrada = dataset.matrix(m_entrada)
    local d_salida = dataset.matrix(m_salida)
    local ds_salida = dataset.indexed(d_salida, {ds_ident})
    return ds_entrada,ds_salida,d_salida
end
-- LECTURA DATOS
train_input, train_output = leer_fichero("ttrain1.txt") test_input, test_output,
d_salida = leer_fichero("tuning1.txt")
-- PARAMETROS AJUSTABLES:
num_ocultas1 = 110
num_ocultas2 = 110
el_learning_rate = 0.03
el_momentum = 0.02
weight_decay = 0.0
semilla_red = 123
semilla_shuffle = 456
veces_train = 100
-- CREAMOS LA RED
num_entradas = train_input:patternSize()
num_salidas = numclases
aleat = random(semilla_red)
otrorand = random(semilla_shuffle)
if num_ocultas2 == 0 then
    cadenared = string.format("%d inputs %d logistic %d softmax", num_entradas, num_ocultas1, num_salidas)
else
    cadenared = string.format("%d inputs %d logistic %d logistic %d softmax",
num_entradas, num_ocultas1, num_ocultas2, num_salidas)
end
lared = Mlp(cadenared) lared:generate(aleat, -0.7, 0.7)
datosentrenar = {
    learning_rate = el_learning_rate, momentum = el_momentum,
    weight_decay = weight_decay, input_dataset = train_input,
    output_dataset = train_output, shuffle = otrorand
}
datostest = {
    input_dataset = test_input,
    output_dataset = test_output,
}
-- ENTRENAMOS LA RED NEURONAL
for epoch = 1,veces_train do
    errortrain = lared:train(datosentrenar)
    errortest = lared:validate(datostest)
end
lared:save("lared.txt")

```

Figura B.2: Script en LUA para el aprendizaje de un MLP

Anexo C

Generación de respuestas

C.1. Plantillas de respuestas definidas para la adquisición del corpus DIHANA

C.1.1. No entendido

No Entendido
Puede repetir de manera clara lo último que ha dicho, por favor.
Su frase ha sido demasiado larga. Repita por favor su consulta utilizando frases más cortas.
Lo siento, creo que no le he entendido bien. Por favor, utilice frases cortas.
Lo siento, no le he entendido. Por favor, hable claro y despacio.
¿Puede repetir su consulta?
Lo siento. No he entendido bien. ¿Puede repetir?
Perdón. No consigo entenderle.

C.1.2. No válido

No válido
Este servicio no está preparado para atender consultas sobre trenes de cercanías o internacionales. ¿Desea realizar otra consulta?
Este servicio no está preparado para realizar reservas. ¿Desea realizar otra consulta?
Este servicio sólo puede ofrecerle precios sin descuentos y sin recargos. ¿Desea realizar otra consulta?
Este servicio no puede informarle sobre las paradas en el recorrido. ¿Desea realizar otra consulta?
Este servicio no está preparado para este tipo de consultas. ¿Desea realizar otra consulta?

C.1.3. Completar (confirmación/pregunta de concepto)

Horarios
¿Quiere saber horarios?
¿Le indico horarios?
¿Qué horario desea?

Ciudad
¿A qué ciudad desea viajar?
¿A qué ciudad desea llegar?

Fecha
¿Qué día desea salir?
¿Qué día desea llegar?

Tipo Tren
¿En qué tipo de tren desea viajar?

C.1.4. Confirmación explícita (de concepto y atributos)

Horarios
¿Me pregunta por horarios?
¿Ha dicho horarios?
¿Le consulto horarios?

Precios
¿Me pregunta por precios?
¿Ha dicho precios?
¿Le consulto precios?

Duración del trayecto
¿Me pregunta por la duración del trayecto?
¿Ha dicho que quiere saber la duración del trayecto?
¿Le consulto la duración del trayecto?

Tipo Tren
¿Me pregunta por tipos de trenes?
¿Ha dicho tipos de trenes?
¿Le consulto tipos de trenes?

Servicios
¿Me pregunta por servicios?
¿Ha dicho servicios?
¿Le consulto servicios?

Ciudad
¿Quiere viajar desde <CIUDAD-ORIGEN>?
¿Quiere salir desde <CIUDAD-ORIGEN>?
¿Quiere viajar a <CIUDAD-DESTINO>?
¿Quiere llegar a <CIUDAD-DESTINO>?

Fecha
El día de la semana no se corresponde con la fecha indicada, confirme por favor si ¿Desea salir <FECHASALIDA>?
El día de la semana no se corresponde con la fecha indicada, confirme por favor si ¿Desea llegar <FECHALLEGADA>?
¿Desea salir <FECHA-SALIDA>?
¿Desea llegar <FECHA-LLEGADA>?

Clase Billete
¿Desea viajar en clase <CLASE-BILLETE>?
¿Desea viajar en <CLASE-BILLETE>?

Intervalo Hora
¿Quiere salir <INTERVALO-HORA-SALIDA>?
¿Quiere llegar <INTERVALO-HORA-LLEGADA>?

Tipo Tren
¿Desea hacer el viaje en un <TIPO-TREN>?

C.1.5. Confirmación implícita

C.1.5.1. Duración

Confirmaciones
Un momento por favor.
Le consulto la duración del trayecto. Un momento por favor.
Le consulto duración de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>en <TIPO-TREN>. Un momento por favor.

Respuesta vacía
No hay ningún tren que cumpla estos requerimientos. ¿Desea cambiar alguna de las características solicitadas?

Respuestas con cardinalidad 1 (I)
El único tren que realiza el trayecto tarda <DURACION>y es un <TIPO-TREN>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>con salida <HORA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>con llegada <HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>con salida <HORA-SALIDA>, llegada <HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>con salida <FECHA-SALIDA><HORASALIDA>y tarda <DURACION>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>con llegada <FECHA-LLEGADA><HORALLEGADA>y tarda <DURACION>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>con salida <FECHA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>con llegada <FECHA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El <NUMERO-RELATIVO>tren es un <TIPO-TREN>y tarda <DURACION>. ¿Desea algo más?
El <NUMERO-RELATIVO>tren es un <TIPO-TREN>con salida <HORA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El <NUMERO-RELATIVO>tren es un <TIPO-TREN>con llegada <HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El <NUMERO-RELATIVO>tren es un <TIPO-TREN>con salida <HORA-SALIDA>, llegada <HORALLEGADA>y tarda <DURACION>. ¿Desea algo más?
El <NUMERO-RELATIVO>tren es un <TIPO-TREN>con salida <FECHA-SALIDA><HORASALIDA>y tarda <DURACION>. ¿Desea algo más?
El <NUMERO-RELATIVO>tren es un <TIPO-TREN>con llegada <FECHA-LLEGADA><HORALLEGADA>y tarda <DURACION>. ¿Desea algo más?
El <NUMERO-RELATIVO>tren es un <TIPO-TREN>con salida <FECHA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El <NUMERO-RELATIVO>tren es un <TIPO-TREN>con llegada <FECHA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más caro es un <TIPO-TREN>y tarda <DURACION>. ¿Desea algo más?
El tren más caro es un <TIPO-TREN>con salida <HORA-SALIDA>y tarda <DURACION>. ¿Desea algo más?

Respuestas con cardinalidad 1 (II)
El tren más caro es un <TIPO-TREN>con llegada <HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más caro es un <TIPO-TREN>con salida <HORA-SALIDA>, llegada <HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más caro es un <TIPO-TREN>con salida <FECHA-SALIDA><HORA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El tren más caro es un <TIPO-TREN>con llegada <FECHA-LLEGADA><HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más caro es un <TIPO-TREN>con salida <FECHA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El tren más caro es un <TIPO-TREN>con llegada <FECHA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más barato es un <TIPO-TREN>y tarda <DURACION>. ¿Desea algo más?
El tren más barato es un <TIPO-TREN>con salida <HORA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El tren más barato es un <TIPO-TREN>con llegada <HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más barato es un <TIPO-TREN>con salida <HORA-SALIDA>, llegada <HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más barato es un <TIPO-TREN>con salida <FECHA-SALIDA><HORA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El tren más barato es un <TIPO-TREN>con llegada <FECHA-LLEGADA><HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más barato es un <TIPO-TREN>con salida <FECHA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El tren más barato es un <TIPO-TREN>con llegada <FECHA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más rápido es un <TIPO-TREN>y tarda <DURACION>. ¿Desea algo más?
El tren más rápido es un <TIPO-TREN>con salida <HORA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El tren más rápido es un <TIPO-TREN>con llegada <HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más rápido es un <TIPO-TREN>con salida <HORA-SALIDA>, llegada <HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más rápido es un <TIPO-TREN>con salida <FECHA-SALIDA><HORA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El tren más rápido es un <TIPO-TREN>con llegada <FECHA-LLEGADA><HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más rápido es un <TIPO-TREN>con salida <FECHA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El tren más rápido es un <TIPO-TREN>con llegada <FECHA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más lento es un <TIPO-TREN>y tarda <DURACION>. ¿Desea algo más?
El tren más lento es un <TIPO-TREN>con salida <HORA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El tren más lento es un <TIPO-TREN>con llegada <HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más lento es un <TIPO-TREN>con salida <HORA-SALIDA>, llegada <HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más lento es un <TIPO-TREN>con salida <FECHA-SALIDA><HORA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El tren más lento es un <TIPO-TREN>con llegada <FECHA-LLEGADA><HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El tren más lento es un <TIPO-TREN>con salida <FECHA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El tren más lento es un <TIPO-TREN>con llegada <FECHA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
Ese tren tarda <DURACION>. ¿Desea algo más?

Respuestas con cardinalidad 2
Sólo hay dos trenes, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Sólo hay dos trenes con salida <INTERVALO-HORA-SALIDA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <INTERVALO-HORA-SALIDA>, los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Sólo hay dos trenes con llegada <INTERVALO-HORA-LLEGADA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con llegada <INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Sólo hay dos trenes con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Sólo hay dos trenes con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>.¿Desea algo más?
Sólo hay dos trenes con salida <FECHA-SALIDA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <FECHA-SALIDA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Sólo hay dos trenes con llegada <FECHA-LLEGADA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con llegada <FECHA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Sólo hay dos trenes con salida <INTERVALO-HORA-SALIDA>y llegada <INTERVALO-HORA-LLEGADA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <INTERVALO-HORA-SALIDA>y llegada <INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>.¿Desea algo más?

Respuestas con cardinalidad mayor a 2 (I)
Hay varios trenes. El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Hay varios trenes con salida <INTERVALO-HORA-SALIDA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <INTERVALO-HORA-SALIDA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Hay varios trenes con llegada <INTERVALO-HORA-LLEGADA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con llegada <INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Hay varios trenes con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?

Respuestas con cardinalidad mayor a 2 (II)
Hay varios trenes con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>. El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Hay varios trenes con salida <FECHA-SALIDA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <FECHA-SALIDA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Hay varios trenes con llegada <FECHA-LLEGADA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con llegada <FECHA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Hay varios trenes con salida <INTERVALO-HORA-SALIDA>y llegada <INTERVALO-HORA-LLEGADA>. El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <INTERVALO-HORA-SALIDA>y llegada <INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?

C.1.5.2. Horarios

Confirmaciones
Le consulto horarios. Un momento por favor.
Le consulto horarios de trenes con salida en <CIUDAD-ORIGEN>destino <CIUDAD-DESTINO>saliendo <FECHA-SALIDA>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO><FECHA-SALIDA><INTERVALO-HORA-SALIDA>. Un momento por favor.
Le consulto horarios de trenes con salida en <CIUDAD-ORIGEN>destino <CIUDAD-DESTINO>saliendo <FECHA-SALIDA>en <TIPO-TREN>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO><FECHA-SALIDA><INTERVALO-HORA-SALIDA>en <TIPO-TREN>. Un momento por favor.
Le consulto horarios de trenes con salida en <CIUDAD-ORIGEN>destino <CIUDAD-DESTINO>saliendo <FECHA-SALIDA>en clase <CLASE-BILLETE>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO><FECHA-SALIDA><INTERVALO-HORA-SALIDA>en clase <CLASE-BILLETE>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>llegando <FECHA-LLEGADA>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>llegando <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>llegando <FECHA-LLEGADA>en <TIPO-TREN>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>llegando <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>en <TIPO-TREN>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>llegando <FECHA-LLEGADA>en clase <CLASE-BILLETE>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>llegando <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>en clase <CLASE-BILLETE>. Un momento por favor.

Respuesta vacía
No hay ningún tren que cumpla estos requerimientos. ¿Desea cambiar alguna de las características solicitadas?

Respuesta con cardinalidad 1
El único tren que realiza el trayecto es un <TIPO-TREN>que sale <HORA-SALIDA>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>que llega <HORA-LLEGADA>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>que sale <HORA-SALIDA>y llega <HORALLEGADA>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>que sale <FECHA-SALIDA><HORASALIDA>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>que llega <FECHA-LLEGADA><HORALLEGADA>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>que sale <FECHA-SALIDA>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>que llega <FECHA-LLEGADA>. ¿Desea algo más?
El <NUMERO-RELATIVO>tren es un <TIPO-TREN>que sale <HORA-SALIDA>y llega <HORALLEGADA>. ¿Desea algo más?
El <NUMERO-RELATIVO>tren es un <TIPO-TREN>que llega <HORA-LLEGADA>. ¿Desea algo más?
El tren más caro es un <TIPO-TREN>que sale <HORA-SALIDA>y llega <HORA-LLEGADA>. ¿Desea algo más?
El tren más barato es un <TIPO-TREN>que sale <HORA-SALIDA>y llega <HORA-LLEGADA>. ¿Desea algo más?
El tren más rápido es un <TIPO-TREN>que sale <HORA-SALIDA>y llega <HORA-LLEGADA>. ¿Desea algo más?
El tren más lento es un <TIPO-TREN>que sale <HORA-SALIDA>y llega <HORA-LLEGADA>. ¿Desea algo más?
Ese tren sale <HORA-SALIDA>y llega <HORA-LLEGADA>. ¿Desea algo más?
Ese tren sale <FECHA-SALIDA>. ¿Desea algo más?
Ese tren llega <FECHA-LLEGADA>. ¿Desea algo más?

Respuestas con cardinalidad 2
Sólo hay dos trenes. El primero sale <HORA-SALIDA>y el otro <HORA-SALIDA>. ¿Desea algo más?
Sólo hay dos trenes. El primero sale <HORA-SALIDA>y llega <HORA-LLEGADA>y el otro sale <HORA-SALIDA>y llega <HORA-LLEGADA>. ¿Desea algo más?
Sólo hay dos trenes con salida <FECHA-SALIDA>. El primero sale <HORA-SALIDA>y el otro <HORA-SALIDA>. ¿Desea algo más?
Sólo hay dos trenes con llegada <FECHA-LLEGADA>. El primero llega <HORA-LLEGADA>y el otro <HORA-LLEGADA>. ¿Desea algo más?
Sólo hay dos trenes. El más rápido sale <HORA-SALIDA>y el más lento <HORA-SALIDA>. ¿Desea algo más?
Sólo hay dos trenes. El más rápido sale <HORA-SALIDA>y llega <HORA-LLEGADA>y el más lento sale <HORA-SALIDA>y llega <HORA-LLEGADA>. ¿Desea algo más?
Sólo hay dos trenes con salida <FECHA-SALIDA>. El más rápido sale <HORA-SALIDA>y el más lento <HORA-SALIDA>. ¿Desea algo más?
Sólo hay dos trenes que salen <FECHA-SALIDA>. ¿Desea algo más?
Sólo hay dos trenes que llegan <FECHA-LLEGADA>. ¿Desea algo más?
Sólo hay dos trenes. El primero llega <HORA-LLEGADA>y el otro <HORA-LLEGADA>. ¿Desea algo más?
Sólo hay dos trenes. El más rápido llega <HORA-LLEGADA>y el más lento <HORA-LLEGADA>. ¿Desea algo más?
Sólo hay dos trenes con llegada <FECHA-LLEGADA>. El más rápido llega <HORA-LLEGADA>y el más lento <HORA-LLEGADA>. ¿Desea algo más?

Respuestas con cardinalidad mayor de 2
Hay varios trenes. El primero sale <HORA-SALIDA>y el último <HORA-SALIDA>. ¿Desea algo más?
Hay varios trenes. El primero sale <HORA-SALIDA>y llega <HORA-LLEGADA>y el último sale <HORA-SALIDA>y llega <HORA-LLEGADA>. ¿Desea algo más?
Hay varios trenes con salida <FECHA-SALIDA>. El primero sale <HORA-SALIDA>y el último <HORA-SALIDA>. ¿Desea algo más?
Hay varios trenes con llegada <FECHA-LLEGADA>. El primero llega <HORA-LLEGADA>y el último <HORA-LLEGADA>. ¿Desea algo más?
Hay varios trenes que salen <FECHA-SALIDA>. ¿Desea algo más?
Hay varios trenes que llegan <FECHA-LLEGADA>. ¿Desea algo más?
Hay varios trenes. El más rápido sale <HORA-SALIDA>y el más lento <HORA-SALIDA>. ¿Desea algo más?
Hay varios trenes. El más rápido sale <HORA-SALIDA>y llega <HORA-LLEGADA>y el más lento sale <HORA-SALIDA>y llega <HORA-LLEGADA>. ¿Desea algo más?
Hay varios trenes con salida <FECHA-SALIDA>. El más rápido sale <HORA-SALIDA>y el más lento <HORA-SALIDA>. ¿Desea algo más?
Hay varios trenes con llegada <FECHA-LLEGADA>. El más rápido llega <HORA-LLEGADA>y el más lento <HORA-LLEGADA>. ¿Desea algo más?
Hay varios trenes. El primero llega <HORA-LLEGADA>y el último <HORA-LLEGADA>. ¿Desea algo más?
Hay varios trenes. El más rápido llega <HORA-LLEGADA>y el más lento <HORA-LLEGADA>. ¿Desea algo más?

C.1.5.3. Precio

Confirmaciones
Le consulto precios. Un momento por favor.
Le consulto precios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>. Un momento por favor.
Le consulto precios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>saliendo <FECHA-SALIDA>. Un momento por favor.
Le consulto precios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>en clase <CLASE-BILLETE>. Un momento por favor.
Le consulto precios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>en tipo tren <TIPO-TREN>. Un momento por favor.
Le consulto precios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>llegando <FECHA-LLEGADA>. Un momento por favor.

Respuesta vacía
No hay ningún tren que cumpla estos requerimientos. ¿Desea cambiar alguna de las características solicitadas?

Respuesta con cardinalidad 1 (I)
El único tren que realiza el trayecto es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
El único tren que sale <INTERVALO-HORA-SALIDA>y realiza el trayecto es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
El único tren que llega <INTERVALO-HORA-LLEGADA>y realiza el trayecto es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Respuesta con cardinalidad 1 (II)

El único tren con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

El único tren con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

El único tren que sale <FECHA-SALIDA>y realiza el trayecto es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

El único tren con llegada <FECHA-LLEGADA>es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

El único tren con salida <INTERVALO-HORA-SALIDA>y llegada <INTERVALO-HORA-LLEGADA>es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

El <NUMERO-RELATIVO>tren es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>el <NUMERO-RELATIVO>tren es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA>el <NUMERO-RELATIVO>tren es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>el <NUMERO-RELATIVO>tren es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA>el <NUMERO-RELATIVO>tren es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA>el <NUMERO-RELATIVO>tren es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <INTERVALO-HORA-LLEGADA>el <NUMERO-RELATIVO>tren es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

El tren más caro es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>el tren más caro es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA>el tren más caro es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>el tren más caro es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Respuesta con cardinalidad 1 (III)

Con llegada <FECHA-LLEGADA>el tren más caro es un <TIPO-TREN>y su precio en clase <CLASEBILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA>el tren más caro es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <INTERVALO-HORA-LLEGADA>el tren más caro es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

El tren más barato es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>el tren más barato es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA>el tren más barato es un <TIPO-TREN>y su precio en clase <CLASEBILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>el tren más barato es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA>el tren más barato es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA>el tren más barato es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <INTERVALO-HORA-LLEGADA>el tren más barato es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

El tren más rápido es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>el tren más rápido es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA>el tren más rápido es un <TIPO-TREN>y su precio en clase <CLASEBILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>el tren más rápido es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA>el tren más rápido es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA>el tren más rápido es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Respuesta con cardinalidad 1 (IV)

Con llegada <INTERVALO-HORA-LLEGADA>el tren más rápido es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

El tren más lento es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>el tren más lento es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA>el tren más lento es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>el tren más lento es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA>el tren más lento es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA>el tren más lento es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <INTERVALO-HORA-LLEGADA>el tren más lento es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

El precio de ese tren en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Ese tren dispone de clase <CLASE-BILLETE>. ¿Desea algo más?

Respuestas con cardinalidad 2 (I)

Sólo hay dos trenes. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el otro es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Los trenes son <TIPO-TREN>y en clase <CLASE-BILLETE>, el billete cuesta <PRECIO>euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Sólo hay dos trenes con salida <INTERVALO-HORA-SALIDA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el otro es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA>los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO>euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Sólo hay dos trenes con llegada <INTERVALO-HORA-LLEGADA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el otro es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO>euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Respuestas con cardinalidad 2 (III)
Con salida <FECHA-SALIDA>sólo hay dos trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el otro es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Con llegada <FECHA-LLEGADA>sólo hay dos trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el otro es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Con salida <INTERVALO-HORA-SALIDA>y llegada <INTERVALO-HORA-LLEGADA>sólo hay dos trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el otro es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Los trenes anteriores disponen de clase <CLASE-BILLETE>. ¿Desea algo más?

Respuestas con cardinalidad mayor a 2 (I)
Hay varios trenes. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el más barato es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Los trenes son <TIPO-TREN>y en clase <CLASE-BILLETE>, el billete cuesta <PRECIO>euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Hay varios trenes con salida <INTERVALO-HORA-SALIDA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el más barato es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Con salida <INTERVALO-HORA-SALIDA>los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO>euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Hay varios trenes con llegada <INTERVALO-HORA-LLEGADA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el más barato es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Con llegada <INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO>euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Hay varios trenes con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el más barato es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO>euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Hay varios trenes con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el más barato es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO>euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Hay varios trenes con salida <FECHA-SALIDA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el más barato es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Respuestas con cardinalidad mayor a 2 (II)

Hay varios trenes con llegada <FECHA-LLEGADA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más barato es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA> los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO> euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA> los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO> euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Hay varios trenes con salida <INTERVALO-HORA-SALIDA> y llegada <INTERVALO-HORA-LLEGADA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más barato es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA> y llegada <INTERVALO-HORA-LLEGADA> los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO> euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más lento es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA> hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más lento es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <INTERVALO-HORA-LLEGADA> hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más lento es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA> <INTERVALO-HORA-SALIDA> hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más lento es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA> <INTERVALO-HORA-LLEGADA> hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más lento es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA> hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más lento es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA> hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más lento es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA> y llegada <INTERVALO-HORA-LLEGADA> hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más lento es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Los trenes anteriores disponen de clase <CLASE-BILLETE>. ¿Desea algo más?

C.1.5.4. Servicios

Confirmaciones
Le consulto servicios. Un momento por favor.
Le consulto servicios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>en un tren <TIPO-TREN>. Un momento por favor.

Respuesta vacía
No hay ningún tren que cumpla estos requerimientos. ¿Desea cambiar alguna de las características solicitadas?

Respuestas con cardinalidad 1
El único tren que realiza el trayecto es un <TIPO-TREN>y dispone de los siguientes servicios: <SERVICIOS>. ¿Desea algo más?
El <NUMERO-RELATIVO>tren es un <TIPO-TREN>y dispone de los siguientes servicios: <SERVICIOS>. ¿Desea algo más?
El tren más caro es un <TIPO-TREN>y dispone de los siguientes servicios: <SERVICIOS>. ¿Desea algo más?
El tren más barato es un <TIPO-TREN>y dispone de los siguientes servicios: <SERVICIOS>. ¿Desea algo más?
El tren más rápido es un <TIPO-TREN>y dispone de los siguientes servicios: <SERVICIOS>. ¿Desea algo más?
El tren más lento es un <TIPO-TREN>y dispone de los siguientes servicios: <SERVICIOS>. ¿Desea algo más?
Ese tren sólo dispone de los siguientes servicios: <SERVICIOS>. ¿Desea algo más?
Se dispone de los siguientes servicios : <SERVICIOS>. ¿Desea algo más?

Respuestas con cardinalidad 2
Hay dos trenes. El tren más rápido es un <TIPO-TREN>con <SERVICIOS>y el otro es un <TIPO-TREN>con <SERVICIOS>. ¿Desea algo más?
Hay un único tipo de tren, que es un <TIPO-TREN>con <SERVICIOS>. ¿Desea algo más?
Se dispone de los siguientes servicios : <SERVICIOS>. ¿Desea algo más?

Respuestas con cardinalidad mayor a 2
Hay varios trenes. El tren más rápido es un <TIPO-TREN>con <SERVICIOS>y el más lento es un <TIPO-TREN>con <SERVICIOS>. ¿Desea algo más?
Hay un único tipo de tren, que es un <TIPO-TREN>con <SERVICIOS>. ¿Desea algo más?
Se dispone de los siguientes servicios : <SERVICIOS>. ¿Desea algo más?

C.1.5.5. Tipo de tren

Confirmaciones
Le consulto tipos de tren. Un momento por favor.
Le consulto tipos de trenes para el trayecto <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>. Un momento por favor.

Respuesta vacía
No hay ningún tren que cumpla estos requerimientos. ¿Desea cambiar alguna de las características solicitadas?

Respuestas con cardinalidad 1
El único tren que realiza el trayecto es un <TIPO-TREN>. ¿Desea algo más?
El <NUMERO-RELATIVO>tren es un <TIPO-TREN>. ¿Desea algo más?
El tren más caro es un <TIPO-TREN>. ¿Desea algo más?
El tren más barato es un <TIPO-TREN>. ¿Desea algo más?
El tren más rápido es un <TIPO-TREN>. ¿Desea algo más?
El tren más lento es un <TIPO-TREN>. ¿Desea algo más?
Ese tren es un <TIPO-TREN>. ¿Desea algo más?

Respuestas con cardinalidad 2
Dispone de dos tipos de trenes. El tren más rápido es un <TIPO-TREN>y el otro es un <TIPO-TREN>. ¿Desea algo más?
Hay un único tipo de tren, que es un <TIPO-TREN>. ¿Desea algo más?
Dispone de dos tipos de trenes con salida <FECHA-SALIDA>. El tren más rápido es un <TIPO-TREN>y el otro es un <TIPO-TREN>. ¿Desea algo más?
Dispone de un único tipo de tren con salida <FECHA-SALIDA>, que es un <TIPO-TREN>. ¿Desea algo más?
Dispone de dos tipos de trenes con llegada <FECHA-LLEGADA>. El tren más rápido es un <TIPO-TREN>y el otro es un <TIPO-TREN>. ¿Desea algo más?
Dispone de un único tipo de tren con llegada <FECHA-LLEGADA>, que es un <TIPO-TREN>. ¿Desea algo más?
Los trenes anteriores son <TIPO-TREN>. ¿Desea algo más?

Respuestas con cardinalidad mayor a 2
Hay varios tipos de trenes. El tren más rápido es un <TIPO-TREN>y el más lento es un <TIPO-TREN>. ¿Desea algo más?
Hay un único tipo de tren, que es un <TIPO-TREN>. ¿Desea algo más?
Hay varios tipos de trenes con salida <FECHA-SALIDA>, El tren más rápido es un <TIPO-TREN>y el más lento es un <TIPO-TREN>. ¿Desea algo más?
Dispone de un único tipo de tren con salida <FECHA-SALIDA>, que es un <TIPO-TREN>. ¿Desea algo más?
Hay varios tipos de trenes con llegada <FECHA-LLEGADA>, El tren más rápido es un <TIPO-TREN>y el más lento es un <TIPO-TREN>. ¿Desea algo más?
Dispone de un único tipo de tren con llegada <FECHA-LLEGADA>, que es un <TIPO-TREN>. ¿Desea algo más?
Los trenes anteriores son <TIPO-TREN>. ¿Desea algo más?

C.1.6. Confirmación mixta Seguro

C.1.6.1. Concepto Horario

Horarios
Quiere horarios de trenes a <CIUDAD-DESTINO>, ...
Horarios de trenes con destino a <CIUDAD-DESTINO>, ...
Quiere horarios saliendo <FECHA-SALIDA>, ...
Quiere horarios llegando <FECHA-LLEGADA>, ...
Horarios a <CIUDAD-DESTINO>, con llegada <FECHA-LLEGADA>, ...

C.1.6.2. Concepto Precio

Precios
Quiere saber el precio a <CIUDAD-DESTINO>, ...
Veamos, el precio de un tren con destino a <CIUDAD-DESTINO>, ...
Quiere precios saliendo <FECHA-SALIDA>, ...
Quiere precios llegando <FECHA-LLEGADA>, ...

C.1.6.3. Concepto Duración

Duración
Pregunta por la duración del trayecto a <CIUDAD-DESTINO>, ...
Veamos, lo que tarda en llegar a <CIUDAD-DESTINO>, ...

C.1.6.4. Frame Tipo de tren

Tipo Tren
¿Desea saber tipos de trenes a <CIUDAD-DESTINO>, ...
Veamos, tipos de trenes a <CIUDAD-DESTINO>, ...

C.1.6.5. Concepto Servicios

Servicios
¿Desea saber los servicios en <TIPO-TREN>, ...
Veamos, servicios de trenes <TIPO-TREN>, ...

C.1.6.6. Segunda parte de la confirmación mixta

Horarios, Duración, Tipo Tren o Servicios
..., ¿desde <CIUDAD-ORIGEN>?
..., ¿saliendo de <CIUDAD-ORIGEN>?

Concepto Precio
..., ¿desde <CIUDAD-ORIGEN>?
..., ¿saliendo de <CIUDAD-ORIGEN>?
..., ¿en <CLASE-BILLETE>?
..., ¿en clase <CLASE-BILLETE>?

C.1.7. Confirmación mixta Concepto Inseguro

C.1.7.1. Primera parte de la confirmación

Primera parte de la confirmación
Saliendo desde <CIUDAD-ORIGEN>, ...

C.1.7.2. Segunda parte de la confirmación

Horarios
..., ¿me pregunta por horarios?
..., ¿ha dicho horarios?

Precio
..., ¿me pregunta por precios?
..., ¿ha dicho precios?

Duración
..., ¿me pregunta por la duración del trayecto?
..., ¿ha dicho que quiere saber la duración del trayecto?

Tipo Tren
..., ¿me pregunta por tipos de trenes?
..., ¿ha dicho tipos de trenes?

Servicios
..., ¿me pregunta por servicios?
..., ¿ha dicho servicios?

C.1.8. Concepto Seguro Hora, Precio, Duración, Tipo de tren o Servicios

C.1.8.1. Primera parte de la confirmación

Horarios
Horarios para llegar a <CIUDAD-DESTINO>, ...
Horarios saliendo <FECHA-SALIDA>, ...
Horarios llegando <FECHA-LLEGADA>, ...
Horarios desde <CIUDAD-ORIGEN>, ...
Horarios, ...

Precios
Precios desde <CIUDAD-ORIGEN>, ...
Precios para llegar a <CIUDAD-DESTINO>, ...
Precios en clase <CLASE-BILLETE>, ...
Precios saliendo <FECHA-SALIDA>, ...
Precios llegando <FECHA-LLEGADA>, ...
Precios, ...

Duración
Duración del trayecto desde <CIUDAD-ORIGEN>, ...
Duración del trayecto para llegar a <CIUDAD-DESTINO>, ...
Duración del trayecto, ...

Tipo Tren
Tipo de tren desde <CIUDAD-ORIGEN>, ...
Tipo de tren para llegar a <CIUDAD-DESTINO>, ...
Tipos de trenes, ...

Servicios
Servicios en tipo <TIPO-TREN>, ...
Veamos, me ha dicho servicios, ...

C.1.8.2. Segunda parte de la confirmación

Precios
..., ¿ quiere salir de <CIUDAD-ORIGEN>?
..., ¿ quiere llegar a <CIUDAD-DESTINO>?
..., ¿ ¿Desea salir <FECHA-SALIDA>?
..., ¿ ¿Desea llegar <FECHA-LLEGADA>?
..., ¿ quiere salir <INTERVALO-HORA-SALIDA>?
..., ¿ quiere llegar <INTERVALO-HORA-LLEGADA>?
..., ¿ ¿Desea viajar en clase <CLASE-BILLETE>?
..., ¿ ¿Desea hacer el viaje en un <TIPO-TREN>?

C.1.9. Nueva Consulta

Precios
No. ¿Desea algo más?
Sí. ¿Desea algo más?

C.2. Plantillas de respuestas definidas tras el etiquetado del corpus.

C.2.1. Preguntas

Pregunta-[Origen]
¿De qué ciudad desea partir?
¿Desde que ciudad quiere salir?

Pregunta-[Destino]
¿A qué ciudad desea viajar?
¿A qué ciudad desea llegar?

Pregunta-[Día]
¿Qué día desea salir?
¿Qué día desea llegar?

C.2.2. Confirmaciones explícitas

C.2.2.1. Conceptos

Confirmación-[Hora_salida]
¿Me pregunta por horarios?
¿Ha dicho horarios?
¿Le consulto horarios?

Confirmación-[Precio]
¿Me pregunta por precios?
¿Ha dicho precios?
¿Le consulto precios?

Confirmación-[Tipo_tren]
¿Me pregunta por tipos de trenes?
¿Ha dicho tipos de trenes?
¿Le consulto tipos de trenes?

C.2.2.2. Atributos

Confirmación-Origen
¿Quiere viajar desde <CIUDAD-ORIGEN>?
¿Quiere salir desde <CIUDAD-ORIGEN>?

Confirmación-Destino
¿Quiere viajar a <CIUDAD-DESTINO>?
¿Quiere llegar a <CIUDAD-DESTINO>?

Confirmación-Día
¿Desea salir <FECHA-SALIDA>?
¿Desea llegar <FECHA-LLEGADA>?

Confirmación-Tipo_tren
¿Desea hacer el viaje en un <TIPO-TREN>?

Confirmación-Clase_billete
¿Desea viajar en clase <CLASE-BILLETE>?
¿Desea viajar en <CLASE-BILLETE>?

Confirmación-Hora_salida
¿Quiere salir <INTERVALO-HORA-SALIDA>?

Confirmación-Hora_llegada
¿Quiere llegar <INTERVALO-HORA-LLEGADA>?

Confirmación-Precio
¿Me he dicho un tren con precio <PRECIO>?
¿Desea un billete <PRECIO>?

C.2.3. Confirmaciones mixtas

Confirmación,Confirmación-Hora_salida,Tipo_tren
Horarios para llegar a <CIUDAD-DESTINO>, ...
Horarios saliendo <FECHA-SALIDA>, ...
Horarios llegando <FECHA-LLEGADA>, ...
Horarios desde <CIUDAD-ORIGEN>, ...
..., ¿desea hacer el viaje en un <TIPO-TREN>?

Confirmación,Confirmación-Hora_salida,Día
Horarios para llegar a <CIUDAD-DESTINO>, ...
Horarios saliendo <FECHA-SALIDA>, ...
Horarios llegando <FECHA-LLEGADA>, ...
Horarios desde <CIUDAD-ORIGEN>, ...
..., ¿desea salir <FECHA-SALIDA>?
..., ¿desea llegar <FECHA-LLEGADA>?

Confirmación,Confirmación-Precio,Día
Precios desde <CIUDAD-ORIGEN>, ...
Precios para llegar a <CIUDAD-DESTINO>, ...
Precios en clase <CLASE-BILLETE>, ...
Precios saliendo <FECHA-SALIDA>, ...
Precios llegando <FECHA-LLEGADA>, ...
..., ¿desea salir <FECHA-SALIDA>?
..., ¿desea llegar <FECHA-LLEGADA>?

Confirmación,Confirmación-Precio,Hora_llegada
Precios desde <CIUDAD-ORIGEN>, ...
Precios para llegar a <CIUDAD-DESTINO>, ...
Precios n clase <CLASE-BILLETE>, ...
Precios saliendo <FECHA-SALIDA>, ...
Precios llegando <FECHA-LLEGADA>, ...
..., ¿desea salir <FECHA-SALIDA>?
..., ¿desea llegar <FECHA-LLEGADA>?

Confirmación,Confirmación-Precio,Origen
Quiere saber el precio a <CIUDAD-DESTINO>, ...
Veamos, el precio de un tren con destino a <CIUDAD-DESTINO>, ...
Quiere precios saliendo <FECHA-SALIDA>, ...
Quiere precios llegando <FECHA-LLEGADA>, ...
..., ¿desde <CIUDAD-ORIGEN>?
..., ¿saliendo de <CIUDAD-ORIGEN>?

Confirmación,Confirmación-Precio,Destino
Precios desde <CIUDAD-ORIGEN>, ...
Precios en clase <CLASE-BILLETE>, ...
Precios saliendo <FECHA-SALIDA>, ...
Precios llegando <FECHA-LLEGADA>, ...
..., ¿ quiere llegar a <CIUDAD-DESTINO>?

Confirmación,Confirmación-Hora _ salida,Origen
Horarios para llegar a <CIUDAD-DESTINO>, ...
Horarios saliendo <FECHA-SALIDA>, ...
Horarios llegando <FECHA-LLEGADA>, ...
..., ¿ quiere salir de <CIUDAD-ORIGEN>?

C.2.4. Respuestas a consultas a la base de datos

C.2.4.1. Respuesta-Hora _ salida

Respuesta vacía
No hay ningún tren que cumpla estos requerimientos. ¿Desea cambiar alguna de las características solicitadas?

Respuestas con cardinalidad 1
El único tren que realiza el trayecto es un <TIPO-TREN>que sale <HORA-SALIDA>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>que sale <HORA-SALIDA>y llega <HORALLEGADA>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>que sale <FECHA-SALIDA><HORASALIDA>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>que sale <FECHA-SALIDA>. ¿Desea algo más?

Respuestas con cardinalidad 2
Sólo hay dos trenes. El primero sale <HORA-SALIDA>y el otro <HORA-SALIDA>. ¿Desea algo más?
Sólo hay dos trenes. El primero sale <HORA-SALIDA>y llega <HORA-LLEGADA>y el otro sale <HORA-SALIDA>y llega <HORA-LLEGADA>. ¿Desea algo más?
Sólo hay dos trenes con salida <FECHA-SALIDA>. El primero sale <HORA-SALIDA>y el otro <HORA-SALIDA>. ¿Desea algo más?

Respuestas con cardinalidad mayor de 2
Hay varios trenes. El primero sale <HORA-SALIDA>y el último <HORA-SALIDA>. ¿Desea algo más?
Hay varios trenes. El primero sale <HORA-SALIDA>y llega <HORA-LLEGADA>y el último sale <HORA-SALIDA>y llega <HORA-LLEGADA>. ¿Desea algo más?
Hay varios trenes con salida <FECHA-SALIDA>. El primero sale <HORA-SALIDA>y el último <HORA-SALIDA>. ¿Desea algo más?

C.2.4.2. Confirmación,Respuesta-[Hora_salida],Hora_salida

Confirmación,Respuesta-[Hora_salida],Hora_salida
Le consulto horarios. Un momento por favor
Conjunto de respuestas Respuesta-Hora_salida

C.2.4.3. Confirmación,Respuesta-Hora_salida,Hora_salida

Confirmación,Respuesta-Hora_salida,Hora_salida
Le consulto horarios de trenes con salida en <CIUDAD-ORIGEN>destino <CIUDAD-DESTINO>saliendo <FECHA-SALIDA>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO><FECHA-SALIDA><INTERVALO-HORA-SALIDA>. Un momento por favor.
Le consulto horarios de trenes con salida en <CIUDAD-ORIGEN>destino <CIUDAD-DESTINO>saliendo <FECHA-SALIDA>en <TIPO-TREN>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO><FECHA-SALIDA><INTERVALO-HORA-SALIDA>en <TIPO-TREN>. Un momento por favor.
Le consulto horarios de trenes con salida en <CIUDAD-ORIGEN>destino <CIUDAD-DESTINO>saliendo <FECHA-SALIDA>en clase <CLASE-BILLETE>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO><FECHA-SALIDA><INTERVALO-HORA-SALIDA>en clase <CLASE-BILLETE>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>llegando <FECHA-LLEGADA>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>llegando <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>llegando <FECHA-LLEGADA>en <TIPO-TREN>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>llegando <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>en <TIPO-TREN>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>llegando <FECHA-LLEGADA>en clase <CLASE-BILLETE>. Un momento por favor.
Le consulto horarios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>llegando <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>en clase <CLASE-BILLETE>. Un momento por favor.
Conjunto de respuestas definido para Respuesta-Hora_salida

C.2.4.4. Respuesta-Hora_llegada

Respuesta vacía
No hay ningún tren que cumpla estos requerimientos. ¿Desea cambiar alguna de las características solicitadas?

Respuesta con cardinalidad 1
El único tren que realiza el trayecto es un <TIPO-TREN>que llega <HORA-LLEGADA>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>que sale <HORA-SALIDA>y llega <HORALLEGADA>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>que llega <FECHA-LLEGADA><HORALLEGADA>. ¿Desea algo más?

Respuestas con cardinalidad 2
Sólo hay dos trenes. El primero sale <HORA-SALIDA>y llega <HORA-LLEGADA>y el otro sale <HORA-SALIDA>y llega <HORA-LLEGADA>. ¿Desea algo más?
Sólo hay dos trenes con llegada <FECHA-LLEGADA>. El primero llega <HORA-LLEGADA>y el otro <HORA-LLEGADA>. ¿Desea algo más?
Sólo hay dos trenes. El primero llega <HORA-LLEGADA>y el otro <HORA-LLEGADA>. ¿Desea algo más?

Respuestas con cardinalidad mayor de 2
Hay varios trenes. El primero sale <HORA-SALIDA>y llega <HORA-LLEGADA>y el último sale <HORA-SALIDA>y llega <HORA-LLEGADA>. ¿Desea algo más?
Hay varios trenes con llegada <FECHA-LLEGADA>. El primero llega <HORA-LLEGADA>y el último <HORA-LLEGADA>. ¿Desea algo más?
Hay varios trenes. El primero llega <HORA-LLEGADA>y el último <HORA-LLEGADA>. ¿Desea algo más?

C.2.4.5. Confirmación,Respuesta-[Hora_llegada],Hora_llegada

Confirmación,Respuesta-[Hora_llegada],Hora_llegada
Le consulto horarios. Un momento por favor.
Conjunto de respuestas definido para Respuesta-Hora_llegada

C.2.4.6. Respuesta-Precio

Respuesta vacía
No hay ningún tren que cumpla estos requerimientos. ¿Desea cambiar alguna de las características solicitadas?

Respuesta con cardinalidad 1 (I)
El único tren que realiza el trayecto es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
El único tren que sale <INTERVALO-HORA-SALIDA>y realiza el trayecto es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
El único tren que llega <INTERVALO-HORA-LLEGADA>y realiza el trayecto es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
El único tren con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
El único tren con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
El único tren que sale <FECHA-SALIDA>y realiza el trayecto es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
El único tren con llegada <FECHA-LLEGADA>es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
El único tren con salida <INTERVALO-HORA-SALIDA>y llegada <INTERVALO-HORA-LLEGADA>es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Respuesta con cardinalidad 1 (III)

Con salida <FECHA-SALIDA>el tren más barato es un <TIPO-TREN>y su precio en clase <CLASEBILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>el tren más barato es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA>el tren más barato es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA>el tren más barato es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <INTERVALO-HORA-LLEGADA>el tren más barato es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

El tren más rápido es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>el tren más rápido es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA>el tren más rápido es un <TIPO-TREN>y su precio en clase <CLASEBILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>el tren más rápido es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA>el tren más rápido es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA>el tren más rápido es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <INTERVALO-HORA-LLEGADA>el tren más rápido es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

El tren más lento es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>el tren más lento es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA>el tren más lento es un <TIPO-TREN>y su precio en clase <CLASEBILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>el tren más lento es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Respuesta con cardinalidad 1 (IV)

Con llegada <FECHA-LLEGADA>el tren más lento es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA>el tren más lento es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <INTERVALO-HORA-LLEGADA>el tren más lento es un <TIPO-TREN>y su precio en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

El precio de ese tren en clase <CLASE-BILLETE>es <PRECIO>euros el billete de ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Ese tren dispone de clase <CLASE-BILLETE>. ¿Desea algo más?

Respuestas con cardinalidad 2 (I)

Sólo hay dos trenes. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el otro es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Los trenes son <TIPO-TREN>y en clase <CLASE-BILLETE>, el billete cuesta <PRECIO>euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Sólo hay dos trenes con salida <INTERVALO-HORA-SALIDA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el otro es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA>los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO>euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Sólo hay dos trenes con llegada <INTERVALO-HORA-LLEGADA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el otro es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO>euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Sólo hay dos trenes con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el otro es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO>euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Sólo hay dos trenes con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el otro es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO>euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Respuestas con cardinalidad 2 (II)

Sólo hay dos trenes con salida <FECHA-SALIDA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el otro es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA> los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO> euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Sólo hay dos trenes con llegada <FECHA-LLEGADA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el otro es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA> los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO> euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Sólo hay dos trenes con salida <INTERVALO-HORA-SALIDA> y llegada <INTERVALO-HORA-LLEGADA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el otro es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA> y llegada <INTERVALO-HORA-LLEGADA> los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO> euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Sólo hay dos trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el otro es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA> sólo hay dos trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el otro es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <INTERVALO-HORA-LLEGADA> sólo hay dos trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el otro es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA> sólo hay dos trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el otro es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA> sólo hay dos trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el otro es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA> sólo hay dos trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el otro es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA> sólo hay dos trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el otro es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA> y llegada <INTERVALO-HORA-LLEGADA> sólo hay dos trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el otro es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Los trenes anteriores disponen de clase <CLASE-BILLETE>. ¿Desea algo más?

Respuestas con cardinalidad mayor a 2 (I)

Hay varios trenes. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más barato es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Los trenes son <TIPO-TREN> y en clase <CLASE-BILLETE>, el billete cuesta <PRECIO> euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Hay varios trenes con salida <INTERVALO-HORA-SALIDA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más barato es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <INTERVALO-HORA-SALIDA> los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO> euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Hay varios trenes con llegada <INTERVALO-HORA-LLEGADA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más barato es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <INTERVALO-HORA-LLEGADA> los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO> euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Hay varios trenes con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más barato es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA> los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO> euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Hay varios trenes con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más barato es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA> los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO> euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Hay varios trenes con salida <FECHA-SALIDA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más barato es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Hay varios trenes con llegada <FECHA-LLEGADA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más barato es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con salida <FECHA-SALIDA> los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO> euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Con llegada <FECHA-LLEGADA> los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO> euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Hay varios trenes con salida <INTERVALO-HORA-SALIDA> y llegada <INTERVALO-HORA-LLEGADA>. En clase <CLASE-BILLETE>, el tren más caro es un <TIPO-TREN> que cuesta <PRECIO> euros la ida y el más barato es un <TIPO-TREN> que cuesta <PRECIO> euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

Respuestas con cardinalidad mayor a 2 (II)
Con salida <INTERVALO-HORA-SALIDA>y llegada <INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>. En clase <CLASE-BILLETE>, el billete cuesta <PRECIO>euros la ida. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el más lento es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Con salida <INTERVALO-HORA-SALIDA>hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el más lento es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Con llegada <INTERVALO-HORA-LLEGADA>hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el más lento es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el más lento es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el más lento es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Con salida <FECHA-SALIDA>hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el más lento es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Con llegada <FECHA-LLEGADA>hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el más lento es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?
Con salida <INTERVALO-HORA-SALIDA>y llegada <INTERVALO-HORA-LLEGADA>hay varios trenes. En clase <CLASE-BILLETE>, el tren más rápido es un <TIPO-TREN>que cuesta <PRECIO>euros la ida y el más lento es un <TIPO-TREN>que cuesta <PRECIO>euros. Recuerde que el billete de ida y vuelta siempre supone un ahorro del 20 por ciento. ¿Desea algo más?

C.2.4.7. Confirmación, Respuesta-Precio, Precio

Confirmaciones
Le consulto precios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>. Un momento por favor.
Le consulto precios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>saliendo <FECHA-SALIDA>. Un momento por favor.
Le consulto precios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>en clase <CLASE-BILLETE>. Un momento por favor.
Le consulto precios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>en tipo tren <TIPO-TREN>. Un momento por favor.
Le consulto precios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>llegando <FECHA-LLEGADA>. Un momento por favor.
Conjunto de respuestas definido para Respuesta-Precio

C.2.4.8. Confirmación,Respuesta-[Precio],Precio

Confirmación,Respuesta-[Precio],Precio
Le consulto precios. Un momento por favor.
Conjunto de respuestas definido para Respuesta-Precio

C.2.4.9. Respuesta-Tipo_tren

Respuesta vacía
No hay ningún tren que cumpla estos requerimientos. ¿Desea cambiar alguna de las características solicitadas?

Respuestas con cardinalidad 1
El único tren que realiza el trayecto es un <TIPO-TREN>. ¿Desea algo más?
El <NUMERO-RELATIVO>tren es un <TIPO-TREN>. ¿Desea algo más?
El tren más caro es un <TIPO-TREN>. ¿Desea algo más?
El tren más barato es un <TIPO-TREN>. ¿Desea algo más?
El tren más rápido es un <TIPO-TREN>. ¿Desea algo más?
El tren más lento es un <TIPO-TREN>. ¿Desea algo más?
Ese tren es un <TIPO-TREN>. ¿Desea algo más?

Respuestas con cardinalidad 2
Dispone de dos tipos de trenes. El tren más rápido es un <TIPO-TREN>y el otro es un <TIPO-TREN>. ¿Desea algo más?
Hay un único tipo de tren, que es un <TIPO-TREN>. ¿Desea algo más?
Dispone de dos tipos de trenes con salida <FECHA-SALIDA>. El tren más rápido es un <TIPO-TREN>y el otro es un <TIPO-TREN>. ¿Desea algo más?
Dispone de un único tipo de tren con salida <FECHA-SALIDA>, que es un <TIPO-TREN>. ¿Desea algo más?
Dispone de dos tipos de trenes con llegada <FECHA-LLEGADA>. El tren más rápido es un <TIPO-TREN>y el otro es un <TIPO-TREN>. ¿Desea algo más?
Dispone de un único tipo de tren con llegada <FECHA-LLEGADA>, que es un <TIPO-TREN>. ¿Desea algo más?
Los trenes anteriores son <TIPO-TREN>. ¿Desea algo más?

Respuestas con cardinalidad mayor a 2
Hay varios tipos de trenes. El tren más rápido es un <TIPO-TREN>y el más lento es un <TIPO-TREN>. ¿Desea algo más?
Hay un único tipo de tren, que es un <TIPO-TREN>. ¿Desea algo más?
Hay varios tipos de trenes con salida <FECHA-SALIDA>, El tren más rápido es un <TIPO-TREN>y el más lento es un <TIPO-TREN>. ¿Desea algo más?
Dispone de un único tipo de tren con salida <FECHA-SALIDA>, que es un <TIPO-TREN>. ¿Desea algo más?
Hay varios tipos de trenes con llegada <FECHA-LLEGADA>, El tren más rápido es un <TIPO-TREN>y el más lento es un <TIPO-TREN>. ¿Desea algo más?
Dispone de un único tipo de tren con llegada <FECHA-LLEGADA>, que es un <TIPO-TREN>. ¿Desea algo más?
Los trenes anteriores son <TIPO-TREN>. ¿Desea algo más?

C.2.4.10. Confirmación,Respuesta-[Tipo_tren],Tipo_tren

Confirmación,Respuesta-[Tipo_tren],Tipo_tren
Le consulto tipos de tren. Un momento por favor.
Conjunto de respuestas definido para Respuesta-Tipo_tren

C.2.4.11. Confirmación,Respuesta-Tipo_tren,Tipo_tren

Confirmación,Respuesta-[Tipo_tren],Tipo_tren
Le consulto tipos de tren. Un momento por favor.
Le consulto tipos de trenes para el trayecto <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>. Un momento por favor.
Conjunto de respuestas definido para Respuesta-Tipo_tren

C.2.4.12. Respuesta-Tiempo_recorrido

Respuesta vacía
No hay ningún tren que cumpla estos requerimientos. ¿Desea cambiar alguna de las características solicitadas?

Respuestas con cardinalidad 1 (I)
El único tren que realiza el trayecto tarda <DURACION>y es un <TIPO-TREN>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>con salida <HORA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>con llegada <HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>con salida <HORA-SALIDA>, llegada <HORA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>con salida <FECHA-SALIDA><HORASALIDA>y tarda <DURACION>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>con llegada <FECHA-LLEGADA><HORALLEGADA>y tarda <DURACION>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>con salida <FECHA-SALIDA>y tarda <DURACION>. ¿Desea algo más?
El único tren que realiza el trayecto es un <TIPO-TREN>con llegada <FECHA-LLEGADA>y tarda <DURACION>. ¿Desea algo más?

Respuestas con cardinalidad 2 (I)
Sólo hay dos trenes, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Sólo hay dos trenes con salida <INTERVALO-HORA-SALIDA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <INTERVALO-HORA-SALIDA>, los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Sólo hay dos trenes con llegada <INTERVALO-HORA-LLEGADA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con llegada <INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Sólo hay dos trenes con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>. El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Sólo hay dos trenes con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>. El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>.¿Desea algo más?

Respuestas con cardinalidad 2 (II)
Sólo hay dos trenes con salida <FECHA-SALIDA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <FECHA-SALIDA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Sólo hay dos trenes con llegada <FECHA-LLEGADA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con llegada <FECHA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Sólo hay dos trenes con salida <INTERVALO-HORA-SALIDA>y llegada <INTERVALO-HORA-LLEGADA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el otro es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <INTERVALO-HORA-SALIDA>y llegada <INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>.¿Desea algo más?

Respuestas con cardinalidad mayor a 2 (I)
Hay varios trenes. El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Hay varios trenes con salida <INTERVALO-HORA-SALIDA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <INTERVALO-HORA-SALIDA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Hay varios trenes con llegada <INTERVALO-HORA-LLEGADA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con llegada <INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Hay varios trenes con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <FECHA-SALIDA><INTERVALO-HORA-SALIDA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?

Respuestas con cardinalidad mayor a 2 (II)
Hay varios trenes con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>. El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con llegada <FECHA-LLEGADA><INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Hay varios trenes con salida <FECHA-SALIDA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <FECHA-SALIDA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Hay varios trenes con llegada <FECHA-LLEGADA>, El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con llegada <FECHA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?
Hay varios trenes con salida <INTERVALO-HORA-SALIDA>y llegada <INTERVALO-HORA-LLEGADA>. El tren más rápido es un <TIPO-TREN>que tarda <DURACION>y el más lento es un tipo <TIPO-TREN>que tarda <DURACION>. ¿Desea algo más?
Con salida <INTERVALO-HORA-SALIDA>y llegada <INTERVALO-HORA-LLEGADA>los trenes son <TIPO-TREN>y tardan <DURACION>. ¿Desea algo más?

C.2.4.13. Respuesta-Servicio

Respuesta vacía
No hay ningún tren que cumpla estos requerimientos. ¿Desea cambiar alguna de las características solicitadas?

Respuestas con cardinalidad 1
El único tren que realiza el trayecto es un <TIPO-TREN>y dispone de los siguientes servicios: <SERVICIOS>. ¿Desea algo más?
Se dispone de los siguientes servicios : <SERVICIOS>. ¿Desea algo más?

Respuestas con cardinalidad 2
Hay dos trenes. El tren más rápido es un <TIPO-TREN>con <SERVICIOS>y el otro es un <TIPO-TREN>con <SERVICIOS>. ¿Desea algo más?
Hay un único tipo de tren, que es un <TIPO-TREN>con <SERVICIOS>. ¿Desea algo más?
Se dispone de los siguientes servicios : <SERVICIOS>. ¿Desea algo más?

Respuestas con cardinalidad mayor a 2
Hay varios trenes. El tren más rápido es un <TIPO-TREN>con <SERVICIOS>y el más lento es un <TIPO-TREN>con <SERVICIOS>. ¿Desea algo más?
Hay un único tipo de tren, que es un <TIPO-TREN>con <SERVICIOS>. ¿Desea algo más?
Se dispone de los siguientes servicios : <SERVICIOS>. ¿Desea algo más?

C.2.4.14. Confirmación,Respuesta-[Servicio],Servicio

Confirmación,Respuesta-[Servicio],Servicio
Le consulto servicios. Un momento por favor.
Le consulto servicios de <CIUDAD-ORIGEN>a <CIUDAD-DESTINO>en un tren <TIPO-TREN>. Un momento por favor.
Conjunto de respuestas definido para Respuesta-Servicio

Respuesta-Clase_billete
La clase de billete es <CLASE-BILLETE>.
La información es viajando en clase <CLASE-BILLETE>.

Respuesta-[Día]
Los trenes son saliendo <FECHA-SALIDA>.
Los trenes son llegando <FECHA-LLEGADA>.

C.2.5. Otras

Nueva_consulta-[Nil]
No. ¿Desea algo más?
Sí. ¿Desea algo más?

Cierre-[Nil]
Gracias por utilizar el sistema de información de trenes. Adiós.

Anexo D

Desarrollo de sistemas de diálogo mediante VoiceXML

El objetivo de este anexo es presentar las ventajas de la tecnología de programación VoiceXML para el desarrollo de aplicaciones de reconocimiento de voz, considerándose el estándar de acceso vocal a la información en Internet.

A lo largo del mismo se exponen las características más representativas de este lenguaje de marcas, describiéndose los diferentes recursos y herramientas de autor disponibles. Asimismo, se muestra un ejemplo práctico de aplicación de VoiceXML para el desarrollo de una centralita telefónica en español.

D.1. Introducción a VoiceXML

El objetivo fundamental de VoiceXML es aprovechar las ventajas de las tecnologías web para el desarrollo de aplicaciones controlables mediante la voz, siendo un lenguaje de alto nivel con una gran cantidad de recursos disponibles.

Permite la integración de servicios de voz y de datos valiéndose del modelo cliente-servidor, posibilitando su uso conjuntamente con lenguajes de programación con uso tan extendido en la web como ASP o PHP. Un servicio de voz en VoiceXML se entiende como una secuencia de diálogos entre un usuario y la plataforma de implementación. Los diálogos provienen de servidores de documentos, externos a la plataforma de implementación.

Se definen dos tipos de diálogos: formularios y menús. Los formularios presentan información y entradas a completar por el locutor. Los Menús muestran selecciones sobre qué acciones debe realizar la aplicación a continuación.

D.1.1. Orígenes de VoiceXML

Los orígenes de VoiceXML se remontan al año 1995 con el Proyecto Phone Markup Language (PML) de AT&T, que tenía como objetivo el desarrollo de un lenguaje que simplificara la producción de aplicaciones de reconocimiento de voz.

Durante 1998 se desarrollan diversas variantes del lenguaje PML por AT&T y Lucent. Motorola desarrolla el lenguaje VoxML, IBM el SpeechML, HP el TalkML y PipeBeach el VoiceHTML. A finales de ese mismo año se acuerdan una serie de recomendaciones para el desarrollo de navegadores guiados mediante la voz. Con este fin, se constituye el VoiceXML

Forum por AT&T, IBM, Lucent y Motorola. El principal objetivo era desarrollar un estándar para aplicaciones de voz que estuviese basado en XML.

En el año 2000 el VoiceXML Forum establece la Versión 1.0 de VoiceXML (Boyer et al., 2000). En 2003 el W3C desarrolla la Versión 2.0 de VoiceXML, fruto del trabajo del consorcio de empresas miembros del grupo. Las últimas recomendaciones de VoiceXML 2.0 (McGlashan et al., 2004) fueron publicadas el 16 de marzo de 2004.

VoiceXML 2.1 provee un conjunto reducido de características adicionales. El principal objetivo de las especificaciones, cuyo último borrador de trabajo (*Working Draft*) fue publicado en septiembre de 2006, es facilitar a los programadores nuevos servicios de voz de forma completamente compatible con VoiceXML.

A finales de 2007 está previsto la publicación del primer borrador de trabajo de VoiceXML 3.0. Con esta nueva especificación se pretende proveer nuevas capacidades de diálogo para construir sistemas con interfaz oral avanzados, de forma que estas nuevas capacidades sean compatibles con otros lenguajes definidos por el W3C.

Otros estándares definidos por el W3C con relación con VoiceXML son SRGS (Speech Recognition Grammar Specification), SSML (Speech Synthesis Markup Language), PLS (Pronunciation Lexicon Specification), SISR (Semantic Interpretation for Speech Recognition), CCXML (Call Control XML) y SCXML (State Chart XML).

SRGS define el formato de las gramáticas a utilizar en el reconocimiento de la voz. SSML define un lenguaje de marcas para representar los mensajes del sistema mediante la combinación de voz pregrabada, voz sintética y música. PLS describe información fonética a utilizar en los procesos de reconocimiento y síntesis de la voz. SISR hace referencia a las anotaciones de las gramáticas basadas en reglas, en formato ECMAScript, para extraer la interpretación semántica del resultado del proceso de reconocimiento. CCXML es un lenguaje de marcas definido para realizar el control de la voz (procesado de la señal) y el control de los recursos telefónicos (multillamadas, llamadas en espera, transferencia de llamadas, etc.).

D.1.2. Estructura y funcionamiento

En VoiceXML, un servicio de voz es una secuencia de diálogos que son proporcionados por servidores de documentos, que acceden a bases de datos y generan diálogos. Un documento en VoiceXML especifica cada uno de los diálogos de la secuencia que debe conducir el Intérprete de VoiceXML. El Servidor de Documentos recoge las entradas de usuario y responde con un nuevo documento, de acuerdo con la arquitectura presentada en la Figura D.1.2.

Las ventajas más destacadas que introduce VoiceXML son:

1. Permite múltiples interacciones en un único documento.
 2. El autor no se ha de preocupar de la programación a bajo nivel ni de detalles de la plataforma.
 3. Separa los diferentes tipos de código.
 4. Permite portabilidad de servicios entre diferentes plataformas.
 5. Facilidad de uso en diferentes tipos de diálogos.
-

Las funcionalidades principales que incorpora son:

1. Salida de voz sintetizada.
2. Salida de ficheros de audio.
3. Reconocimiento de entradas de voz.
4. Reconocimiento de entradas DTMF.
5. Grabación de diálogos.
6. Control del flujo del diálogo.
7. Funciones de telefonía (llamadas, desconexión).

El funcionamiento básico consiste en recoger la entrada del usuario (voz o datos), asignar esta entrada a variables y tomar decisiones en base a estas variables, estableciéndose enlaces con documentos VoiceXML.

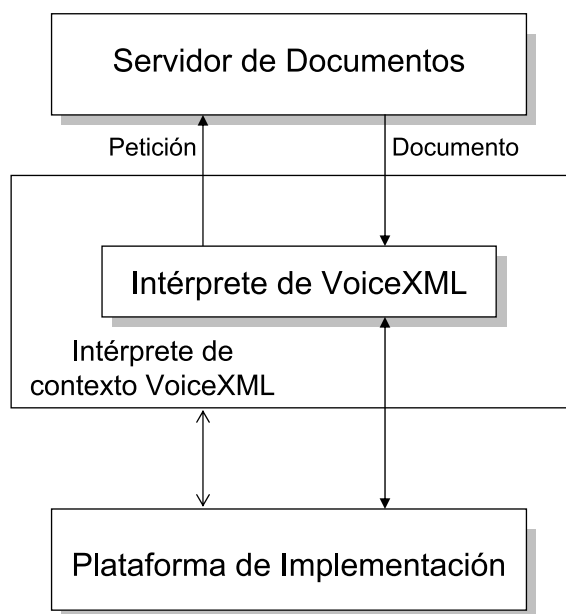


Figura D.1: Arquitectura de VoiceXML

Los principios de diseño de VoiceXML son:

1. Estandarización (portabilidad).
2. Permitir diversos tipos de plataforma gracias a los diferentes formatos de ficheros soportados.
3. Reconocimiento semántico a través de gramáticas.
4. Proveer diferentes opciones al programador.

Por último, los requerimientos de la plataforma se citan a continuación:

1. Protocolo URI para enlazar los diferentes documentos.
2. Formatos de audio soportados.
3. Detección DTMF.
4. Reconocimiento dinámico de las gramáticas de voz (SRGS, ABNF, JSGF).
5. Grabación de audio.

D.1.3. Conceptos básicos

Se puede entender un documento VXML como una máquina de estados que va recorriendo el usuario durante el diálogo. Cada diálogo determina su sucesor, utilizándose URI para enlazar los diversos documentos. La ejecución finaliza cuando un diálogo no tiene transición o cuando así se especifica.

D.1.3.1. Diálogos y subdiálogos

Existen dos tipos de diálogos:

1. **FORMULARIOS:** Definen una interacción usuario-máquina que recoge los valores de una serie de campos a almacenar en variables. Cada campo especifica una gramática que define si la entrada proporcionada es válida o no. Se puede utilizar una gramática para todo el formulario, rellenando todos los campos en la misma iteración.
2. **MENÚS:** Conjunto de opciones y transiciones a otros diálogos basados en la elección.

Los subdiálogos son llamadas a funciones, invocando una nueva iteración y retornando al formulario original. Se trata de descomponer diálogos complejos en partes independientes para estructurarlos más adecuadamente, de forma que puedan ser aprovechados en otros contextos similares.

Otros conceptos importantes son:

1. **Sesiones:** Desde que el usuario comienza a interactuar con el VoiceXML IC hasta que finaliza el diálogo.
 2. **Aplicaciones:** Una aplicación es un conjunto de documentos que comparten el mismo documento raíz (ROOT). El ROOT se carga cuando el usuario interactúa con cualquier documento de la aplicación y está disponible hasta que el usuario cambie de aplicación.
 3. **Gramáticas:** Cada diálogo posee una o varias gramáticas de voz o de DTMF asociadas. Las gramáticas pueden estar activas aunque los diálogos asociados no lo estén.
 4. **Eventos:** Se definen mecanismos para tratar entradas de usuario y eventos no incluidos en el formulario (que el usuario no responda, falta de inteligibilidad, solicitud de ayuda, etc.). Cada evento define su función de tratamiento.
 5. **Enlaces:** Permiten especificar gramáticas, transferir el control a otra URI...
-

D.2. Constructores de diálogo

D.2.1. Formularios

Es el componente fundamental de los documentos VXML. Contienen:

1. Campos de entrada (items) y de control.
2. Declaración de variables.
3. Tratamiento de eventos.
4. Acciones a ejecutar cuando se completen determinados campos.

Para realizar la interpretación de los formularios se utiliza el algoritmo FIA, que recorre el formulario almacenando el contenido de los campos cuando la condición de guarda se cumple, reproduce los prompts asociados al formulario y comprueba si la condición *<filled>* (acciones a ejecutar cuando un determinado campo se completa) del Form se cumple. Se definen dos tipos de formularios siguiendo el criterio del modo de interpretación:

1. **Formularios directos:** La estrategia se basa en ir recorriendo los campos de forma secuencial, eludiendo aquellos que se van completando adecuadamente.
2. **Iniciativa mixta:** Definiendo *form-level grammars* (gramáticas definidas externamente al formulario), las entradas se pueden completar en cualquier orden y se pueden rellenar varios campos en una iteración. Además, pueden haber varias gramáticas correspondientes a formularios diferentes activas al mismo tiempo, determinando el sistema a qué campo del formulario se ha dado respuesta.

En un formulario pueden aparecer dos tipos de campos:

1. **Items de entrada:** Incorporan prompts para informar al usuario sobre qué debe hacer, gramáticas que determinan si la entrada es válida o no y tratamiento de eventos. Se añade la acción *<filled>* para incorporar la acción a realizar cuando una variable de entrada se completa adecuadamente.
2. **Items de control:** Se trata de las etiquetas *<block>* (presenta información y no pide entrada de datos) e *<initial>* (bloque inicial del formulario para presentar información o ayuda sobre el mismo).

Cada campo del formulario tiene asociada una variable, que al final contendrá el resultado de interpretar el formulario y que se puede referenciar a lo largo de la aplicación.

D.2.2. Menús

Se trata del caso más simple de formulario, incluyendo un único campo anónimo para forzar al usuario a realizar una elección y realizar una transición en función de ésta. Puede tener sus propias gramáticas asociadas.

Las opciones se especifican mediante la etiqueta `<choice>`, donde se indica una gramática de voz y/o de DTMF y un enlace a acudir cuando se selecciona la alternativa. El atributo `accept` se utiliza para determinar si el usuario debe indicar la opción completa (*exact*) o sirven subfrases de la misma (*approximate*).

Respecto al modelo de interpretación, el funcionamiento es idéntico al de un formulario con un único campo que se encarga de realizar todo el trabajo, estableciéndose las indicaciones correspondientes al enlace mostrado en el objeto `<choice>`.

D.3. Modelado del lenguaje en VoiceXML

El mecanismo elegido por VoiceXML para introducir un modelo de lenguaje se basa en la utilización de gramáticas. Una gramática determina la secuencia de palabras aceptables durante el proceso de reconocimiento automático del habla, y asimismo, influye en el proceso de comprensión de las frases, pues indica la interpretación semántica de unidades sintácticas de las mismas.

En mayo de 2004 el W3C editó una serie de recomendaciones sobre el formato de gramáticas para el reconocimiento de voz (Hunt y McGlashan, 2004). Este documento puede verse como un complemento a las especificaciones de VoiceXML 2.0, tratando los casos de entradas de voz y entradas en formato DTMF.

En el SRGS se proponen dos formatos fundamentales de gramáticas:

1. ABNF (Augmented BNF): Se trata de un formato en texto plano basada en formatos anteriores como el BNF y JSFG (*Java Speech Grammar Format*).
2. XML: Se utiliza el formato XML, delimitando los diferentes campos entre marcas.

Ambos formatos son independientes del contexto y son semánticamente equivalentes, desde el punto de vista que ((Hunt y McGlashan, 2004), Apéndice F):

1. Aceptan el mismo lenguaje como entrada.
2. Procesan cadenas de entrada de forma idéntica.

En VoiceXML también se contemplan modelos de n-gramas, introducción de medidas de confianza y métodos de aplicación de suavizado como Back-Off.

D.3.1. Gramáticas

D.3.1.1. Gramáticas para el reconocimiento de voz

Se indican en el elemento `<grammar>`. Sus funciones son:

1. Especificar una serie de pronunciaciones que el usuario debe mencionar para ejecutar una acción o proporcionar información.
-

2. Para una pronunciación concordante, devuelve la interpretación semántica correspondiente.

La etiqueta está definida para permitir cualquier formato de gramática que cumpla los dos requerimientos anteriores (SRGS, ABNF). Las plataformas de VoiceXML deben ser conformes a los requerimientos de la SRGS.

Las gramáticas pueden ser introducidas de forma interna o externa. Las gramáticas internas se incluyen directamente en la etiqueta `<grammar>`. Las gramáticas externas se incluyen a través de URIs.

El peso de las gramáticas se indica mediante el atributo *weight* siguiendo el formato indicado en el SRGS, estableciéndose de este modo una jerarquía entre las diferentes gramáticas que conforman un formulario.

D.3.1.2. **Ámbito de las gramáticas**

Las gramáticas internas a los campos de entrada sólo están activas cuando se ejecuta el campo correspondiente. Las gramáticas por enlaces poseen el alcance del elemento que engloba el link. Las gramáticas asociadas a formularios poseen el alcance del formulario, del documento o de la aplicación completa (caso de estar englobadas en el documento ROOT). Las gramáticas asociadas a menús poseen como alcance por defecto el del diálogo correspondiente, pero pueden darse los mismos casos que para las gramáticas del formulario.

Un formulario puede tener activas varias gramáticas de forma simultánea y desactivar otro conjunto.

D.3.1.3. **Activación de las gramáticas**

Cuando el interprete permanece a la espera de una entrada tras visitar un campo, las gramáticas activas son:

1. Gramáticas asociadas a dicho campo, inclusive la de los campos link.
2. Gramáticas del formulario.
3. Gramáticas contenidas en links dentro del documento.
4. Gramáticas contenidas en el documento ROOT (links, menús, formularios).
5. Gramáticas definidas en la captura de eventos de la plataforma (ayuda, salida, cancelar).

El orden de precedencia es el mostrado. Si existe más de una gramática activa del mismo nivel, la precedencia la marca el orden de los documentos. Como mínimo debe haber una gramática activa cuando se espera la respuesta de usuario. Se pueden desactivar gramáticas, de forma que simplemente éste vigente la del formulario en cuestión.

La interpretación semántica viene descrita en el SRGS y SISR (Tichelen, 2003). El resultado de la interpretación semántica debe pasarse a variables ECMAScript, funcionalidad que debe ser soportada por el reconocedor. Cada campo de entrada tiene asociado

un *slot name*, que almacena el resultado del reconocimiento en vistas a determinar si la entrada suministrada concuerda o no con la gramática.

En caso de gramáticas asociadas a formularios, el resultado de reconocer adecuadamente uno o varios campos supone la asignación de las variables ECMAScript y slots correspondientes.

Las gramáticas asociadas a campos concretos sólo se activan cuando el algoritmo visita dicho campo.

D.4. Recursos disponibles

A la hora de desarrollar aplicaciones en VoiceXML y verificar posteriormente su correcto funcionamiento se presentan dos alternativas:

1. Utilizar una aplicación de escritorio que permita crear los diferentes componentes de la aplicación y probar su funcionamiento sin necesidad de utilizar una conexión a red.
2. Crear los ficheros que compondrán la aplicación a través de editores y posteriormente alojar la aplicación en un servidor de VoiceXML externo. Esta alternativa conlleva la necesidad de obtener una cuenta de usuario para realizar el hospedaje de archivos en el servidor y la utilización de números de teléfono internacionales cada vez que se desee acceder a la aplicación.

Las diferentes empresas que conforman el VoiceXML Forum han optado mayoritariamente por la segunda opción (www.xml-journal.com/voicexml), existiendo actualmente un gran número de servidores gratuitos donde poder hospedar las aplicaciones. A continuación se resumen las principales características de las aplicaciones de RAH desarrolladas por los servidores más relevantes:

1. **Be Vocal Café:** El reconocedor de voz de Be Vocal (www.bevocal.com) utiliza dos mecanismos fundamentales para aumentar la precisión en el reconocimiento. Por un lado, devuelve los N mejores resultados, por otro, es capaz de retornar múltiples interpretaciones para los casos en los que el resultado esté presente en varias gramáticas. El reconocedor es dependiente del usuario.
 2. **Loquendo Café:** Loquendo ASR (www.loquendocafe.com) es independiente del locutor y es capaz de reconocer frases con gran número de palabras, incluso en ambientes ruidosos. El vocabulario es abierto y está optimizado para la línea telefónica. La precisión en el reconocimiento se basa en la utilización de modelos acústicos entrenados con grandes corpus, la integración de modelos de Markov y redes neuronales y la selección de fonemas especializada para cada idioma. Además se incorporan modelos acústicos mixtos para tratar la variabilidad regional en los acentos. En este campo, existe disponible un ASR para español-castellano y español-catalán.
 3. **Nuance Voice Platform:** Nuance (www.nuance.com/prodserv/nvp.html) dispone de un RAH independiente del usuario con un gran vocabulario, capaz de funcionar en ambientes ruidosos. Además, el reconocedor posee la funcionalidad de adaptarse
-

a la experiencia del locutor, la detección automática del idioma del locutor y herramientas de entrenamiento basadas en las características de voz locales.

En cuanto a aplicaciones de escritorio, la más relevante por lo que respecta al número de funcionalidades de VoiceXML soportadas, es el IBM Voice Toolkit.

D.4.1. IBM Voice Toolkit para WebSphere Studio

Voice Toolkit (www.ibm.com/software/speech/enterprise/vtoolkit.html) ofrece las siguientes funcionalidades:

1. Creación de proyectos de voz.
2. Desarrollo de ficheros en VoiceXML.
3. Editar archivos y diálogos en VoiceXML mediante el editor de VoiceXML.
4. Creación de gramáticas en diferentes formatos.
5. Añadir entradas a los diccionarios y crear diccionarios propios.
6. Creación de pronunciaciones para vocabulario específico mediante el constructor de pronunciaciones.
7. Grabación de mensajes de audio y configuración para su uso en aplicaciones de VoiceXML.
8. Disposición de componentes de diálogo reutilizables (plantillas de comercio electrónico, validación de información...) (www.ibm.com/websphere/appserver)
9. Realizar llamadas a aplicaciones web mediante la utilización de servlets y JavaBeans, permitiendo pasar datos desde aplicaciones en VoiceXML a aplicaciones web.
10. Obtener resultados de aplicaciones web.
11. Comprobar el funcionamiento práctico de las aplicaciones.

Los requerimientos de hardware son: Mínimo procesador Intel 500 MHz con 768 MB de RAM. Mínimo 1GB de disco duro disponible. Existe un listado de micrófonos y tarjetas de sonido recomendadas.

D.4.1.1. Instalación del Voice Toolkit

Para realizar la instalación del toolkit hay que seguir una serie de pasos en un orden establecido. Si surgiese algún problema durante la instalación, la experiencia hace recomendar la desinstalación de los módulos anteriores y el retorno al primer paso de la instalación. Debido a que los paquetes más representativos de la herramienta (reconocedor y conversor TTS) están en construcción para LINUX, a continuación se mencionan los pasos a seguir para realizar la instalación bajo Windows 2000 o XP:

PASOS PREVIOS:

1. Asegurarse de tener instalado el Service Pack 2 de Microsoft.
2. Instalar la máquina virtual de Java en caso de no tenerla disponible. Es necesario el paquete J2RE (Java 2 Runtime Environment). La versión recomendada es la 1.3.1.

SOFTWARE REQUERIDO:

1. El primer paso que realiza el instalador del toolkit consiste en verificar si está instalado el software IBM WebSphere Studio Site Developer 5.0.1. o el WebSphere Studio Application Developer 5.0.1. Se recomienda la instalación de la segunda de las aplicaciones, al requerir de un número de recursos menor, lo que facilita la instalación.
2. Instalar IBM Server SDK 1.5. (www.ibm.com/software/speech/enterprise). Esta herramienta permite testar las aplicaciones desarrolladas a modo de *voice browser*. Para instalar esta aplicación es necesario descargarse al menos uno de los lenguajes disponibles. Para español están disponibles los paquetes “español alfabetización tradicional” y “español Sudamérica”. Para cada idioma hay disponibles dos paquetes, uno con la versión TTS y otro con la versión CTTS (Concatenative TTS). CTTS ofrece una mayor calidad en la conversión de texto a voz a fuerza de consumir un mayor número de recursos.

INSTALACIÓN DE LA APLICACIÓN:

El primer paso que realiza el asistente consiste en comprobar si el software requerido está presente y configurado correctamente, notificando de igual forma si existe software que interfiere en el funcionamiento correcto de la plataforma y que debe desinstalarse.

Una vez instalada la aplicación cabe realizar la configuración de la misma indicando las rutas en las que se han instalado los diferentes componentes necesarios.

D.4.1.2. Reconocimiento de voz

IBM Voice Toolkit incorpora el reconocedor IBM Via Voice, que es dependiente del usuario y se vale de modelos acústicos adaptados al canal telefónico. Estos modelos son propietarios de IBM. El funcionamiento del reconocedor consiste en apoyarse en las gramáticas activas, procesando la señal de audio de entrada y comparando los formatos de sonido con los formatos básicos de sonidos hablados, e intentando recuperar la combinación más probable que representa la entrada de audio. Por último, el reconocedor compara los sonidos con la lista de palabras y frases de las gramáticas activas.

Via Voice se vale de diccionarios que contiene las pronunciaciones de un gran número de palabras. Es posible incorporar a los diccionarios disponibles todas aquellas palabras que se desee que sean aceptadas por el reconocedor y que no estén contempladas por defecto. Para ello, es necesario indicar su modelo de fonemas según la IPA (International Phonetic Association) (www.arts.gla.ac.uk/IPA/ipa.html). Hay disponible un constructor de pronunciaciones para llevar a cabo esta funcionalidad (Figura D.2).



Figura D.2: Constructor de pronunciaciones de Voice Toolkit

D.5. Aplicación práctica: Desarrollo de una centralita en castellano

En este apartado se expone una aplicación práctica de los contenidos teóricos mostrados en los apartados anteriores. Como ejemplo de aplicación, se ha tomado como tarea una centralita telefónica. Las siguientes tablas muestran las gramáticas utilizadas en formato JSGF. En primer lugar, aparece la gramática fundamental de la centralita y seguidamente una gramática para la confirmación de los datos obtenidos del reconocedor.

```
#JSGF V1.0;
grammar centralita;
public <centralita> = por favor <peticion> | <peticion> [por favor];
<nombremasc> = pepe | juan | josé | luis | victor | andrés | guillermo | paulo | antonio | pablo | raul | germán |
david | miguel | carlos | mario | felipe | manuel | manolo | mariano | nacho | ignacio | jorge;
<nombrefem> = maría | ana | luisa | isabel | paz | antonia | lidón | lledó;
<nombre> = (<nombremasc> | <nombrefem>);
<apellido> = martínez | lópez | jiménez | marzal | vilá | peris | castellanos | aibar | prat | castaño | valls | amengüal
| montoliu | sanz | gómez | aliaga | fabregat | porcar | varó | pelayo | toledo | lobo | climent | ventura | garcía | ibáñez
| palomar | llorens | vilar | zarco | yagüe | llopis | espósito | badenas | monfort | granada | vizcaíno | iborra;
<persona> = ([el señor] (<nombremasc> [<apellido>] | [<nombremasc>] <apellido>)) | ([la señora]
(<nombrefem> [<apellido>] | [<nombrefem>] <apellido>));
<digito> = cero | uno | dos | tres | cuatro | cinco | seis | siete | ocho | nueve;
<numero> = <digito> [<digito> [<digito> [<digito>]];
<extension> = ( el número | la extensión | el | el teléfono ) <numero>;
<pedirhablar> = (deseo | desearía | me gustaría | quisiera | quiero | quería) hablar;
<pedirpasar> = póngame | me pone | me pasa | pásame | puede ponerme | puede pasarme | me pasaría;
<pedir> = <pedirhablar> | <pedirpasar>;
<pedirextension> = <pedir> con <extension>;
<pedirpersona> = <pedir> con <persona> | se puede poner <persona> | <pedir> con el secretario de <persona>
| <pedir> con la secretaria de <persona> | <pedir> con el jefe de <persona> | <pedir> con la jefa de <persona>
| <pedir> con el despacho de <persona> | <pedirpasar> con la extensión de <persona>;
<peticion> = <pedirextension> | <pedirpersona>;
```

```
#JSGF V1.0;
grammar confirma;
public <confirma> = <si> | <no>;
<si>= si | vale | es correcto | bien | correcto;
<no>= no | mal | está mal | incorrecto;
```

La siguiente tabla muestra el código VXML de la centralita. Para su programación se ha insertado un único formulario. En primer lugar, se han definido tres variables para personalizar un saludo según el intervalo horario en el que se produzca la llamada. Seguidamente se introducen dos campos de entrada, uno para que el usuario indique la persona o extensión con la que desea hablar siguiendo el modelo de lenguaje definido en la gramática, y un segundo campo para que el usuario confirme si la salida del reconocedor es correcta.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<vxml version="1.0" xml:lang="es-ES" xmlns="www.w3.org/2001/vxml">
<meta name="GENERATOR" content="Voice Toolkit for WebSphere Studio"/>
<form>
<var name="horas"/>
<var name="minutos"/>
<var name="saludo"/>
<block>
<script>
var d = new Date();
horas = d.getHours();
minutos = d.getMinutes();
saludo="Buenos días.";
if ('horas /> 14') saludo="Buenas tardes.";
if ('horas />20') saludo="Buenas noches.";
if ('horas /<7') saludo="Buenas noches.";
</script>
</block>
<block> <value expr="saludo"/> Son las <value expr="horas"/> horas, <value expr="minutos"/> minutos
</block>
<block>Bienvenido al servicio de centralita. ¿En qué puedo ayudarle?</block>
<field name="frase">
<grammar src="names.jsgf"></grammar>
<filled>
<if cond="frase == 'salir'">
<prompt>Gracias por utilizar la centralita de nuestra empresa. Adios.</prompt>
<exit></exit>
</if>
<if cond="frase == 'ayuda'">
<prompt>Por favor, indíqueme con que persona o extensión desea hablar</prompt>
</if>
</filled>
</field>
<field name="confirma">
<prompt>He entendido: <value expr="frase"/> ¿es correcto?</prompt>
<grammar src="confirma.jsgf"></grammar>
<filled>
<if cond="confirma == 'correcto'">
<audio src="no me llames dolores llámame lola.wav"></audio>
</if>
<if cond="confirma == 'no'">
<prompt>Por favor, indíqueme con que persona o extensión desea hablar</prompt>
<clear namelist="frase"></clear>
<clear namelist="confirma"></clear>
</if>
</filled>
</field>
</form>
</vxml>
```

Si se produce una confirmación positiva, se reproduce un fichero de audio antes de traspasar el control del diálogo al siguiente estado. En cualquier instante, el usuario puede

solicitar ayuda o abandonar la aplicación.

Para evaluar el funcionamiento de esta aplicación con el IBM SDK en modo audio cabe ejecutar el siguiente comando:

```
%IBMVS %/bin/vs_audio_es_ES centralita.vxml
```

Tras realizar la primera ejecución de la aplicación, SDK detectó 22 términos con dificultad de entendimiento. Estas palabras coincidían con gran parte de los nombres y apellidos definidos en la gramática, y fueron incorporados a los diccionarios del reconocedor a través de su codificación de fonemas extraída de la tabla de la IPA.

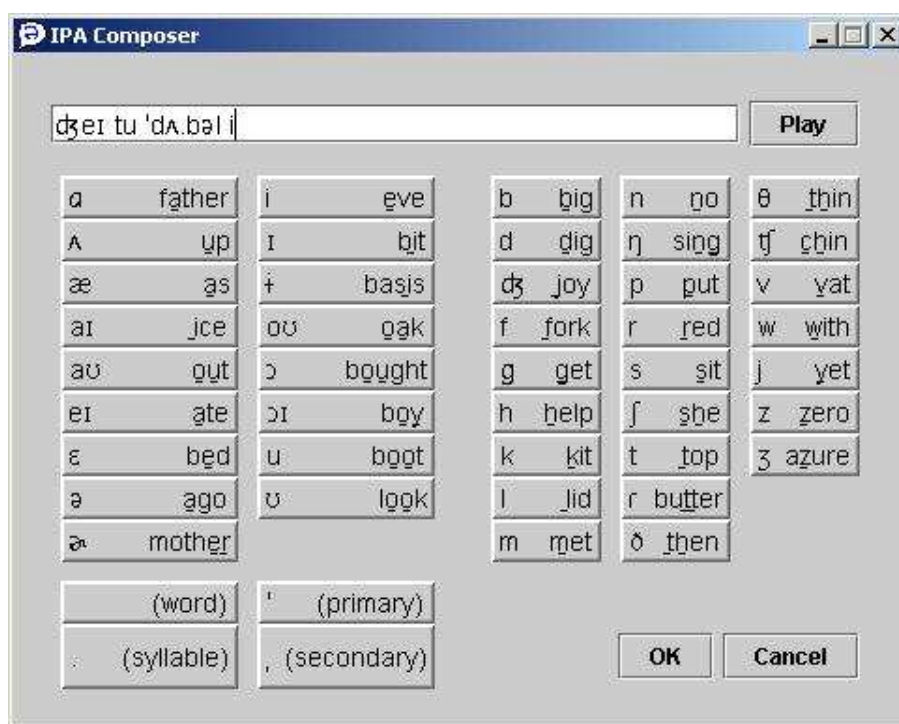


Figura D.3: Tabla de fonemas definida por la IPA

D.6. Implementaciones de las especificaciones de interfaces vocales del W3C

D.6.1. VoiceXML 2.0 y 2.1

- BeVocal Café (cafe.bevocal.com) es un entorno de desarrollo basado en la web que incluye un intérprete de VoiceXML 2.0 (y 1.0) y las herramientas necesarias para depurar y testar aplicaciones. El intérprete desarrollado soporta las últimas recomendaciones definidas por el W3C, incluyendo verificación del usuario o la utilización de datos en formato XML.

- Conita (www.conita.com) ha implementado un interprete VoiceXML basado en Open VXI.
 - Eloquant (www.eloquant.com) ha implementado un intérprete de VoiceXML 2.0. Soporta las especificaciones de interpretación semántica definidos por el W3C para el reconocimiento de la voz. Eloquant ofrece hospedaje para aplicaciones vocales.
 - HeyAnita (www.heyanita.com) soporta la mayoría de los estándares definidos por el VoiceXML Forum, incluyendo VoiceXML 2.0. Ofrece el servidor FreeSpeech para el desarrollo y hospedaje de aplicaciones vocales. Además facilita la utilización de reconocedores, sintetizadores y dispositivos de telefonía.
 - HP (www.hp.com) ofrece la plataforma OpenCall Media, que soporta VoiceXML bajo entornos HP-Unix y Linux con posibilidad de uso de los protocolos de telefonía ISUP/SIP/RDSI, software de reconocimiento y síntesis MRCP (por ejemplo, Nuance y SpeechWorks) y CCXML 1.0. Además, se ha desarrollado una utilidad de escritorio VoiceXML SDK para el desarrollo de aplicaciones y el entorno OpenCall Speech Web, para el desarrollo y hospedaje de portales de voz.
 - IBM (www.ibm.com) dispone de un gran número de herramientas para el desarrollo, depuración y prueba de aplicaciones VoiceXML. El software de aplicaciones VoiceXML de IBM incorporan la mayoría de las especificaciones VBWG, incluyendo VoiceXML 2.0 y SRGS 1.0.
 - El servidor Omvia Media desarrollado por Intervoice (www.intervoice.com) incluye un navegador VoiceXML 2.0. Soporta múltiples reconocedores y sintetizadores en una misma aplicación. Además, ha desarrollado InVision Studio, una herramienta para el diseño, desarrollo y evaluación de aplicaciones VoiceXML 2.0.
 - JVoiceXML (jvoicexml.sourceforge.net) es un intérprete de VoiceXML *open source* desarrollado en JAVA, soportando APIs de JAVA como JSAPI y JTAPI. Se trata de una plataforma independiente de la aplicación que puede utilizarse de forma gratuita.
 - Loquendo (www.loquendo.com) ha desarrollado un intérprete que soporta documentos VoiceXML 1.0 y 2.0. Está integrado dentro de la Plataforma Loquendo e incluye una gran variedad de servicios de voz. Además, LoquendoCafé facilita recursos y herramientas para aprender a crear aplicaciones vocales, así como tener la posibilidad de hospedar las aplicaciones desarrolladas y probar su funcionamiento (vía telefónica) en una gran variedad de idiomas.
 - Lucent (www.lucent.com) ha desarrollado la pasarela MiLife VoiceXML, que provee acceso telefónico a servicios de voz en entornos web.
 - Motorola (www.motorola.com) ha desarrollado el sistema VoxGateway (VoiceXML 2.0), integrado en un gran número de plataformas y portales de voz.
 - Nuance (www.nuance.com) ofrece una plataforma VoiceXML, herramientas gráficas para el desarrollo de aplicaciones, un navegador vocal y una plataforma para el desarrollo y prueba de prototipos.
-

- OpenVXI (fife.speech.cs.cmu.edu/openvxi) es un intérprete de VoiceXML *open source* desarrollado por la CMU y SpeechWorks. Cumple la mayoría de las especificaciones definidas para VoiceXML 2.0.
- OptimSys (www.optimsys.cz) ofrece una plataforma VoiceXML escalable, modular y flexible denominada OptimTalk. Incluye un intérprete VoiceXML y CCXML y un software SDK. OptimTalk permite la elección de las tecnologías de reconocimiento, síntesis de voz y telefonía que desee el desarrollador.
- PublicVoiceXML (www.publicvoicexml.org) es un navegador *open source* que implementa el estándar VoiceXML 2.0. Está diseñado para simplificar las características del hardware telefónico necesario, soportando funcionalidades de telefonía móvil.
- SpeechWorks (www.speechworks.com) ha desarrollado un toolkit para desarrolladores de aplicaciones VoiceXML 2.0. El navegador OpenSpeech, disponible para los sistemas operativos Linux y Windows, incluye entre otras funcionalidades un intérprete VoiceXML.
- La plataforma Tellme (www.tellme.com) responde actualmente a más de un millón de llamadas diarias. Tellme Studio permite la creación de aplicaciones VoiceXML de forma gratuita.
- Vocalocity (www.vocalocity.net) ha implementado una plataforma que cumple las últimas especificaciones de VoiceXML 2.0. Soporta la posibilidad de más de un teléfono, reconocedores y sintetizadores.
- VoiceGenie (www.voicegenie.com) ha desarrollado una plataforma que soporta 4 tipos de reconocedores (SpeechWorks, BBN, AT&T Watson y Nuance) y 7 sintetizadores (incluyendo SpeechWorks, AT&T Natural Voices, Rhetorical y SVOX). Dispone de un portal web para el desarrollo y prueba gratuita de aplicaciones. Además, ha desarrollado la plataforma VoiceGenieCCXML, compatible con las especificaciones CCXML del W3C y capaz de interactuar con la plataforma VoiceXML VoiceGenie y las desarrolladas por otras empresas.
- Voxeo (www.voxeo.com) ha desarrollado la plataforma VoiceCenter, que cumple por completo las especificaciones VoiceXML 2.0 y 2.1.
- VoxPilot (www.voxpilot.com) ofrece un entorno web para el desarrollo y distribución de aplicaciones multilingües VoiceXML 2.0.

D.6.2. Especificaciones para las gramáticas de reconocimiento de VOZ

- Las especificaciones de SRGS 1.0 incluyen informes de BeVocal, IBM (se incorporan las especificaciones VBWG, incluyendo SRGS 1.0), Loquendo (Loquendo Speech Server), Lucent, Microsoft (Microsoft Speech Server, Microsoft Speech Application SDK y las utilidades Microsoft SALT para Internet Explorer) y SpeechWorks (ScanSoft).

- La plataforma OptimTalk de la empresa OptimSys proporciona una implementación de las especificaciones SRGS, capaz de interactuar con los reconocedores compatibles con Microsoft API 5.1.
- Grammar Studio, de Voice Web Solutions (www.voicewebsolutions.net), es un editor GRXML de escritorio compatible con las especificaciones SRGF del W3C para VoiceXML y aplicaciones multimodales SALT, telefonía y aplicaciones multimodales X+V que utilicen gramáticas SRGS.
- La plataforma VoiceCenter de Voxeo soporta por completo las recomendaciones SRGS 1.0 del W3C.
- VoxPilot incluye en su plataforma VoiceXML 2.0 una implementación de SRGS.

D.6.3. Interpretación semántica de gramáticas de reconocimiento de voz

- El reconocedor Loquendo soporta las recomendaciones de interpretación semántica SISR 1.0 (*Semantic Interpretation for Speech Recognition*) utilizando gramáticas SRGS 1.0.
- Microsoft Speech Server implementa representación semántica de gramáticas SRGS basándose también en las especificaciones SISR.
- La plataforma OptimTalk de OptimSys contiene un intérprete semántico que cumple SISR. Puede utilizarse con la implementación de SRGS realizada por OptimSys o para la interpretación semántica de otros tipos de gramáticas.

D.6.4. CCXML (Call Control XML)

- La plataforma OptimSys incluye una implementación modular y escalable de las especificaciones CCXML. El intérprete CXML es independiente de los dispositivos de telefonía y protocolos asociados, lo que permite una integración sencilla con la infraestructura telefónica que se esté utilizando mediante OptimTalk SDK.
 - Phonologies (www.phonologies.com) ha desarrollado Oktopous PIK, una implementación *open source* de las especificaciones CCXML del W3C programada en C++. Esta implementación es independiente también de los dispositivos y protocolos de telefonía utilizados.
 - La plataforma VoiceCenter de Voxeo soporta por completo las especificaciones definidas por el W3C para CCXML.
-

Bibliografía

Aist, G., J. Dowding, B.A. Hockey, M. Rayner, J. Hieronymus, D. Bohus, B. Boven, N. Blaylock, E. Campana, S. Early, G. Gorrell, y S. Phan. 2003. Talking through procedures: An intelligent Space Station procedure assistant. En *Proc. of EACL-2003*, páginas 187–190, Budapest (Hungria).

Alcácer, N., J.M. Benedí, F. Blat, F. García, R. Granell, L.F. Hurtado, C. Martínez, y F. Torres. 2004a. Normativa de etiquetado de diálogo para DIHANA. Informe técnico, Proyecto DIHANA TIC2002-04103-C03. DSIC. Universitat Politècnica de València, Valencia (España).

Alcácer, N., R. Granell, S. Grau, y D. Griol. 2004b. Adquisición de un corpus en el proyecto DIHANA. En *Curso de Tecnologías Lingüísticas. Fundación Duques de Soria*, Soria (España).

Alcácer, Nieves, María J. Castro, Isabel Galiano, Ramón Granell, Sergio Grau, y David Griol. 2004c. Diseño de un corpus de diálogo: DIHANA. En *Actas de las III Jornadas en Tecnología del Habla*, páginas 131–135, Valencia (España).

Alexandersson, J. y T. Becker. 2001. Overlay as the basic operation for discourse processing in a multimodal dialogue system. En *Proc. of 2nd IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, páginas 376–383, Seattle (Estados Unidos).

Allen, J., D. Byron, M. Dzikovska, G. Ferguson, L. Galescu, y A. Stent. 2001. Towards conversational human-computer interaction. En *AI Magazine, 2001*, páginas 27–37.

Allen, J. y M. Core. 1997. Draft of DAMSL: Dialog Act Markup in Several Layers. Informe técnico, Dept. of Computer Science, Univ. of Rochester, Rochester, NY (Estados Unidos).

Allen, James, George Ferguson, Bradford W. Miller, Eric K. Ringger, y Teresa Sikorski Zollo. 2000. Dialogue systems: From theory to practice in TRAINS-96. En *Handbook of Natural Language Processing*, páginas 347–376.

Allen, J.F. y C.R. Perault. 1980. Analyzing intentions in dialogues. En *Artificial Intelligence*, volumen 15(3), páginas 143–178.

Álvarez, J., D. Tapias, C. Crespo, I. Cortázar, y F. Martínez. 1997. Development and evaluation of the ATOS spontaneous speech conversational system. En *Proc. of*

- International Conference on Acoustics, Speech, and Signal Processing (ICASSP'97)*, páginas 1139–1142.
- Andeani, G., D. Di Fabbrizio, M. Gilbert, D. Gillick, D. Hakkani-Tur, y O. Lemon. 2006. Let's DISCOH: Collecting an Annotated Open Corpus with Dialogue Acts and Reward Signals for Natural Language Helpdesks. En *Proc. of IEEE 2006 Workshop on Spoken Language Technology (SLT)*, páginas 218–221, Palm Beach (Aruba).
- Appelt, D.E. 1985. Planning English Sentences. En *Cambridge, University Press*.
- ARPA. 1994. Speech and Natural Language Workshop. En *Book of Proceedings*, San Mateo (Estados Unidos).
- Aubert, Xavier y Hermann Ney. 1995. Large Vocabulary Continuous Speech Recognition Using Word Graphs. En *Proc. of International Conference on Acoustics, Speech, and Signal Processing (ICASSP'95)*, páginas 49–52, Detroit (Estados Unidos).
- Aust, H., M. Oerder, F. Seide, y V. Steinbiss. 1995. The Philips automatic train timetable information system. En *Speech Communication*, 17, páginas 249–262.
- Aust, H., M. Oerder, M. Seide, y V. Steinbiss. 1994. The Philips automatic train timetable information system. En *Proc. of the Interactive Voice Technology for Telecommunications Applications (IVTTA'94)*, páginas 67–72, Kyoto (Japón).
- Baggia, P., G. Castagneri, y M. Danieli. 2000. Field trials of the Italian ARISE train timetable system. En *Speech Communication*, volumen 31, páginas 355–367.
- Bahl, Lalit R., Frederick Jelinek, y Robert L. Mercer. 1990. A maximum likelihood approach to continuous speech recognition. En *Readings in Speech recognition*, páginas 308–319, San Francisco, CA (Estados Unidos). Morgan Kaufmann Publishers Inc.
- Baptist, L. y S. Seneff. 2000. Genesis-II: A Versatile System for Language Generation in Conversational System Applications. En *Proc. of International Conference on Spoken Language Processing (ICSLP'00)*, páginas 271–274, Beijing (China).
- Benedí, J.M., E. Lleida, A. Varona, M.J. Castro, I. Galiano, R. Justo, I. López, y A. Miguel. 2006. Design and acquisition of a telephone spontaneous speech dialogue corpus in Spanish: DIHANA. En *Proc. of International Conference on Language Resources and Evaluation (LREC'06)*, páginas 1636–1639, Génova (Italia).
- Benedí, J.M., A. Varona, y E. Lleida. 2004. DIHANA: Sistema de diálogo para el acceso a la información en habla espontánea en diferentes entornos. En *Actas de las III Jornadas en Tecnología del Habla*, páginas 141–146, Valencia (España).
- Bennett, Christina, Ariadna Font Llitjós, Stefanie Shriver, Alexander Rudnicky, y Alan W Black. 2002. Building VoiceXML-based applications. En *Proc. of International Conference on Spoken Language Processing (ICSLP'02)*, páginas 2245–2248, Denver (Estados Unidos).
- Bernsen, N.O. y L. Dybkjaer. 1997. The DISC project. En *ELRA Newsletter*, volumen 2(2), páginas 6–8.
-

- Beskow, J., K. Elenius, y S. McGlashan. 1997. Olga - A dialogue system with an animated talking agent. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'97)*, páginas 1651–1654, Rodas (Grecia).
- Biermann, W. y P.M. Long. 1996. The composition of messages in speech-graphics interactive systems. En *Proc. of the International Symposium on Spoken Dialogue*, páginas 97–100, Philadelphia (Estados Unidos).
- Billi, R. y L.F. Lamel. 1997. Railtel: Railway telephone services. En *Speech Communication*, volumen 23, páginas 63–82.
- Bird, Steven y Mark Liberman. 2001. A formal framework for linguistic annotation. En *Speech Communication*, volumen 33, páginas 23–60.
- Bohlin, Peter, Robin Cooper, Elizabet Engdahl, y Staffan Larsson. 1999. Information States and Dialogue Move Engines. En *Proc. of the IJCAI-99 Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, páginas 25–31, Murray Hill (Estados Unidos).
- Bohus, D., S. Grau, D. Huggins-Daines, V. Keri, G. Krishna, R. Kumar, A. Raux, y S. Tomko. 2007a. Conquest - an Open-Source Dialog System for Conferences. En *Proc. of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, páginas 9–12, Rochester, NY (Estados Unidos).
- Bohus, D., A. Raux, T. Harris, M. Eskenazi, y A. Rudnicky. 2007b. Olympus: an open-source framework for conversational spoken language interface research. En *Proc. of HLT-NAACL'07 Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technology*, páginas 32–39, Rochester, NY (Estados Unidos).
- Bohus, D. y A. Rudnicky. 2002. LARRI: A Language-Based Maintenance and Repair Assistant. En *Proc. of IDS'02*, Kloster Irsee (Alemania).
- Bohus, D. y A. Rudnicky. 2003. RavenClaw: Dialog management using hierarchical task decomposition and an expectation agenda. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'03)*, páginas 597–600, Ginebra (Suiza).
- Bonafonte, A., P. Aibar, E. Castell, E. Lleida, J.B. Mariño, E. Sanchís, y M. I. Torres. 2000. Desarrollo de un sistema de diálogo oral en dominios restringidos. En *I Jornadas en Tecnología del Habla*, Sevilla (España).
- Bonneau-Maynard, H. y L. Devillers. 1998. Dialog strategies in a tourist information spoken dialog system. En *Proc. of International Conference Speech and Computer (SPECOM'98)*, páginas 115–118, San Petersburgo (Rusia).
- Bos, J. y T. Oka. 2002. An inference-based approach to dialogue system design. En *Proc. of 19th International Conference on Computational Linguistics, COLING 2002*, páginas 113–119, Taipei (Taiwan).
-

- Bos, Johan, Ewan Klein, Oliver Lemon, y Tetsushi Oka. 1999. The verbmobil prototype system - a software engineering perspective. En *Journal of Natural Language Engineering*, volumen 5(1), páginas 95–112.
- Bos, Johan, Ewan Klein, Oliver Lemon, y Tetsushi Oka. 2003. DIPPER: Description and Formalisation of an Information-State Update Dialogue System Architecture. En *Proc. of 4th SIGdial Workshop on Discourse and Dialogue*, páginas 115–124, Sapporo (Japón).
- Bouwman, G. y J. Hulstijn. 1998. Dialogue strategy redesign with reliability measures. En *Proc. of the First International Conference on Language Resources and Evaluation (LREC'98)*, páginas 191–198, Granada (España).
- Boyer, L., P. Danielsen, J. Ferrans, G. Karam, D. Ladd, B. Lucas, y K. Rehor. 2000. Voice Extensible Markup Language (VoiceXML) Version 1.0. En *Recomendación del W3C*. www.w3.org/TR/voicexml.
- Bray, T., J. Paoli, C. M. Sperberg-McQueen, E. Maler, y F. Yergeau. 2006. Extensible Markup Language (XML) 1.0 (Fourth Edition). www.w3.org/TR/REC-xml. *Recomendación del W3C*.
- Carletta, J., S. Evert, U. Heid, J. Kilgour, J. Robertson, y H. Voormann. 2003. The NITE XML Toolkit: Flexible annotation for multimodal language data. En *Behavior Research Methods, Instruments, and Computers*, volumen 35(3), páginas 353–363.
- Carletta, Jean y Amy Isard. 1999. The MATE Annotation Workbench: User Requirements. En Marilyn Walker, editor, *Towards Standards and Tools for Discourse Tagging: Proceedings of the Workshop*. Association for Computational Linguistics, Somerset, New Jersey (Estados Unidos), páginas 11–17.
- Carlson, R. y S. Hunnicutt. 1996. Generic and domain-specific aspects of the Waxholm NLP and Dialog modules. En *Proc. of International Conference on Spoken Language Processing (ICSLP'96)*, páginas 677–680, Philadelphia (Estados Unidos).
- Cassidy, Steve y Jonathan Harrington. 2004. Multi-level Annotation of Speech: An Overview of the Emu Speech Database Management System. En *Speech Communication*, volumen 33, páginas 61–67.
- Castagneri, G., P. Baggia, y M. Danieli. 1998. Field trials of the Italian ARISE train timetable system. En *Proc. of the Interactive Voice Technology for Telecommunications Applications Workshop (IVTTA'98)*, páginas 97–102.
- Castro, María José, Wladimiro Díaz, Francesc J. Ferri, José Ruiz-Pinales, René Jaime-Rivas, Fernando Blat, Salvador España, Pablo Aibar, Sergio Grau, y David Griol. 2005. A Holistic Classification System for Check Amounts Based on Neural Networks with Rejection. En *Lecture Notes in Computer Science. Pattern Recognition and Machine Intelligence PREMI*, páginas 310–314.
-

- Castro, M.J., S. Grau, N. Alcácer, D. Griol, F. García, R. Granell, y L.F. Hurtado. 2004. Adquisición de un corpus de diálogos para el proyecto DIHANA. Informe técnico, Proyecto DIHANA TIC2002-04103-C03. DSIC. Universitat Politècnica de València, Valencia (España).
- Catizone, R., A. Setzer, y Y. Wilks. 2003. Multimodal Dialogue Management in the COMIC Project. En *Proc. of EAACL'03 Workshop on Dialogue Systems: interaction, adaptation, and styles of management*, páginas 25–34, Budapest (Hungría).
- Cheyer, Adam y David Martin. 2001. The Open Agent Architecture. En *Journal of Autonomous Agents and Multi-Agent Systems*, volumen 4(1), páginas 143–148.
- Chu-Carroll, Jennifer y Jill Suzanne Nickerson. 2000. Evaluating Automatic Dialogue Strategy Adaptation for a Spoken Dialogue System. En *Proc. of the first conference on North American chapter of the Association for Computational Linguistics*, volumen 4, páginas 202–209, Seattle (Estados Unidos).
- Clarkson, Philip y Ronald Rosenfeld. 1997. Statistical Language Modeling Using the CMU–Cambridge Toolkit. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'97)*, páginas 2707–2710, Rodas (Grecia).
- Cohen, P.R. y H.J. Levesque. 1990. Rational interaction as the basis for communication. En *P. R. Cohen, J. Morgan, and M. E. Pollack, editors, Intentions in Communication*. MIT Press.
- Cuayáhuitl, H., S. Renals, O. Lemon, y H. Shimodaira. 2005. Human-Computer Dialogue Simulation Using Hidden Markov Models. En *Proc. of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU'05)*, páginas 290–295.
- Cuayáhuitl, H., S. Renals, O. Lemon, y H. Shimodaira. 2006a. Learning Multi-Goal Dialogue Strategies Using Reinforcement Learning with Reduced State-Action Spaces. En *Proc. of the 9th International Conference on Spoken Language Processing (Interspeech/ICSLP)*, páginas 469–472, Pittsburgh (Estados Unidos).
- Cuayáhuitl, Heriberto, Steve Renals, Oliver Lemon, y Hiroshi Shimodaira. 2006b. Reinforcement learning of dialogue strategies with hierarchical abstract machines. En *Proc. of IEEE/ACL Workshop on Spoken Language Technology (SLT)*, páginas 182–186, Palm Beach (Aruba).
- DARPA. 1992. Speech and Natural Language Workshop. En *Book of Proceedings*, San Mateo (Estados Unidos).
- Day, David, John Aberdeen, Lynette Hirschman, Robyn Kozierok, Patricia Robinson, y Marc Vilain. 1997. Mixed-initiative development of language processing systems. En *Proc. of the fifth conference on Applied natural language processing*, páginas 348–355, San Francisco (Estados Unidos).
- Degerstedt, Lars y Arne Jönsson. 2006. LinTest, A development tool for testing dialogue systems. En *Proc. of the 9th International Conference on Spoken Language Processing (Interspeech/ICSLP)*, páginas 489–492, Pittsburgh (Estados Unidos).
-

- Den, E., L. Boves, L. Lamel, y P. Baggia. 1999. Overview of the ARISE project. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'99)*, volumen 3, páginas 1527–1530, Budapest (Hungría).
- Denecke, Matthias. 2002. Rapid prototyping for spoken dialogue systems. En *Proc. of the 19th International Conference on Computational linguistics*, páginas 1–7, Taipei (Taiwan).
- Devillers, L., H. Maynard, y S. Rosset. 2004. The French Media/Evalda project: the evaluation of the understanding capability of spoken language dialog systems. En *Proc. of International Conference on Language Resources and Evaluation (LREC'04)*, volumen 6, páginas 2131–2134, Lisboa (Portugal).
- Doherty, P., G. Granlund, K. Kuchcinski, E. Sandewall, K. Nordberg, E. Skarman, y J. Wiklund. 1998. The WITAS Unmanned Aerial Vehicle Project. En *Proc. of the 14th European Conference on Artificial Intelligence (ECAI-00)*, páginas 747–755, Berlín (Alemania).
- Dowding, John, Jean Mark Gawron, Douglas E. Appelt, John Bear, Lynn Cherny, Robert Moore, y Douglas B. Moran. 1993. GEMINI: A Natural Language System for Spoken-Language Understanding. En *Proc. of the 31st annual meeting on Association for Computational Linguistics (ACL)*, páginas 54–61, Columbus, Ohio (Estados Unidos).
- Dybkjaer, L. y N. Bernsen. 2000. Usability issues in spoken language dialogue systems. En *Natural Language Engineering (2000)*. Cambridge University Press, volumen 6, páginas 243–271.
- Dybkjaer, L., N. Bernsen, y W. Minker. 2004. Evaluation and usability of multimodal spoken language dialogue systems. En *Speech Communication*, volumen 43, páginas 33–54.
- EAGLES. 1996. Evaluation of Natural Language Processing Systems. Informe técnico, Final Report, EAGLES Document EAG-EWG-PR2. Center for Sprogteknologi, Copenhagen (Dinamarca).
- Eckert, W., E. Levin, y R. Pieraccini. 1997. User modeling for spoken dialogue system evaluation. En *Proc. of IEEE Automatic Speech Recognition and Understanding Workshop (ASRU'97)*, páginas 80–87, Santa Barbara, California (Estados Unidos).
- Eckert, W., E. Levin, y R. Pieraccini. 1998. Automatic evaluation of spoken dialogue systems. Informe técnico, TR98.9.1, ATT Labs Research.
- Elenius, D. 2001. *Härkning ett hot mot talarverifieringssystem?* Ph.D. tesis, CTMH, KTH, Estocolmo (Suecia).
- Fabbrizio, Giuseppe Di y Charles Lewis. 2004. Florence: a Dialogue Manager Framework for Spoken Dialogue Systems. En *Proc. of International Conference on Spoken Language Processing (ICSLP'04)*, páginas 3065–3068, Jeju, Jeju Island, Korea.
-

- Failenschmid, K., D. Williams, L. Dybkjaer, y N.O. Bernsen. 1999. DISC Deliverable D3.6. En *www.disc2.dk*.
- Feng, Junlan, D. Hakkani-Tur, G. Di Fabbrizio, M. Gilbert, y M. Beutnagel. 2006. WebTalk: Towards Automatically Building dialog services by exploiting the content and structure of websites. En *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'06)*, páginas 573–576.
- Ferguson, George y James Allen. 1998. TRIPS: An Intelligent Integrated Problem-Solving Assistant. En *Proc. of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, páginas 567–573, Madison (Estados Unidos).
- Fernández, F., Javier Ferreiros, V. Sama, J.M. Montero, y R. García. 2005. Demostración de una interfaz vocal para el control de un sistema de alta fidelidad. En *Revista de la Sociedad Española de Procesamiento del Lenguaje Natural*, volumen 35, páginas 451–452.
- Flammia, G. y V. Zue. 1995. Empirical Evaluation of Human Performance and Agreement in Parsing Discourse Constituents in Spoken Dialogue. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'95)*, volumen 3, páginas 1965–1968, Madrid (España).
- Fraser, M. y G. Gilbert. 1991. Simulating speech systems. En *Computer Speech and Language*, volumen 5, páginas 81–99.
- Fu, K.S. y T.L. Booth. 1986. Grammatical Inference: Introduction and Survey. Parts I and II. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(3):343–359, 409–423.
- Fukada, T., D. Koll, A. Waibel, y K. Tanigaki. 1998. Probabilistic dialogue extraction for concept based multilingual translation systems. En *Proc. of International Conference on Spoken Language Processing (ICSLP'98)*, volumen 6, páginas 2771–2774, Sydney (Australia).
- Galiano, Isabel. 2004. Estrategia del Mago de Oz en DIHANA. Informe técnico, Proyecto DIHANA TIC2002-04103-C03. DSIC. Universitat Politècnica de València, Valencia (España).
- García, F., L.F. Hurtado, D. Griol, M. Castro, E. Segarra, y E. Sanchis. 2007. Recognition and Understanding Simulation for a Spoken Dialog Corpus Acquisition. En *Springer-Verlag Lecture notes in artificial intelligence (LNAI) 4629*, páginas 574–581, Pilsen (República Checa).
- García, F., L.F. Hurtado, E. Sanchis, y E. Segarra. 2003. The incorporation of Confidence Measures to Language Understanding. En *International Conference on Text Speech and Dialogue (TSD'03). Lecture Notes in Artificial Intelligence series 2807*, páginas 165–172, Ceské Budejovice (República Checa).
- García, Fernando. 2003. *Una aproximación estocástica a la comprensión del lenguaje*. Ph.D. tesis, DSIC - UPV, Valencia (España).
-

- García, P., E. Segarra, E. Vidal, y I. Galiano. 1990. On the use of the morphic generator grammatical inference methodology in automatic speech recognition. En *International Journal of Pattern Recognition and Artificial Intelligence (IJPRIA)*, volumen 4, páginas 667–685.
- Gauvain, J.L., S. Bennacef, L. Devillers, y L. Lamel. 1995. Spoken language system development for the Mask Kiosk. En *Proc. of IEEE Automatic Speech Recognition and Understanding Workshop (ASRU'95)*, páginas 119–120, Salt Lake City (Estados Unidos).
- Georgila, K., J. Henderson, y O. Lemon. 2005. Learning user simulations for information state update dialogue systems. En *Proc. of the 9th European Conference on Speech Communication and Technology (Eurospeech'05)*, páginas 893–896, Lisboa (Portugal).
- Gibbon, Dafydd, Roger Moore, y Richard Winski, 1997. *Handbook of Standards and Resources for Spoken Language Systems*. Mouton de Gruyter; Pbk. Ed edition.
- Gieselmann, Petra y Alex Waibel. 2006. Dynamic extension of a grammar-based dialogue system: Constructing an all-recipes knowing robot. En *Proc. of the 9th International Conference on Spoken Language Processing (Interspeech/ICSLP)*, páginas 449–452, Pittsburgh (Estados Unidos).
- Gilbert, M., J. Wilpon, B. Stern, y G. Di Fabrizio. 2005. Intelligent virtual agents for contact center automation. En *IEEE Signal Processing Magazine*, volumen 22, páginas 32–41.
- Glass, J. 2003. A Probabilistic Framework for Segment-Based Speech Recognition. En *Computer Speech and Language*, volumen 17(2), páginas 137–152.
- Glass, J., G. Flammia, D. Goodine, M. Phillips, J. Polifroni, S. Sakai, S. Seneff, y V. Zue. 1995. Multilingual spoken-language understanding in the MIT Voyager system. En *Speech Communication*, volumen 17, páginas 1–18.
- Glass, J. y E. Weinstein. 2001. SPEECHBUILDER: Facilitating Spoken Dialogue System Development. En *Proc. of European Conference on Speech Communication and Technology*, páginas 1335–1339, Aalborg (Dinamarca).
- Glass, Michael y Barbara Di Eugenio. 2002. MUP: the UIC standoff markup tool. En *Proc. of the 3rd SIGdial Workshop on Discourse and dialogue*, páginas 37–41, Morristown, NJ (Estados Unidos). Association for Computational Linguistics.
- Goddeau, D., H. Meng, J. Polifroni, S. Seneff, y S. Busayapongchai. 1996. A Form-Based Dialogue Manager for Spoken Language Applications. En *Proc. of International Conference on Spoken Language Processing (ICSLP'96)*, volumen 2, páginas 701–704, Philadelphia (Estados Unidos).
- Goffin, V., C. Allauzen, E. Bocchieri, D. Hakkani-Tur, A. Ljolje, S. Parthasarathy, M. Rahim, G. Riccardi, y M. Saraclar. 2005. The AT&T WATSON Speech Recognizer. En *Proc. of International Conference on Acoustics, Speech, and Signal Processing (ICASSP'05)*, páginas 1033–1036, Philadelphia (Estados Unidos).
-

- González, Meritxell y Marta Gatiús. 2005. Un Sistema de Diálogo Multicanal para Acceder a la Información y Servicios de las Administraciones Públicas. En *Revista de la Sociedad Española de Procesamiento del lenguaje natural*, volumen 35, páginas 285–292.
- Gorin, A.L., G. Riccardi, y J.H. Wright. 1997. How may I help you? En *Speech Communication*, volumen 23, páginas 113–127.
- Granell, R., F. Blat, M.J. Castro, S. Grau, y D. Griol. 2005. An Approach to Dialogue Act Classification based on Utterances and Dialogue History. En *Proc. of International Conference Speech and Computer (SPECOM'05)*, páginas 429–432, Patras (Grecia).
- Grau, Sergio, Encarna Segarra, Emilio Sanchis, Fernando García, y Lluís F. Hurtado. 2006. Incorporating semantic knowledge to the language model in a speech understanding system. En *IV Jornadas en Tecnología del Habla*, páginas 145–148, Zaragoza (España).
- Griol, D., F. Blat, M.J. Castro, I. Galiano, F. García, S. Grau, L.F. Hurtado, E. Sanchis, y E. Segarra. 2006. Normativa de Representación Semántica en el Proyecto DIHANA. Informe técnico, Proyecto DIHANA TIC2002-04103-C03. DSIC. Universitat Politècnica de València, Valencia (España).
- Griol, D., L.F. Hurtado, E. Sanchis, y E. Segarra. 2004. Gestor de Diálogo basado en reglas para el Proyecto DIHANA. En *Actas de las III Jornadas en Tecnología del Habla, Valencia (España)*, páginas 53–58, Valencia (España).
- Griol, D., L.F. Hurtado, E. Sanchis, y E. Segarra. 2005. Dos aproximaciones basadas en reglas para la gestión del diálogo. En *Revista de la Sociedad Española de Procesamiento del Lenguaje Natural*, volumen 35, páginas 213–220, Granada (España).
- Griol, D., F. Torres, L.F. Hurtado, S. Grau, F. García, E. Sanchis, y E. Segarra. 2006a. A dialog system for the DIHANA project. En *Proc. of International Conference Speech and Computer (SPECOM'06)*, páginas 131–136, San Petersburgo (Rusia).
- Griol, D., F. Torres, L.F. Hurtado, S. Grau, E. Sanchis, y E. Segarra. 2006b. Development and evaluation of the DIHANA project dialog system. En *Proc. of Interspeech-06 Satellite Workshop Dialogue on Dialogues. Multidisciplinary Evaluation of Advanced Speech-based Interactive Systems*, páginas 9–12, Pittsburgh (Estados Unidos).
- Griol, D., F. Torres, L.F. Hurtado, E. Sanchis, y E. Segarra. 2005. Different approaches to the dialogue management in the DIHANA project. En *Proc. of International Conference Speech and Computer (SPECOM'05)*, páginas 203–206, Patras (Grecia).
- Griol, David, Lluís F. Hurtado, Emilio Sanchis, y Encarna Segarra. 2006a. Evaluación de un Corpus de Diálogos mediante un Gestor de Diálogo Estocástico. En *Proc. of IV Jornadas en Tecnología del Habla*, páginas 129–134, Zaragoza (España).
- Griol, David, Lluís F. Hurtado, Emilio Sanchis, y Encarna Segarra. 2006b. Un método para el tratamiento de situaciones no vistas en la gestión del diálogo. En *Revista*
-

de la Sociedad Española de Procesamiento del Lenguaje Natural, volumen 37, páginas 301–308.

Griol, David, Lluís F. Hurtado, Emilio Sanchis, y Encarna Segarra. 2007a. Acquiring and Evaluating a Dialog Corpus through a Dialog Simulation Technique. En *Proc. of the 8th SIGdial Workshop on Discourse and Dialogue*, páginas 39–42, Amberes (Bélgica).

Griol, David, Lluís F. Hurtado, Emilio Sanchis, y Encarna Segarra. 2007b. Adaptación de un Gestor de Diálogo Estadístico a una Nueva Tarea. En *Revista de la Sociedad Española de Procesamiento del Lenguaje Natural*, volumen 39, páginas 231–238.

Griol, David, Lluís F. Hurtado, Encarna Segarra, y Emilio Sanchis. 2006. Managing unseen situations in a Stochastic Dialog Model. En *Proc. of AAAI Workshop Statistical and Empirical Approaches for Spoken Dialogue Systems*, páginas 25–30, Boston (Estados Unidos).

Griol, David, Encarna Segarra, Lluís F. Hurtado, Francisco Torres, Fernando García, María J. Castro, y Emilio Sanchis. 2007. Sistema de Diálogo Estadístico y Adquisición de un Nuevo Corpus de Diálogos. En *Revista de la Sociedad Española de Procesamiento del Lenguaje Natural*, volumen 39, páginas 283–284.

Gustafson, J., L. Bell, J. Beskow, J. Boye, R. Carlson, J. Edlund, B. Granström, D. House, y M. Wirén. 2000. AdApt - a multimodal conversational dialogue system in an apartment domain. En *Proc. of International Conference on Spoken Language Processing (ICSLP'00)*, volumen 2, páginas 134–137.

Gustafson, J., N. Lindberg, y M. Lundeberg. 1999. The August spoken dialogue system. En *Proc. of International Conference on Spoken Language Processing (Interspeech'99)*, Budapest (Hungría).

Hardy, H., A. Biermann, R.B. Inouye, A. McKenzie, T. Strzalkowski, C. Ursu, N. Webb, y M. Wu. 2006. The Amitiés system: Data-driven techniques for automated dialogue. En *Speech Communication*, volumen 48, páginas 354–373.

Hardy, H., T. Strzalkowski, y M. Wu. 2003. Dialogue management for an automated multilingual call center. En *Proc. of the HLT-NAACL 2003 Workshop on Research Directions in Dialogue Processing*, páginas 10–12, Edmonton (Canadá).

Hazen, T., S. Seneff, y J. Polifroni. 2002. Recognition confidence scoring and its use in speech understanding systems. En *Computer Speech and Language*, volumen 16, páginas 49–67.

Heisterkamp, Paul. 2001. Linguatronic product-level speech system for Mercedes-Benz cars. En *Proc. of the First International Conference on Human Language Technology Research (HLT'01)*, páginas 1–2, San Diego (Estados Unidos).

Hernando, Javier. 2002. Sistema de Información Meteorológica Automática por Teléfono ATTEMPS. En *Revista de la Sociedad Española de Procesamiento del Lenguaje Natural*, volumen 29, páginas 311–312.

- Holzapfel, Hartwig. 2005. Towards development of multilingual spoken dialogue systems. En *Proc. of the 2nd Language and Technology Conference (L&T'05)*, Poznan (Polonia).
- Horvitz, E. y T. Paek. 1999. A computational architecture for conversation. En *Proc. of the 7th International Conference on User Modeling*, páginas 201–210, Banff (Canadá).
- Hunt, A. y S. McGlashan. 2004. Speech Recognition Grammar Specification Version 1.0. Recomendación del W3C. En www.w3.org/TR/speech-grammar/.
- Hurtado, L., E. Segarra, F. García, y E. Sanchis. 2004. Language Understanding using n-multigram models. En *Advances in Natural Language Processing, Proceedings of 4th International Conference EsTAL*, volumen 3230 de *Lecture Notes in Computer Science*. Springer-Verlag, páginas 207–219.
- Hurtado, L.F, F. Blat, F. García, S. Grau, D. Griol, E. Sanchis, E. Segarra, y F. Torres. 2005. Sistema de diálogo para el Proyecto DIHANA. En *Revista de la Sociedad Española de Procesamiento del Lenguaje Natural*, volumen 35, páginas 453–454, Granada (España).
- Hurtado, Lluís F. 2004. *Modelos de lenguaje basados en palabras y segmentos y su aplicación a la comprensión del habla*. Ph.D. tesis, DSIC - UPV, Valencia (España).
- Hurtado, Lluís F., Fernando Blat, y David Griol. 2006. Extensión del formato XML para las comunicaciones en el sistema de Diálogo DIHANA. Informe técnico, Proyecto DIHANA TIC2002-04103-C03. DSIC. Universitat Politècnica de València, Valencia (España).
- Hurtado, Lluís F., David Griol, Emilio Sanchis, y Encarna Segarra. 2005. A stochastic approach to dialog management. En *Proc. of IEEE Workshop Automatic Speech Recognition and Understanding (ASRU'05)*, páginas 226–231, San Juan (Puerto Rico).
- Hurtado, Lluís F., David Griol, Encarna Segarra, y Emilio Sanchis. 2006. A Stochastic Approach for Dialog Management based on Neural Networks. En *Proc. of the 9th International Conference on Spoken Language Processing (Interspeech/ICSLP)*, páginas 49–52, Pittsburgh (Estados Unidos).
- Hurtado, Lluís F., David Griol, Encarna Segarra, y Emilio Sanchis. 2007a. A Statistical User Simulation Technique for the Improvement of a Spoken Dialog System. En *Lecture Notes in Computer Science LNCS. Progress in Pattern Recognition, Image Analysis and Applications*, volumen 4756, páginas 743–752.
- Hurtado, Lluís F., David Griol, Encarna Segarra, y Emilio Sanchis. 2007b. Adapting a Statistical Dialog for a New Domain. En *Proc. of the 11th Workshop on the Semantics and Pragmatics of Dialogue, DECALOG'07*, páginas 171–172, Rovereto (Italia).
- Ierusalimschy, R., L. H. de Figueiredo, y W. Celes. 2003. Lua 5.0 reference manual. Informe técnico, PUC-Rio.
- Ierusalimschy, Roberto. 2003. *Programming in Lua*. Published by Lua.org.
-

- Ito, Akinori, Keisuke Shimada, Motoyuki Suzuki, y Shozo Makino. 2006. A User Simulator Based on VoiceXML for Evaluation of Spoken Dialog Systems. En *Proc. of the 9th International Conference on Spoken Language Processing (Interspeech/ICSLP)*, páginas 1045–1048, Pittsburgh (Estados Unidos).
- Jelinek, F. 1990. Self-organized language modeling for speech recognition. En *Readings in speech recognition*, páginas 450–506, San Francisco, CA (Estados Unidos). Morgan Kaufmann Publishers Inc.
- Jelinek, F., J.D. Lafferty, y R.L. Mercer. 1992. Basic methods of probabilistic context free grammars. En *Speech Recognition and Understanding. Recent Advances, Trends and Applications. Springer Verlag*, páginas 345–360.
- Johansson, M. 1997. Phoneme recognition as a hearing aid in telephone communication. En *MA Thesis. Dept. of Linguistics. Uppsala University, Uppsala (Suecia)*.
- Jokinen, Kristiina, Kari Kanto, Antti Kerminen, y Jyrki Rissanen. 2004. Evaluation of Adaptivity and User Expertise in a Speech-Based E-Mail System. En *Proc. of the 20th ACL International Conference on Computational Linguistic*, páginas 44–52, Ginebra (Suiza).
- Jokinen, Kristiina, Kari Kanto, y Jyrki Rissanen. 2004. Adaptative User Modelling in AthosMail. En *Lecture Notes in Computer Science. Springer*, volumen 3196/2004, páginas 149–158.
- Juhar, J., S. Ondas, A. Cizmar, M. Rusko, G. Rozinaj, y R. Jarina. 2006. Development of Slovak GALAXY/VoiceXML Based Spoken Language Dialogue System to Retrieve Information from Internet. En *Proc. of the 9th International Conference on Spoken Language Processing (Interspeech/ICSLP)*, páginas 485–488, Pittsburgh (Estados Unidos).
- Justo, Raquel, M. Inés Torres, y Lluís F. Hurtado. 2006. Modelos de lenguaje basados en categorías semánticas en un sistema de diálogo de habla espontánea en castellano. En *Proc. of IV Jornadas en Tecnologia del Habla*, páginas 89–92, Zaragoza (España).
- Kaelbling, Leslie Pack, Michael L. Littman, y Anthony R. Cassandra. 1998. Planning and acting in partially observable stochastic domains. En *Artificial Intelligence*, volumen 101, páginas 99–134.
- Kearns, Michael, Charles Isbell, Satinder Singh, Diane Litman, y Jessica Howe. 2002. CobotDS: A Spoken Dialogue System for Chat. En *Proc. of the Eighteenth National Conference on Artificial Intelligence (AAAI-2002)*, páginas 425–430, Edmonton (Canadá).
- Kowtko, J., S. Isard, y G.M. Doherty. 1991. Conversational games within dialogue. En *Proc. of the ESPRIT Workshop on Discourse Coherence*, páginas 4–6, Edimburgo (Escocia).
-

- Krsmanovic, Filip, Curtis Spencer, Daniel Jurafsky, y Andrew Y. Ng. 2006. Have we meet? MDP Based Speaker ID for Robot Dialogue. En *Proc. of the 9th International Conference on Spoken Language Processing (Interspeech/ICSLP)*, páginas 461–464, Pittsburgh (Estados Unidos).
- Lamel, L. 1998. Spoken Language Dialog System Development and Evaluation at LIMSI. En *Proc. of the International Symposium on Spoken Dialogue (ISSD'98)*, páginas 9–17, Sydney (Australia).
- Lamel, L., S. Bennacef, S. Rosset, L. Devillers, S. Foukia, J. Gangolf, y J. Gauvain. 1997. The LIMSI RailTel System: Field trail of a telephone service for rail travel information. *Speech Communication*, 23:67–82.
- Lamel, L., S. Rosset, J. Gauvain, y S. Bennacef. 1999. The LIMSI ARISE system for train travel information. En *Proc. of International Conference on Acoustics, Speech, and Signal Processing (ICASSP'99)*, páginas 501–504, Phoenix, Arizona (Estados Unidos).
- Lamel, L., S. Rosset, J. Gauvain, S. Bennacef, M. Garnier-Rizet, y B. Prouts. 2000. The LIMSI ARISE system. En *Speech Communication*, volumen 31, páginas 339–353.
- Larsson, S., A. Berman, J. Bos, L. Grönqvist, y P. Ljunglöf. 1999. A model of dialogue moves and information state revision. Informe técnico, D5.1 Trindi (Task Oriented Instructional Dialogue).
- Lau, Raymond, Giovanni Flammia, Christine Pao, y Victor Zue. 1997. WebGalaxy. Integrating spoken language and hypertext navigation. En *Proc. of the International Conference on Spoken Language Processing (InterSpeech'97)*, páginas 883–886, Rodas (Grecia).
- Lehtinen, G., S. Safra, M. Gauger, J.L. Cochard, B. Kaspar, M. Hennecke, A. Tsopanoglou, D. Louloudis, y M. Mantakas. 2000. IDAS : Interactive Directory Assistance Services. En *Proc. of VOTS-2000 Workshop*, Bélgica.
- Lemon, Oliver, Kallirrogi Georgila, y James Henderson. 2006. Evaluating Effectiveness and Portability of Reinforcement Learned Dialogue Strategies with real users: the TALK TownInfo Evaluation. En *Proc. of IEEE-ACL Workshop on Spoken Language Technology (SLT 2006)*, páginas 178–181, Palm Beach (Aruba).
- Levin, E., S. Narayanan, R. Pieraccini, K. Biatov, E. Bocchieri, G. Di Fabbrizio, W. Eckert, S. Lee, A. Pokrovsky, M. Rahim, P. Ruscitti, y M. Walker. 2000. The AT&T-Darpa Communicator Mixed-Initiative Spoken Dialog System. En *Proc. of the International Conference in Spoken Language Processing (ICSLP'00)*, páginas 122–125, Beijing, China).
- Levin, E. y R. Pieraccini. 1995. Chronus, the next generation. En *Proc. of the DARPA Speech and Natural Language Workshop*, páginas 269–271, Austin, Texas (Estados Unidos).
-

- Levin, E. y R. Pieraccini. 1997. A stochastic model of human-machine interaction for learning dialog strategies. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'97)*, páginas 1883–1896, Rodas (Grecia).
- Levin, E., R. Pieraccini, y W. Eckert. 2000. A stochastic model of human-machine interaction for learning dialog strategies. En *IEEE Transactions on Speech and Audio Processing*, volumen 8(1), páginas 11–23.
- Litman, Diane, Satinder Singh, Michael Kearns, y Marilyn Walker. 2000. NJFun: a reinforcement learning spoken dialogue system. En *Proceedings of the ANLP/NAACL 2000 Workshop on Conversational Systems*, páginas 17–20, Seattle (Estados Unidos). Association for Computational Linguistics.
- Litman, Diane J. y Shimei Pan. 2002. Designing and Evaluating an Adaptive Spoken Dialogue System. En *User Modeling and User-Adapted Interaction*, volumen 12(2-3), páginas 111–137.
- Litman, Diane J. y Scott Silliman. 2004. ITSPOKE: An Intelligent Tutoring Spoken Dialogue System. En *Proc. of the Human Language Technology Conference: 4th Meeting of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL)*, páginas 233–236, Boston (Estados Unidos).
- Littman, M.L. 1994. The witness algorithm: Solving partially observable Markov decision processes. Informe técnico, CS-94-40, Brown University, Department of Computer Science, Providence, RI.
- Lleida, Eduardo. 2005. Librería de comunicaciones basada en paquetes XML: DIHANNA v2.0. Informe técnico, Grupo de Tecnologías de las Comunicaciones. I3A. Centro Politécnico Superior. Universidad de Zaragoza, Zaragoza (España).
- Lleida, Eduardo, Encarna Segarra, M. Inés Torres, y J. Macías-Guarasa. 2006. EDE-CÁN: sistEma de Diálogo multidominio con adaptación al contExto aCústico y de Aplicación. En *IV Jornadas en Tecnología del Habla*, páginas 291–296, Zaragoza (España).
- Llisterri, J. y M.J. Machuca, 2006. *Los sistemas de diálogo*, capítulo 5. E. Segarra: La interpretación semántica. Bellaterra - Soria: Universitat Autònoma de Barcelona, Servei de Publicacions - Fundación Duques de Soria (Manuals de la Universitat Autònoma de Barcelona, Lingüística, 45).
- López-Cózar, R., Z. Callejas, G. Montoro, y P. Haya. 2006. DS-UCAT: Sistema de diálogo multimodal y multilingüe para un entorno educativo. En *Proc. IV Jornadas en Tecnología del Habla*, páginas 135–140, Zaragoza (España).
- López-Cózar, R., P. García, J. Díaz, y A. J. Rubio. 1997. A voice activated dialogue system for fast-food restaurant applications. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'97)*, páginas 1783–1786, Rodas (Grecia).
-

- López-Cózar, R. y R. Granell. 2004. Sistema de Diálogo Basado en VoiceXML para Proporcionar Información de Viajes en Tren. En *Revista de la Sociedad Española de Procesamiento del Lenguaje Natural*, volumen 33, páginas 171–178.
- López-Cózar, R. y A. J. Rubio. 1997. SAPLEN: Un Sistema de Diálogo en Lenguaje Natural para una Aplicación Comercial. En *Revista de la Sociedad Española de Procesamiento del Lenguaje Natural*, volumen 20, páginas 65–81.
- López-Cózar, R., A.J. Rubio, P. García, y J.C. Segura. 1999. Uso de Valores de Confianza y Expectativas en el Sistema de Diálogo SAPLEN. En *Revista de la Sociedad Española de Procesamiento del Lenguaje Natural*, volumen 24, páginas 37–41.
- López-Cózar, R., E. Sanchís, E. Segarra, y J.M. Benedí. 2004. Incorporación de VoiceXML en el Sistema de Diálogo DIHANA. En *Proc. del V Congreso de Interacción Persona-Ordenador*, páginas 253–256, Lleida (España).
- Macías-Guarasa, J., L.F. D'Haro, J.M. Pardo, F. Fernández, Ricardo de Córdoba, V. Sama, Javier Ferreiros, R. San-Segundo, y J.M. Montero. 2004. Realización de sistemas de diálogo en una plataforma compatible con VoiceXML: Proyecto GEMINI. En *Revista de la Sociedad Española de Procesamiento del Lenguaje Natural*, volumen 33, páginas 103–110.
- Macherey, K. y H. Ney. 2003. Features of tree based dialogue course management. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'03)*, volumen 1, páginas 601–604, Ginebra (Suiza).
- Mariño, José B. 2006. Avivavoz: Tecnologías para la traducción de voz. En *IV Jornadas en Tecnología del Habla*, páginas 285–290, Zaragoza (España).
- Martínez, C., E. Sanchis, F. García, y P. Aibar. 2002. A labeling proposal to annotate dialogues. En *Proc. of International Conference on Language Resources and Evaluation (LREC'02)*, páginas 1577–1582, Las Palmas de Gran Canaria (España).
- Matheson, C., M. Poesio, y D. Traum. 2000. Modelling grounding and discourse obligations using update rules. En *Proc. of the 1st Annual Meeting of the North American Association for Computational Linguistics (NAACL'00)*, páginas 1–8, Seattle (Estados Unidos).
- McCallum, A. y K. Nigam. 1998. A comparison of event models for Naive Bayes text classification. En *Proc. of AAAI-98 Workshop on Learning for Text Categorization*, páginas 41–48, Madison (Estados Unidos).
- McGlashan, S., D.C. Burnett, J. Carter, P. Danielsen, J. Ferrans, A. Hunt, B. Lucas, B. Porter, K. Rehor, y S. Tryphonas. 2004. Voice Extensible Markup Language (VoiceXML) Version 2.0. En *Recomendación del W3C*. www.w3.org/TR/voicexml20/.
- McTear, M. 1998. Modelling Spoken Dialogues with State Transition Diagrams: Experiences with the CSLU Toolkit. En *Proc. of 5th International Conference on Spoken Language Processing (ICSLP'98)*, volumen 4, páginas 1223–1226, Sydney (Australia).
-

- McTear, Michael F., 2004. *Spoken Dialogue Technology: Towards the Conversational User Interface*. Springer.
- Melin, H. 2001. ATLAS: A generic software platform for speech technology based applications. En *TMH Quarterly Progress and Status Report (TMH-QPSR)*, volumen 1, páginas 29–42.
- Melin, H., A. Sandell, y M. Ihse. 2001. CTT-bank: A speech controlled telephone banking system - an initial evaluation. En *TMH Quarterly Progress and Status Report (TMH-QPSR)*, volumen 1, páginas 1–27.
- Meng, H., S. Busayapongchai, J. Glass, D. Goddeau, L. Hetherington, E. Hurley, C. Pao, J. Polifroni, S. Seneff, y V. Zue. 1996. WHEELS: A Conversational System in the Automobile Classifieds Domain. En *Proc. of International Conference on Spoken Language Processing (ICSLP'96)*, volumen 1, páginas 542–545, Philadelphia (Estados Unidos).
- Meng, Heleng H., Carmen Wai, y Roberto Pieraccini. 2003. The Use of Belief Networks for Mixed-Initiative Dialog Modeling. En *IEEE Transactions on Speech and Audio Processing*, volumen 11(6), páginas 757–773.
- Meng, H.M., S. Lee, y C. Wai. 2000. CU FOREX: a bilingual spoken dialog system for foreign exchange enquiries. En *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'00)*, volumen 2, páginas 229–232, Estambul (Turquía).
- Miguel, A., M.I. Galiano, R. Granell, L.F. Hurtado, y E. Sanchis. 2003. La plataforma de adquisición de diálogos en el proyecto DIHANA. En *Revista de la Sociedad Española para el Procesamiento del Lenguaje Natural*, volumen 31, páginas 343–344.
- Minker, W. 1998. Stochastic versus rule-based speech understanding for information retrieval. En *Speech Communication*, volumen 25(4), páginas 223–247.
- Minker, W., U. Haiber, P. Heisterkamp, y S. Scheible. 2004. The Seneca Spoken Language Dialogue System. En *Speech Communication*, volumen 43, páginas 1–2.
- Minker, W., A. Waibel, y J. Mariani. 1999. *Stochastically-Based Semantic Analysis*. Kluwer Academic Publishers, Dordrecht (Holanda).
- Mäkelä, Kaj, Esa Pekka Salonen, Markku Turunen, Jaakko Hakulinen, y Roope Raisamo. 2001. Conducting a Wizard of Oz Experiment on a Ubiquitous Computing System Doorman. En *Proc. of the International Workshop on Information Presentation and Natural Multimodal Dialogue*, páginas 115–119, Verona (Italia).
- Müller, Christel y Fred Runge. 1993. Dialogue design principles - key for usability of voice processing. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'93)*, páginas 943–946, Berlín (Alemania).
- Müller, Christoph y Michael Strube. 2003. Multi-Level Annotation in MMAX. En *Proc. of the 4th SIGdial Workshop on Discourse and Dialogue*, páginas 61–67, Sapporo (Japón).
-

- Montero, J.M. 1998. Manual del SERVIVOX. En *GTH-DIE, UPM*.
- Netter, K., S. Armstrong, T. Kiss, J. Klein, S. Lehmann, D. Milward, S. Regnier-Prost, R. Schäer, y T. Wegst. 1998. DiET - Diagnostic and Evaluation Tools for Natural Language Applications. En *Proc. of the 1st International Conference on Language Resources and Evaluation (LREC'98)*, Granada (España).
- Nielsen, Palle Bach y Anders Baekgaard. 1992. Experience with a dialogue description formalism for realistic applications. En *Proc. of International Conference on Spoken Language Processing (ICSLP'92)*, páginas 719–722, Banff (Canadá).
- O'Neill, Ian, Philip Hanna, Xingkun Liu, y Michael McTear. 2003. The Queen's Communicator: An Object-Oriented Dialogue Manager. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'03)*, páginas 593–596, Ginebra (Suiza).
- Packucs, B. y H. Melin. 2001. PER: A speech based automated entrance receptionist. En *Proc. of of the 13th Nordic Conference of Computational Linguistics (NoDaLiDa'01)*, Uppsala (Suecia).
- Paek, T. y E. Horvitz. 1999. Uncertainty, utility, and misunderstanding: A decision-theoretic perspective on grounding in conversational systems. En *Proc. of AAAI Fall Symposium on Psychological Models of Psychological Models of Communication*, páginas 85–92, North Falmouth (Estados Unidos).
- Paek, T. y E. Horvitz. 2000. Conversation as action under uncertainty. En *Proc. of 16th Conference on Uncertainty in Artificial Intelligence*, páginas 455–464, San Francisco (Estados Unidos).
- Pargellis, A., H. Kuo, y C. Lee. 1999. Automatic dialogue generator creates user defined applications. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'99)*, páginas 1175–1178, Budapest (Hungria).
- Pargellis, A., H. Kuo, y C. Lee. 2004. An automatic dialogue generation platform for personalized dialogue applications. En *Speech Communication*, volumen 42, páginas 329–351.
- Peckham, J. 1993. A new generation of spoken dialogue systems: results and lessons from the SUNDIAL project. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'93)*, volumen 1, páginas 33–42, Berlín (Alemania).
- Pérez, Guillermo, Gabriel Amores, y Pilar Manchón. 2006. A multimodal architecture for home control by disabled users. En *Proc. of IEEE/ACL Workshop on Spoken Language Technology (SLT)*, páginas 134–137, Palm Beach (Aruba).
- Pérez-Piñar, D. y C. García. 2005a. Application of confidence measures for dialogue systems through the use of parallel speech recognizers. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'05)*, páginas 2785–2788, Lisboa (Portugal).
-

- Pérez-Piñar, David y Carmen García. 2005b. Verificación de tema en sistemas de diálogo mediante la aplicación de un test de hipótesis bayesiano. En *Revista de la Sociedad Española de Procesamiento del Lenguaje Natural*, volumen 35, páginas 221–227.
- Pieraccini, R., E. Levin, y W. Eckert. 1997. AMICA: The AT&T mixed initiative conversational architecture. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'97)*, páginas 1875–1878, Rodas (Grecia).
- Pietquin, O. y R. Beaufort. 2005. Comparing ASR modeling methods for spoken dialogue simulation and optimal strategy learning. En *Proc. of the 9th European Conference on Speech Communication and Technology (Interspeech/Eurospeech'05)*, páginas 861–864, Lisboa (Portugal).
- Pietquin, O. y T. Dutoit. 2005. A probabilistic framework for dialog simulation and optimal strategy learning. En *IEEE Transactions on Speech and Audio Processing, Special Issue on Data Mining of Speech, Audio and Dialog*, volumen 14, páginas 589–599.
- Pineau, J., G. Gordon, y S. Thrun. 2003. Point-based value iteration: An anytime algorithm for POMDPs. En *Proc. of International Joint Conference on Artificial Intelligence (IJCAI)*, páginas 1025–1032, Acapulco (México).
- Polifroni, Joseph, Grace Chungand, y Stephanie Seneff. 2003. Towards the Automatic Generation of Mixed-Initiative Dialogue Systems from Web Content. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'03)*, páginas 193–196, Ginebra (Suiza).
- Potamianos, A., E. Ammicht, y E. Fosler-Lussier. 2003. Modality tracking in the Multimodal Bell Labs Communicator. En *Proc. of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU 03)*, páginas 192–197, St. Thomas, U.S. Virgin Islands.
- Purver, Matthew, Florin Ratiu, y Lawrence Cavedon. 2006. Robust Interpretation in Dialogue by Confidence Scores with Contextual Features. En *Proc. of the 9th International Conference on Spoken Language Processing (Interspeech/ICSLP)*, páginas 1–4, Pittsburgh (Estados Unidos).
- Rabiner, L.R., B.H. Juang, y C.H. Lee. 1996. An overview of automatic speech recognition. En Kluwer Academic Publishers, editor, *Automatic Speech and speaker Recognition: Advanced Topic*, páginas 1–30.
- Rahim, M., R. Pieraccini, W. Eckert, E. Levin, G. Di Fabbrizio, G. Riccardi, C. Kamm, y S. Narayanan. 2000. A Spoken Dialog System for Conference/Workshop Services. En *Proc. of International Conference on Spoken Language Processing (ICSLP'00)*, páginas 736–739, Beijing (China).
- Raux, A., B. Langner, A. Black, y M. Eskenazi. 2005. Let's Go Public! Taking a Spoken Dialog System to the Real World. En *Proc. of the International Conference on Spoken Language Processing (Interspeech'05)*, páginas 885–888, Lisboa (Portugal).
-

- Reithinger, Norbert y Michael Kipp. 1998. Large Scale Dialogue Annotation in Verbobil. En *Proc. of the Workshop on Recent Advances in Corpus Annotation at the 10th European Summer School in Logic, Language and Information (ESSLLI-98)*, Saarbrücken (Alemania).
- Rodríguez, Leandro, Víctor Darriba, Carmen García, y Santiago Pardo. 2002. Un sistema de diálogo para la consulta de correo electrónico en lenguaje natural. En *Revista de la Sociedad Española de Procesamiento del Lenguaje Natural*, volumen 29, páginas 181–188.
- Roque, A., A. Leuski, V. Rangarajan, S. Robinson, A. Vaswani, S. Narayanan, y D. Traum. 2006. Radiobot-CFF: A Spoken Dialogue System for Military Training. En *Proc. of the 9th International Conference on Spoken Language Processing (Interspeech/ICSLP)*, páginas 477–480, Pittsburgh (Estados Unidos).
- Rosenfeld, R. 1995. The CMU Statistical Language Modeling Toolkit and its use in the 1994 ARPA CSR Evaluation. En *Proc. of the ARPA Spoken Language Systems Technology Workshop*, San Mateo (Estados Unidos).
- Rosenfeld, Ronald, Xiaojin Zhu, Stefanie Shriver, Arthur Toth, Kevin Lenzo, y Alan W Black. 2000. Towards a universal speech interface. En *Proc. of International Conference on Spoken Language Processing (ICSLP'00)*, volumen 2, páginas 102–105, Beijing (China).
- Rudnický, A., E. Thayer, P. Constantinides, C. Tchou, R. Shern, K. Lenzo, W. Xu, y A. Oh. 1999. Creating natural dialogs in the Carnegie Mellon Communicator system. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'99)*, volumen 1(4), páginas 1531–1534.
- Rumelhart, D. E., G. E. Hinton, y R. J. Williams, 1986. *PDP: Computational models of cognition and perception, I*, capítulo Learning internal representations by error propagation, páginas 319–362. MIT Press.
- Sama, V., J. Ferreiros, F. Fernández, R. San Segundo, y J. Pardo. 2005. Utilización de medidas de confianza en sistemas de comprensión del habla. En *Revista de la Sociedad Española de Procesamiento del Lenguaje Natural*, volumen 35, páginas 229–234.
- San-Segundo, R., JM. Montero, J. Ferreiros, R. Córdoba, y JM. Pardo. 2001a. Designing Confirmation Mechanisms and Error Recover Techniques in a Railway Information System for Spanish. En *Proc. of the Second SIGdial Workshop on Discourse and Dialogue*, Aalborg (Dinamarca).
- San-Segundo, R., B. Pellom, K. Hacioglu, W. Ward, y J.M. Pardo. 2001b. Confidence measures for spoken dialogue systems. En *Proc. of International Conference on Acoustics, Speech, and Signal Processing (ICASSP'01)*, volumen 1, páginas 393–396, Salt Lake City, Utah (Estados Unidos).
- San-Segundo, Rubén. 2004. La evaluación objetiva de sistemas de diálogo. En *Curso de Tecnologías Lingüísticas. Fundación Duques de Soria*, Soria (España).
-

- Sanchis, E., D. Griol, L.F. Hurtado, y E. Segarra. 2006. A platform for the development of spoken dialog systems. En *Proc. of International Conference on Software Engineering Advances (ICSEA 2006)*, páginas 69–73, Tahití.
- Schatzmann, J., K. Georgila, y S. Young. 2005. Quantitative Evaluation of User Simulation Techniques for Spoken Dialogue Systems. En *Proc. of 6th SIGdial Workshop on Discourse and Dialogue*, páginas 45–54, Lisboa (Portugal).
- Schatzmann, J., K. Weilhammer, M. Stuttle, y S. Young. 2005. Effects of the User Model on Simulation-based Learning of Dialogue Strategies. En *Proc. of IEEE Workshop Automatic Speech Recognition and Understanding (ASRU'05)*, páginas 220–225, San Juan (Puerto Rico).
- Schatzmann, J., K. Weilhammer, M. Stuttle, y S. Young. 2006. A Survey of Statistical User Simulation Techniques for Reinforcement-Learning of Dialogue Management Strategies. En *Knowledge Engineering Review*, volumen 21(2), páginas 97–126.
- Scheffler, K. y S. Young. 1999. Simulation of human-machine dialogues. Informe técnico, CUED/F-INFENG/TR 355, Cambridge University Engineering Dept., Cambridge (Reino Unido).
- Scheffler, K. y S. Young. 2000. Probabilistic simulation of human-machine dialogues. En *Proc. of International Conference on Acoustics, Speech, and Signal Processing (ICASSP'00)*, páginas 1217–1220, Estambul (Turquía).
- Scheffler, K. y S. Young. 2001a. Automatic learning of dialogue strategy using dialogue simulation and reinforcement learning. En *Proc. of Human Language Technology (HLT'02)*, páginas 12–18, San Diego (Estados Unidos).
- Scheffler, Konrad y Steve Young. 2001b. Corpus-based Dialogue Simulation for Automatic Strategy Learning and Evaluation. En *Proc. of the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-2001). Workshop on Adaptation in Dialogue Systems*, Pittsburgh (Estados Unidos).
- Searle, John. 1969. Speech acts: An essay in the philosophy of language. En *Cambridge University Press*, Cambridge (Reino Unido).
- Segarra, E. y L. Hurtado. 1997. Construction of Language Models using Morfic Generator Grammatical Inference MGGI Methodology. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'97)*, páginas 2695–2698, Rodas (Grecia).
- Segarra, E., E. Sanchis, F. García, y L.F. Hurtado. 2002. Extracting semantic information through automatic learning techniques. En *International Journal of Pattern Recognition and Artificial Intelligence*, volumen 16(3), páginas 301–307, Salt Lake City (Estados Unidos).
- Segarra, Encarna. 1993. *Una aproximación inductiva a la comprensión del discurso continuo*. Ph.D. tesis, DSIC - UPV, Valencia (España).
-

- Seneff, S. 1992. TINA: A natural language system for spoken language applications. En *Computational Linguistics*, volumen 18(1), páginas 61–86.
- Seneff, S., C. Chuu, y D.S. Cyphers. 2000. Orion: From On-line Interaction to Off-line Delegation. En *Proc. of International Conference on Spoken Language Processing (ICSLP'00)*, volumen 2, páginas 142–145, Beijing (China).
- Seneff, S., L. Hirschman, y V. Zue. 1991. Interactive problem solving and dialogue in the ATIS domain. En *Proc. of the Fourth ARPA Speech and Natural Language Workshop*, páginas 354–359, Pacific Grove, California (Estados Unidos).
- Seneff, S., E. Hurley, R. Lau, C. Pao, P. Schmid, y V. Zue. 1998. Galaxy-II: A reference architecture for conversational system development. En *Proc. of International Conference on Spoken Language Processing (ICSLP'98)*, páginas 931–934, Sydney (Australia).
- Seneff, S., R. Lau, y J. Polifroni. 1999. Organization, communication, and control in the galaxy-II conversational system. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'99)*, páginas 1271–1274, Budapest (Hungria).
- Seneff, S. y J. Polifroni. 1996. A New Restaurant Guide Conversational System: Issues in Rapid Prototyping for Specialized Domains. En *Proc. of International Conference on Spoken Language Processing (ICSLP'96)*, volumen 2, páginas 665–668, Philadelphia (Estados Unidos).
- Seneff, S. y J. Polifroni. 2000. Dialogue management in the Mercury flight reservation system. En *Proc. of the 1st Conference of the North American Chapter of the Association for Computational Linguistics and 6th Conference on Applied Natural Language Processing (ANLP-NAACL 2000)*, páginas 11–16, Seattle (Estados Unidos).
- Singh, S., M.S. Kearns, D. Litman, y M. Walker. 1999. Reinforcement learning for spoken dialogue systems. En *Proc. of Neural Information Processing Systems (NIPS'99)*, páginas 956–962, Denver (Estados Unidos).
- Singh, S., D. Litman, M. Kearns, y M. Walker. 2002. Optimizing dialogue management with reinforcement learning: Experiments with the NJFun system. En *Journal of Artificial Intelligence Research (JAIR)*, volumen 16, páginas 105–133.
- Skantze, G., J. Edlund, y R. Carlson. 2006. Talking with Higgins: Research challenges in a spoken dialogue system. En *Proc. of Perception and Interactive Technologies (PIT'06)*, páginas 193–196, Kloster Irsee (Alemania).
- Sturm, J., E. den Os, y L. Boves. 1999a. Dialogue management in the Dutch Arise train timetable information system. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'99)*, páginas vol. 3, 1419–1422, Budapest (Hungria).
- Sturm, J., E. den Os, y L. Boves. 1999b. Issues in Spoken Dialogue Systems: Experiences with the Dutch ARISE System. En *Proceedings of ESCA Workshop on Interactive Dialogue in MultiModal Systems*, páginas 1–4, Kloster Irsee (Alemania).
-

- Sutton, Richard S. y Andrew G. Barto. 1998. Reinforcement learning: An introduction. En *A Bradford Book. The MIT Press*, Cambridge, Massachusetts (Estados Unidos).
- Tichelen, L.V. 2003. Semantic Interpretation for Speech Recognition. En *W3C Working Draft*.
- Tomko, S. y R. Rosenfeld. 2004. Speech Graffiti vs. Natural Language: Assessing the User Experience. En *Proc. of Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL'04)*, páginas 73–76, Boston (Estados Unidos).
- Torres, F., L.F. Hurtado, F. García, E. Sanchis, y E. Segarra. 2005. Error handling in a stochastic dialog system through confidence measures. En *Speech Communication*, volumen 45(3), páginas 211–229.
- Torres, F., E. Sanchis, y E. Segarra. 2003. Development of a stochastic dialog manager driven by semantics. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'03)*, páginas 605–608, Ginebra (Suiza).
- Torres, F., E. Sanchis, y E. Segarra. 2005. Learning of stochastic dialog models through a dialog simulation technique. En *Proc. of the 9th European Conference on Speech Communication and Technology (Eurospeech'05)*, Lisboa (Portugal).
- Torres, Francisco. 2006. *Sistemas de diálogo basados en modelos estocásticos*. Ph.D. tesis, DSIC - UPV, Valencia (España).
- Traum, D., J. Bos, R. Cooper, S. Larsson, I. Lewin, C. Matheson, y M. Poesio. 1999. A model of dialogue moves and information state revision. En *Tech. rept. Deliverable D2.1. Trindi*.
- TRINDI Consortium. 2001. TRINDI (Task Oriented Instructional Dialogue) Book Draft. www.ling.gu.se/projekt/trindi/book.ps.
- Turunen, Markku y Jaakko Hakulinen. 2000. Mailman - a Multilingual Speech-only E-mail Client Based on an Adaptive Speech Application Framework. En *Proc. of Workshop on Multi-Lingual Speech Communication (MSC'00)*, páginas 7–12, Kyoto (Japón).
- Turunen, Markku, Jaakko Hakulinen, Kari-Jouko Rähkä, Esa-Pekka Salonen, Anssi Kainulainen, y Perttu Prusi. 2005. An architecture and applications for speech-based accessibility systems. En *IBM Systems Journal*, volumen 44(3), páginas 485–504.
- Uría, A., A. Ortega, M.I. Torres, A. Miguel, V. Gujarrubia, L. Buera, J. Garmendia, E. Lleida, O. Aizpuru, A. Varona, E. Alonso, y O. Saz. 2006. A virtual butler controlled by speech. En *Proc. IV Jornadas en Tecnología del Habla*, páginas 297–301, Zaragoza (España).
- VanLehn, K., P. W. Jordan, C. Rosé, D. Bhembe, M. Böttner, A. Gaydos, M. Makatchev, U. Pappuswamy, M. Ringenber, A. Roque, S. Siler, R. Srivastava, y R. Wilson. 2002. The architecture of Why2-Atlas: A coach for qualitative physics essay writing. En *Proc. of the 6th International Conference on Intelligent Tutoring Systems. LNCS 2363*, páginas 158–167.
-

- Vaquero, C., O. Saz, E. Lleida, J.M. Marcos, y C. Canalís. 2006. VOCALIZA: An application for computer-aided speech therapy in spanish language. En *Proc. IV Jornadas en Tecnología del Habla*, páginas 321–326, Zaragoza (España).
- Walker, M., I. Langkilde, J. Wright, A. Gorin, y D. Litman. 2000. Learning to predict problematic situations in a spoken dialogue system: Experiments with How May I Help You. En *Proc. of the North American Meeting of the Association for Computational Linguistics*, páginas 210–217, Seattle (Estados Unidos).
- Walker, M.A., D.J. Litman, C.A. Kamm, y A. Abella. 1998. Evaluating spoken dialogue agents with PARADISE: Two case studies. En *Computer Speech and Language*, volumen 12, páginas 317–347.
- Walker, Marilyn, Donald Hindle, Jeanne Fromer, Giuseppe Di Fabbrizio, y Craig Mestel. 1997. Evaluating Competing Agent Strategies for a Voice Email Agent. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'97)*, páginas 2219–2222, Rodas (Grecia).
- Ward, W. 1991. Evaluation of the CMU ATIS System. En *Proc. DARPA Speech and Natural Language Workshop*, páginas 101–105, Pacific Grove (Estados Unidos).
- Ward, W. y S. Issar. 1994. Recent improvements in the CMU spoken language understanding system. En *Proc. of the ARPA Human Language Technology Workshop*, páginas 213–216, Plainsboro (Estados Unidos).
- Ward, W. y B. Pellom. 1999. The CU Communicator system. En *Proc. of IEEE Automatic Speech Recognition and Understanding Workshop (ASRU'99)*, páginas 341–344, Keystone, Colorado (Estados Unidos).
- Watkins, C. 1989. *Learning from Delayed Rewards*. Ph.D. tesis, Cambridge University, Cambridge (Reino Unido).
- Weng, Fuliang, Sebastian Varges, Badri Raghunathan, Florin Ratiu, Heather Pon-Barry, Brian Lathrop, Qi Zhang, Tobias Scheideck, Harry Bratt, Kui Xu, Matthew Purver, Rohit Mishra, Madhuri Raya, Stanley Peters, Yao Meng, Lawrence Cavedon, y Liz Shriberg. 2006. CHAT: A Conversational Helper for Automotive Tasks. En *Proc. of the 9th International Conference on Spoken Language Processing (Interspeech/ICSLP)*, páginas 1061–1064, Pittsburgh (Estados Unidos).
- Wessel, F., K. Macherey, y R. Schlüder. 1998. Using word probabilities as confidence measures. En *Proc. of International Conference on Acoustics, Speech, and Signal Processing (ICASSP'98)*, volumen 1, páginas 225–228, Seattle, Washington (Estados Unidos).
- Williams, J., P. Poupart, y S. Young. 2006. Partially Observable Markov Decision Processes with Continuous Observations for Dialogue Management. En *Recent Trends in Discourse and Dialogue*. Eds L. Dybkjaer and W. Minker, Springer, páginas 191–217.
-

- Williams, J. y S. Young. 2006a. Scaling POMDPs for dialog management with composite summary point-based value iteration (CSPBVI). En *Proc. of AAAI Workshop on Statistical and Empirical Approaches for Spoken Dialogue Systems*, páginas 37–42, Boston (Estados Unidos).
- Williams, J. y S. Young. 2006b. Scaling POMDPs for dialog management with composite summary point-based value iteration (CSPBVI). En *Proc. of AAAI Workshop on Statistical and Empirical Approaches for Spoken Dialogue Systems*, páginas 37–42, Boston (Estados Unidos).
- Williams, J. y S. Young. 2007a. Partially Observable Markov Decision Processes for Spoken Dialog Systems. En *Computer Speech and Language*, volumen 21(2), páginas 393–422.
- Williams, J. y S. Young. 2007b. Scaling POMDPs for Spoken Dialog Management. En *IEEE Audio, Speech and Language Processing*, volumen September 2007.
- Williams, Sandra. 1996. Dialogue management in a mixed-initiative, cooperative, spoken language system. En *11th Twente Workshop on Language Technology (TWLT11)*, páginas 199–208, Enschede (Países Bajos).
- Witten, I.H. y T.C. Bell. 1991. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. En *IEEE Trans. Information Theory*, volumen 37, páginas 1085–1094.
- Yi, J., J. Glass, y L. Hetherington. 2000. A Flexible, Scalable Finite-State Transducer Architecture for Corpus-Based Concatenative Speech Synthesis. En *Proc. of International Conference on Spoken Language Processing (ICSLP'00)*, volumen 3, páginas 322–325, Beijing (China).
- Young, S. 2000. Probabilistic Methods in Spoken Dialogue Systems. En *Philosophical Trans Royal Society (Series A)*, volumen v358 i1769, páginas 1389–1402.
- Young, S. 2002a. Talking to machines (statistically speaking). En *Proc. of International Conference Spoken Language Processing (ICSLP'02)*, páginas 9–16, Denver (Estados Unidos).
- Young, S. 2002b. The Statistical Approach to the Design of Spoken Dialogue Systems. Informe técnico, CUED/F-INFENG/TR.433, Cambridge University Engineering Department, Cambridge (Reino Unido).
- Young, S., J. Williams, J. Schatzmann, M. Stuttle, y K. Weilhammer. 2005. The Hidden Information State Approach to Dialogue Management. Informe técnico, Department of Engineering. University of Cambridge, Cambridge (Reino Unido).
- Young, Steve. 2006. Using POMDPs for Dialog Management. En *Proc. of IEEE-ACL Workshop on Spoken Language Technology (SLT 2006)*, páginas 8–13, Palm Beach (Aruba).
-

- Young, Steve, Jost Schatzmann, Karl Weilhammer, y Hui Ye. 2007. The Hidden Information State Approach to Dialogue Management. En *Proc. of 32nd IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volumen 4, páginas 149–152, Honolulu, Haway (USA).
- Zamora, Francisco. 2005. Proyecto Fin de Carrera: Implementación eficiente del algoritmo de retropropagación del error con “momentum” para redes “hacia-delante” generales. Departamento de Sistemas Informáticos y Computación. Universitat Politècnica de València.
- Zhang, R. y A. Rudnicky. 2001. Word level confidence annotation using combinations of features. En *Proc. of European Conference on Speech Communications and Technology (Eurospeech'01)*, páginas 2105–2108, Aalborg (Dinamarca).
- Zue, V., S. Seneff, J. Glass, J. Polifroni, C. Pao, T. Hazen, y L. Hetherington. 2000. JUPITER: A telephone-based conversational interface for weather information. En *IEEE Transactions on Speech and Audio Processing*, volumen 8(1), páginas 85–96.
- Zue, V., S. Seneff, J. Polifroni, M. Phillips, C. Pao, D. Goddeau, J. Glass, y E. Brill. 1993. The MIT ATIS System: December 1993 Progress Report. En *Proc. of ARPA Spoken Language Technology Workshop*, páginas 339–353, Princeton (Estados Unidos).
- Zue, V., S. Seneff, J. Polifroni, M. Phillips, C. Pao, D. Goodine, D. Goddeau, y J. Glass. 1994. PEGASUS: A spoken dialogue interface for on-line air travel planning. En *Speech Communication*, volumen 15, páginas 331–340.
- Zukerman, I. y D. Litman. 2001. Natural language processing and user modeling: Synergies and limitations. En *User modeling and user-adapted interaction*, volumen 11, páginas 129–158.
-