Interficies Multimodales basadas en Avatares Virtuales

Memoria del proyecto de final de carrera
correspondiente a los estudios de
Ingeniería Superior en Informática
presentado por
Francisco Gutiérrez Romero
y dirigido por
Jordi Gonzàlez i Sabaté.

Bellaterra, 12 de Septiembre de 2011

El firmante, Jordi Gonzàlez i Sabaté, profesor titular del Departament de Ciències de la Computació de la Universitat Autònoma de Barcelona

CERTIFICA:
Que la presente memoria ha sido realizada bajo su dirección por Francisco Gutiérrez Romero

Bellaterra, 12 de Septiembre de 2011

Firmado:

Dedicado a las personas que...

Están dispuestas a trabajar en sí mismas antes de pretender transformar el mundo.

Han aprendido que si quieren disfrutar del Arco Iris deben estar dispuestas a soportar la lluvia.

Siguen conservando su capacidad de admiración, sorpresa y curiosidad.

Y dedicado especialmente a tí...

Vanessa, por haber estado a mi lado, haberme dado tu apoyo y regalado tu amor durante todo este tiempo.

"Cualquier tecnología lo suficientemente avanzada es
indistinguible de la magia"
ARTHUR C. CLARKE

"El verdadero problema no es si las máquinas piensan,
sino si los hombres lo hacen."
B.F. SKINNER

"La Ciencia es una sola luz,
e iluminar con ella cualquier parte
es iluminar con ella el mundo entero"
ISAAC ASIMOV

"No me siento obligado a creer que el mismo Dios
que nos ha dotado de sentido común, raciocinio e intelecto
tuviera como intención privarnos de su uso"
GALILEO GALILEI

Agradecimientos

Muchas personas me han ayudado, de una u otra forma, a desarrollar este proyecto, y a ellas debo dar las gracias. En especial, a mi jefe de proyecto Jordi Gonzàlez i Sabaté, por haber creído en mi capacidad para lograrlo desde un inicio y por haber sabido sacar lo mejor de mí. También quiero agradecer a Marc Castelló y a Carles Fernández Tena su colaboración y ayuda. Mi más sentido agradecimiento a todos aquellos compañeros con los que he compartido asignaturas y prácticas interminables, sobre todo, a aquellos que estuvieron empujándome y animándome en este último año, en el que realicé más de 100 créditos. Por último, agradezco la confianza que en mí depositó mi madre cuando decidí realizar esta carrera y en el transcurso de ésta. A todos vosotros, muchas gracias.

Índice general

1. Introducción **1**

1.1. Objetivos 6

1.2. Planificación 8

1.3. Organización de la memoria 8

2. Estado del arte **13**

3. Base del proyecto **19**

3.1. Análisis y requerimientos 19

3.2. Xface 20

3.2.1. Introducción 20

3.2.2. Funcionamiento 22

3.3. Flite Text to Speech 24

3.3.1. Introducción 25

3.3.2. Estructura 26

3.4. Julius Speech to Text 28

3.5. NL Natural Language 30

3.5.1. NLG Natural Language Generation 31

3.5.2. NLU Natural Language Understanding 33

3.6. Base de Datos 35

4. Diseño y desarrollo **37**

4.1. Entorno gráfico 37

4.1.1. Background 38

4.1.2. Los mensajes 38
4.1.3. Sincronismo 40
4.1.4. Entorno web 40
4.2. Text to Speech 41
4.2.1. Diccionario 43
4.2.2. Emoticonos 43
4.3. Speech to Text 44
4.3.1. Gramática de reconocimiento 44
4.4. Interacción 47
4.4.1. Mensajes 47
4.4.2. Acciones 48
4.4.3. Consultas 50
4.5. Consciencia 52

5. Resultados **57**
5.1. Introducción 57
5.2. Especificaciones del software 57
5.3. Test 58

6. Conclusión y trabajo futuro **63**
6.1. Conclusión 63
6.2. Trabajo futuro 64
6.2.1. Otras vías de investigación 65

A. ¿Sueñan los Avatares Virtuales... ? **67**
A.1. Planteamiento 67
A.2. Consciencia 69
A.3. Modelo Computacional del Cerebro Consciente 69
A.4. Inteligencia Artificial Fuerte 71
A.5. Test de Turing y la Habitación China de Searle 74

B. Formatos de Xface **79**
B.1. Estándard FA de MPEG-4 79

B.2. Lista de comandas 80
B.3. Ejemplo de fichero FDP de Xface 82

Bibliografia **87**

Índice de figuras

1.1. Herramienta de desarrollo Xface Toolkit 3
1.2. Max Headrooom 6
1.3. Escenario 7
1.4. Diagrama de Gantt. 9
1.5. Detalle de las tareas realizadas. 10

2.1. Sistema de traducción Verbmobil 15
2.2. Consultas sobre video Bilvideo 16

3.1. Detalle de las tareas realizadas 20
3.2. Evolución del desarrollo de un Avatar con Xface Toolkit . . . 21
3.3. Distribución de los módulos de la librería central de Xface. . . 23
3.4. Esquema del tratamiento del Lenguaje Natural. 31
3.5. Esquema del módulo NLU 34
3.6. Tablas de la base de datos 36

4.1. Background 39
4.2. Entorno web 41
4.3. Text to Speech 42
4.4. Estados de ánimo del Avatar 45
4.5. Entidades de la escena. 48
4.6. Bucle general de la aplicación 49
4.7. Lector de nuevos mensajes. 50
4.8. Genera el texto de la nueva acción. 51
4.9. Lector de nuevas consultas. 53

4.10. Aprendizaje de sucesos. 55

5.1. Acciones en tiempo real 59
5.2. Lista de agentes que están en la entrada 59
5.3. Prioridad de las acciones 60
5.4. Lo más importante sucedido hoy 60

A.1. Esquema del conocimiento. 71
A.2. Test de Turing. 75
A.3. La habitación China de Searle. 76

Índice de cuadros

4.1. Fonemas en Flite 44
4.2. .grammar file 46
4.3. .voca file 46
4.4. Ejemplo de la Tabla ″Learn″ de la base de datos 55

B.1. Ejemplo de archivo de configuración de XFace FDP. 83
B.2. Definición de las FAPUs. 83
B.3. Especificación de los modelos 3D para cada *keyframe*. 84
B.4. Especificación de los modelos 3D básicos para formar la cabeza en posición de descanso. 84
B.5. Especificación de más modelos para los *keyframes*. 85
B.6. Función de deformación a aplicar con los pesos. Los vértices del modelo se dan indexados. 86

Capítulo 1

Introducción

Hoy en día la utilización y la integración de los ordenadores en nuestra vida cotidiana es una realidad. Los ordenadores son una herramienta fundamental en oficinas de trabajo, tanto en supermercados como en tiendas, en centros de educación de todo tipo y en otros servicios públicos, además de serlo también en nuestras propias casas. Sin embargo, la mayoría de usuarios que utilizan ordenadores son gente inexperta, por lo que necesitan una preparación previa, con el desembolso económico que eso representa. Este hecho provoca la necesidad de crear aplicaciones más sencillas para gente sin preparación y de esta forma evitar el tiempo y el dinero destinado a la preparación del usuario.

Por lo tanto, una solución es hacer que las máquinas se adapten a la comunicación de los hombres, en vez de adaptarnos nosotros a ellas. Este simple hecho es una de las razones de la utilización del NL (Lenguaje Natural). Actualmente, la investigación en lenguaje natural esta diversificada en muchos campos: extracción de información en múltiples lenguas, generación de resúmenes, búsqueda de respuestas, clasificación de textos, traducciones automáticas o reconocimiento del habla [1].

La investigación en lenguaje natural se lleva desarrollando desde la déca-

da de los cincuenta, donde surgió por motivos militares con miras al desciframiento de mensajes captados a los ejércitos enemigos, centrándose en el problema de la traducción automática. En un principio, se creyó que el desarrollo de aplicaciones en lenguaje natural se lograría en poco tiempo, pero progresivamente comenzaron a surgir multitud de incógnitas. Este hecho hizo que muchos proyectos se quedaran sin financiación. Pocos siguieron con la investigación, pero en los últimos diez o quince años se ha vuelto a despertar un gran interés en esta tecnología, que muchos no han dudado en calificar como ″los cimientos de la sociedad de la información del futuro″ [8] [9].

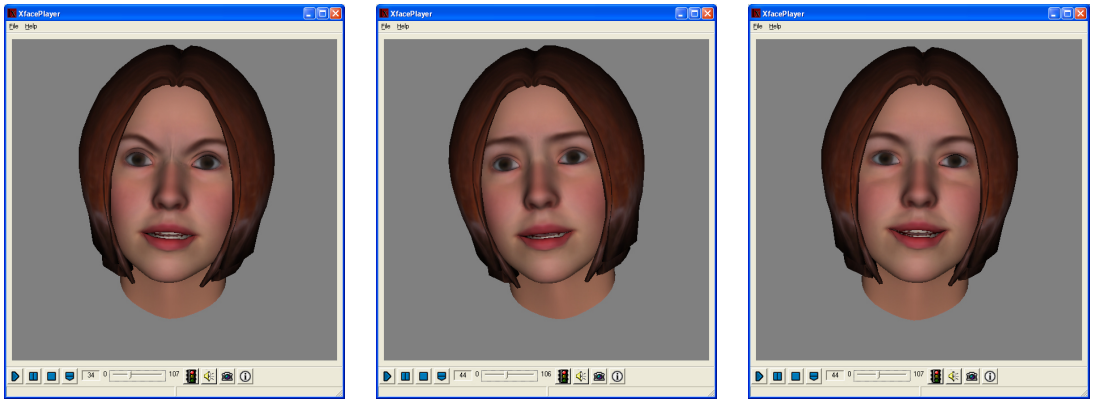
Así pues, desde la perspectiva de la Inteligencia Artificial, se puede decir que la investigación en el campo del lenguaje natural se ha centrado en dos objetivos [7]:

1- Facilitar la comunicación entre hombre y máquina para usuarios no especializados.

2- Modelar los procesos cognitivos que entran en juego en la comprensión del lenguaje para diseñar sistemas que realicen tareas lingüísticas complejas.

Por otro lado, el uso de cabezas 3D virtuales está ganando popularidad en sectores como la industria del entretenimiento, videojuegos, servicio al cliente, interacción humana con la computadora o realidad virtual, pero debido a la escasez de herramientas que facilite el desarrollo de éstas, su evolución es escasa y poco reutilizable.

Los emoticonos son considerados como un *Standard de facto* a la hora de expresar emociones a través de Internet. Un conjunto de carácteres que vistos de lado parecen una cara sonriente o una enfadada, son utilizados por todos para mostrar estados de ánimo en chats y mensajería instantánea.
Incluso se ha ido más allá substituyendo los conjuntos de carácteres que for-



Enfadado Triste Contento

Figura 1.1: La división de Tecnologías Cognitivas y de la Comunicación (Cognitive and Communication Technologies, TCC) del ITC-irst (Instituto Trentino di Cultura – Il centro per la Recerca Scientifica e Tecnologica) ha desarrollado un conjunto de librerías y herramientas *open source* e independiente de plataforma bajo el nombre de *Xface ToolKit*.

man un emoticono por una imagen de una cara que exprese lo mismo, más clara y con más posibilidades. Por ello, el uso de emoticonos puede ser muy útil en vistas a mejorar la transmisión de emociones a través de cabezas en 3D. Podemos observar en la figura 1.1 algunos de los estados de ánimo que puede presentar un Avatar.

Actualmente, existen diversas situaciones donde un computador puede colaborar en tareas de comunicación humanas de manera multimodal, es decir, allí donde tiene lugar una interacción conjunta a más de un nivel (auditivo, visual, táctil y/o gestual) entre el hombre y la máquina [2].

Es necesario clasificar estas situaciones en tres tipos básicos, según la función del ordenador [3]:

1. El ordenador actúa como mensajero, es decir, permite la comunicación a larga distancia entre personas que trabajan juntas o que están colaborando en un objetivo en común. En esta situación, los documentos que se inter-

cambian o manipulan deberían ser multimedia para poder mantener toda la información.

2. La máquina anima una realidad virtual que extiende las capacidades creativas y expresivas humanas. El usuario se ve immerso en un mundo con el que interactúa, normalmente a nivel gestual.

3. El ordenador actúa como un compañero, es decir, colaborando con el usuario en una cierta tarea, sirviéndose del diálogo para entender los objetivos inclusive en las intenciones del usuario. Esto puede incrementar la eficiencia de la sesión de trabajo.

En todos estos casos, la interacción entre humanos y máquinas se puede concebir a nivel multisensorial. Estas interacciones se pueden definir como multimodales si cumplen las dos condiciones:

Primero, la interacción se debería enfocar en diversas modalidades (o modos) motores y sensoriales humanas, como pueden ser la visión, la voz, tanto hablada como escuchada y la gesticulación (moverse, señalar, escribir, dibujar..) de manera simultánea y cooperativa. Segundo, la máquina ha de entender e interpretar la información que le llega de diversos dispositivos de entrada y salida (llamados media).

Para el diseñador de este tipo de interficies, las aplicaciones descritas anteriormente muestran ciertas características comunes, como por ejemplo la transmisión de información, pero varían en el manejo y la interpretación de la información de entrada y salida. Por lo tanto, mientras que, por ejemplo, en la realidad virtual se da más importancia al comportamiento del usuario por encima del diálogo con la máquina, el cual se ve reducida a un simple esquema de acción-reacción, sin embargo, en la colaboración de tareas pasa totalmente al contrario:

En este caso, la relación entre usuario y sistema (o, mejor dicho, la relación entre operador y tarea) es más importante, ya que aquí el usuario está solo interactuando con la máquina.

Por lo tanto, las capacidades de comunicación de ésta se deberían modelar en base a la comunicación humana, con el fin de aumentar la eficiencia y la fiabilidad en la ejecución de la tarea.

El trabajo que se presenta en esta memoria, forma parte del proyecto europeo HERMES [4] que tiene como objetivo global ofrecer un Agente Conversacional Personificado (Embodied Conversational Agents, ECA) capaz de interactuar como una persona a nivel auditivo, gestual y emotivo. Es decir, capaz de oir y hablar, de moverse, gesticular, entender emociones y expresar de una forma realista en respuesta a ello. Por lo tanto, se trata de una interficie multimodal de tercer tipo.

Continuando con el proyecto HERMES, éste, se divide en otros subproyectos, donde los datos de salida se separan y son tratados independientemente. Más concretamente, la idea inicial del proyecto final de carrera, inicialmente propuesto, se centra en la cabeza y la cara del agente, Max Headroom.

Partiendo de las herramientas de software libre XFace [5], el cual permite crear cabezas parlantes (Talking Heads, o también llamados Avatares) en inglés e italiano respectivamente, y expresar emociones.

El objetivo es hacer que cualquier modelo de cabeza 3D pueda hablarnos de aquello que ocurre en un escenario al mismo tiempo de mostrarnos emociones, tal y como hacía el Avatar simulado Max Headroom hace 20 años en la televisión que podemos observar en la figura 1.2.

Figura 1.2: Max Headroom es el nombre del protagonista de un prototipo de Inteligencia Artificial. Fue una serie de televisión presentada en 1987 en Estados Unidos de características innovadoras y estilo futurista, con influencias del movimientos cyberpunk.

1.1. Objetivos

El objetivo principal de este proyecto, es el de desarrollar una aplicación capaz de interactuar con un sistema de visión instalado en un determinado lugar. Este sistema se encarga de observar todas aquellas acciones producidas por agentes en una escena. Una vez definida la acción, quien la realiza y donde, se generará un predicado que se insertará en una base de datos.
Las aplicaciones a desarrollar han de ser capaces de recoger este predicado, pasarlo a lenguaje natural y comprender aquellas partes de éste más relevantes, como el tiempo verbal o el sujeto de la acción.

Una vez generada dicha frase, un Avatar virtual 3D deberá reproducirla mediante voz y realizar los movimientos propios del habla para cada una de las palabras que componen la frase. Además, debe ser capaz de comprender qué se le está pidiendo y responder lo más acertadamente posible a consultas tanto escritas como habladas.

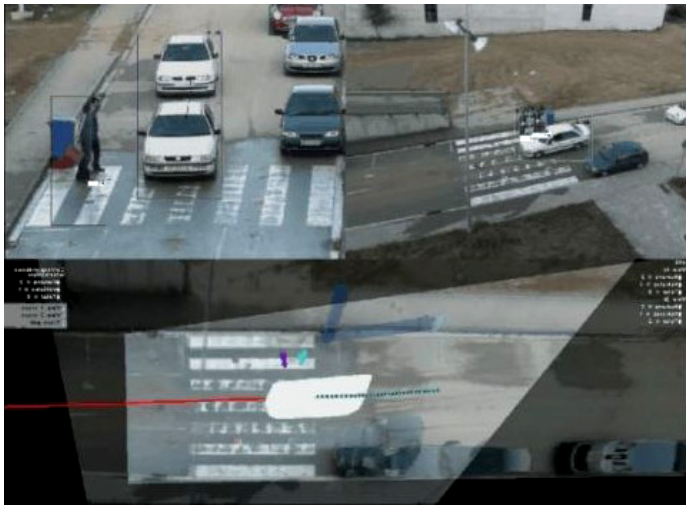


Figura 1.3: Escenario

Por lo tanto, no solo se limitará a decirnos aquello que sucede en la zona de visión, sino que también ha de ser capaz de responder a una serie de consultas sobre agentes, acciones o lugares.

Por último, deberá preveer qué sucesos podrán ocurrir en un momento dado, teniendo en cuenta las veces que estos han ido sucediendo habitualmente. Por lo que se le estará dotando de un cierto nivel de aprendizaje sobre los acontecimientos.

La finalidad de este proyecto es la de facilitar la tarea de búsqueda de sucesos ocurridos en unas ciertas circunstancias. Simplemente, con realizar la consulta, ¿Qué es lo más importante que ha sucedido hoy?, será suficiente para que el sistema nos reponda sobre dicha consulta y de esta forma ahorrar tiempo y esfuerzo.

El desarrollo de esta aplicación se basa en el diseño e implementación de un módulo capaz de generar lenguaje natural y otro módulo encargado de la compresión del lenguaje generado. También se ha de realizar la adaptación a nuestras necesidades de diferentes programas ya existentes, tales como Text

to Speech y Speech to Text. Esto es así, debido a que serán necesarias ciertas características que actualmente no poseen. Y por último, implementar y diseñar un entorno visual, donde una Avatar 3D irá comentando todo aquello que sucede en el exterior o simplemente contestando a nuestras peticiones.

1.2. Planificación

En la figura 1.4 se visualiza la planificación de tareas realizadas durante el proyecto.

En la figura 1.5 se puede ver la duración de las tareas durante el proyecto. La estimación de horas semanales es de 27 por semana para poder realizar la lista de tareas.

1.3. Organización de la memoria

La estructura de la memoria es la siguiente:

Capítulo 1. **Introducción:** Se explican las bases y dificultades de este proyecto, así como los principales objetivos para su utilización.

Capítulo 2. **Estado del arte:** Se hace mención del proyecto del cual forma parte este PFC y se describe otros trabajos similares dentro del campo estudiado.

Capítulo 3. **Base del proyecto:** En este capítulo se hace un estudio de los requisitos del proyecto, así como una breve descripción de las tecnologías utilizadas y de los módulos que se han creado o integrado en el sistema.

Capítulo 4. **Desarrollo e implementación:** Se enumaran las decisiones de diseño tomadas, los métodos utilizados en la realización de los módulos y los cambios para la correcta integración de todas las herramientas.

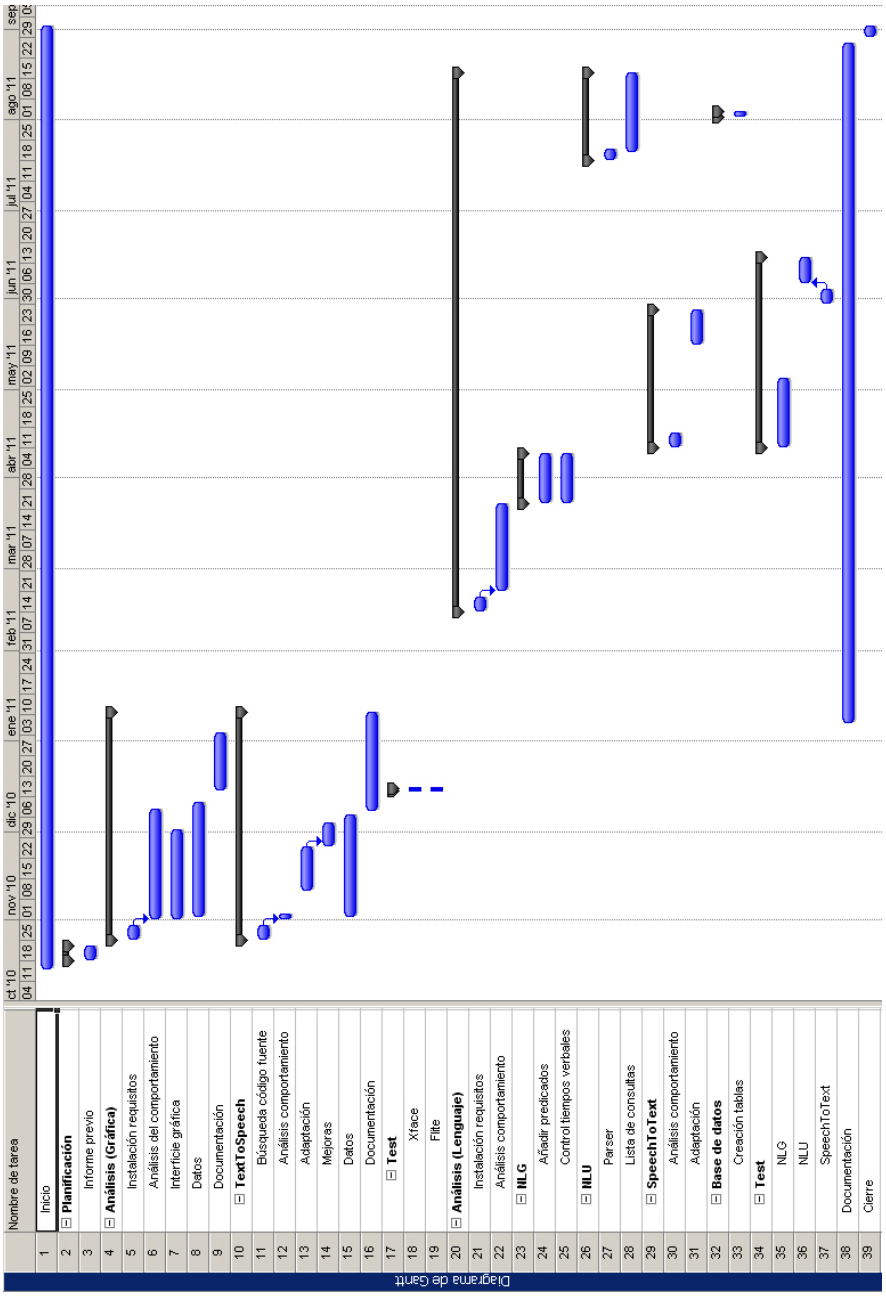


Figura 1.4: Diagrama de Gantt.

Capítulo 5. **Resultados:** Se analizan y se muestran los resultados obtenidos a partir del conjunto de pruebas realizadas.

Diagrama de Gantt

	Nombre de tarea	Comienzo	Fin		Duración
1	Inicio	vie 15/10/10	jue 01/09/11		230 días?
2	**Planificación**	**lun 18/10/10**	**vie 22/10/10**		**5 días?**
3	Informe previo	lun 18/10/10	vie 22/10/10		5 días?
4	**Análisis (Gráfica)**	**lun 25/10/10**	**lun 10/01/11**		**56 días?**
5	Instalación requisitos	lun 25/10/10	vie 29/10/10		5 días?
6	Análisis del comportamiento	lun 01/11/10	mié 08/12/10		28 días?
7	Interficie gráfica	lun 01/11/10	mié 01/12/10		23 días?
8	Datos	mar 02/11/10	vie 10/12/10		29 días?
9	Documentación	mié 15/12/10	lun 03/01/11		14 días?
10	**TextToSpeech**	**lun 25/10/10**	**lun 10/01/11**		**56 días?**
11	Búsqueda código fuente	lun 25/10/10	vie 29/10/10		5 días?
12	Análisis comportamiento	lun 01/11/10	mar 02/11/10		2 días?
13	Adaptación	jue 11/11/10	jue 25/11/10		11 días?
14	Mejoras	vie 26/11/10	vie 03/12/10		6 días?
15	Datos	mar 02/11/10	lun 06/12/10		25 días?
16	Documentación	mié 08/12/10	lun 10/01/11		24 días?
17	**Test**	**mié 15/12/10**	**mié 15/12/10**		**1 día?**
18	Xface	mié 15/12/10	mié 15/12/10		1 día?
19	Flite	mié 15/12/10	mié 15/12/10		1 día?
20	**Análisis (Lenguaje)**	**lun 14/02/11**	**mar 16/08/11**		**132 días?**
21	Instalación requisitos	lun 14/02/11	vie 18/02/11		5 días?
22	Análisis comportamiento	lun 21/02/11	mar 22/03/11		22 días?
23	**NLG**	**mié 23/03/11**	**vie 08/04/11**		**13 días?**
24	Añadir predicados	mié 23/03/11	vie 08/04/11		13 días?
25	Control tiempos verbales	mié 23/03/11	vie 08/04/11		13 días?
26	**NLU**	**lun 18/07/11**	**mar 16/08/11**		**22 días?**
27	Parser	lun 18/07/11	jue 21/07/11		4 días?
28	Lista de consultas	jue 21/07/11	mar 16/08/11		19 días?
29	**SpeechToText**	**lun 11/04/11**	**vie 27/05/11**		**35 días?**
30	Análisis comportamiento	lun 11/04/11	vie 15/04/11		5 días?
31	Adaptación	lun 16/05/11	vie 27/05/11		10 días?
32	**Base de datos**	**mar 02/08/11**	**mié 03/08/11**		**2 días?**
33	Creación tablas	mar 02/08/11	mié 03/08/11		2 días?
34	**Test**	**lun 11/04/11**	**mar 14/06/11**		**47 días?**
35	NLG	lun 11/04/11	mié 04/05/11		18 días?
36	NLU	lun 06/06/11	mar 14/06/11		7 días?
37	SpeechToText	lun 30/05/11	vie 03/06/11		5 días?
38	Documentación	vie 07/01/11	vie 26/08/11		166 días?
39	Cierre	lun 29/08/11	jue 01/09/11		4 días?

Figura 1.5: Detalle de las tareas realizadas.

Capítulo 6. **Conclusión y trabajo futuro:** Se resume el trabajo realizado y los objetivos obtenidos. También, se explica aquellas posibles ampliaciones o mejoras que se podrían realizar en un futuro sobre este proyecto.

Apéndices. Este capítulo se divide en dos partes bien diferenciadas. Por un lado, desde un punto de vista más filosófico, se hace un planteamiento directo sobre que es realmente la consciencia y si sería posible, mediante la ingeniería, construir algún sistema electrónico dotado con ésta. Y por otro

lado, en el siguiente apéndice, se trata más a fondo el contenido de los ficheros que representan al Avatar y se adjunta la lista de órdenes que éste es capaz de comprender y realizar.

Capítulo 2

Estado del arte

La investigación actual propone diferentes alternativas en el área del lenguaje natural. La mayoría de ellas destacan la ventaja del uso de las ontologías. Las aplicaciones de las bases de datos léxico-semánticas son variadas y facilitan operaciones tales como la sinonímia, medida de distancia entre palabras o desambiguación semántica. Las ontologías son la organización de un conjunto clasificado de conceptos pertenecientes a un dominio y las relaciones que pueden formarse entre ellos. Éstas se utilizan principalmente para compartir, reutilizar o intercambiar el conocimiento que envuelve un cierto dominio de interés y permiten reducir la ambiguedad semántica entre otras ventajas.

Erozel y Cicekli[6], presentan una propuesta de interficies de búsquedas en NL sobre bases de datos de video. Utiliza un *parser* que funciona como un diccionario de palabras propio utilizando una base de datos léxico-semántico *WordNet* y métodos para medir la distancia semántica entre palabras, cuando no existe una coincidencia exacta entre las palabras de la consulta y la terminología de la base de datos.

El documento también destaca las dificultades de implementar interficies de búsqueda en lenguaje natural, ya que todavía son el medio más flexible para expresar consultas, y éstas están muy limitadas por el dominio y por la

capaciadad de los parsers.

En diferencia a las aplicaciones que utilizan lenguaje natural, existe un gran número de herramientas para la anotación de video, como puede ser VideoAnnEx Annotation Tool, VARS, Anvil, Elan, Frameline 47 Video Notation, ViPER-GT, Vannotea, etc. Muchas de éstas permiten especificar regiones de interés, descripciones de situaciones, anotaciones personales del usuario o incluso anotación de sonido. A continuación se analizarán las aplicaciones basadas en lenguaje natural.

ELF *English Language Front End* Sistema para generar preguntas en lenguaje natural y poder extraer la información sin necesidad de tener conocimientos del lenguaje SQL. Es una aplicación creada por *ELF Software Co. en 1999*, y es compatible con cualquier base de datos [10].

Semantra Aplicación enfocada al mundo empresarial. Es un software que funciona mediante lenguaje natural, con el propósito de que cualquier empleado de una empresa pueda utilizarlo sin necesidad de saber tecnicismos o que deba pasar por un entrenamiento previo, ahorrando dinero a la empresa. Por lo tanto, gracias a esta aplicación, cualquier usuario que necesite saber datos concretos sobre temas empresariales, puede obtenerlos realizando la pregunta adecuada, como si estuviera haciéndosela a otra persona. Semantra utiliza un motor de semántica para analizar el lenguaje natural, una ontología *OntoloNet* que concentra una gran variedad de conceptos utilizados en el mundo empresarial (dominio restringido) y por último, transforma la pregunta en una consulta SQL gracias a un generador de consultas dinámico. El resultado es mostrado en gráficas, tablas, etc [11].

Verbmobil *Verbmobil* es un sistema de traducción de voz a voz entre interlocutores de teléfono que permite la traducción bidireccional de tres idiomas (Alemán, Inglés y Japonés), donde las conversaciones deben ser restringidas a temas de negocios [12] [13]. En la figura 2.1 se visualiza el funcio-

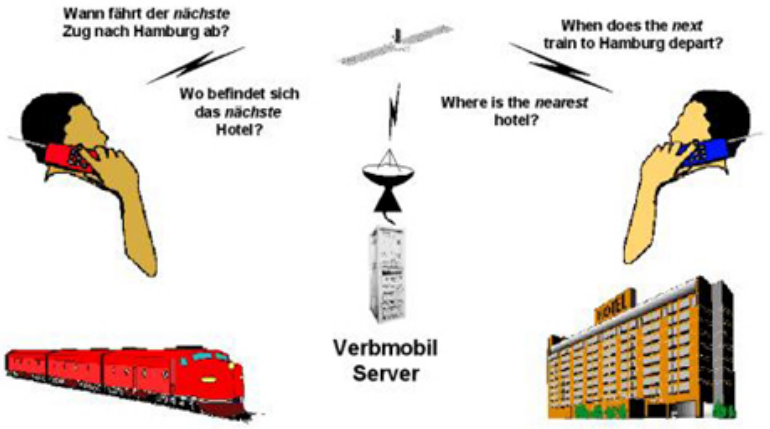


Figura 2.1: Proyecto alemán, en el que participaron instituciones públicas y privadas, cuyo proceso duró ocho años (1993-2000)

namiento de este sistema.

Trains Proyecto desarrollado por la Universidad de Rochester (EE.UU) por James Allen. Su objetivo es vender boletos de tren vía telefónica. El programa analiza la pregunta que realiza el cliente y la transforma en una consulta SQL, para ejecutar la búsqueda en su base de datos. Por lo tanto, el programa mantiene una conversación con el cliente, explicándole los precios y condiciones, escuchando preguntas y comentarios sobre qué boleto es más conveniente. Finalmente, cuando llegan a un acuerdo, el programa reserva el boleto [14] [15].

BilVideo Sistema creado por Bilkent University Multimedia Database Group *BILMDG* para realizar consultas sobre videos, con la posibilidad de realizarlas mediante lenguaje natural. Para realizar este tipo de consultas, se utiliza un amplio conjunto de reglas realizadas en Prolog. En estas reglas se pueden añadir combinaciones de elementos que especifiquen dirección, topología, relaciones 3D o trayectorias de objetos. En la figura 2.2 se visualiza el

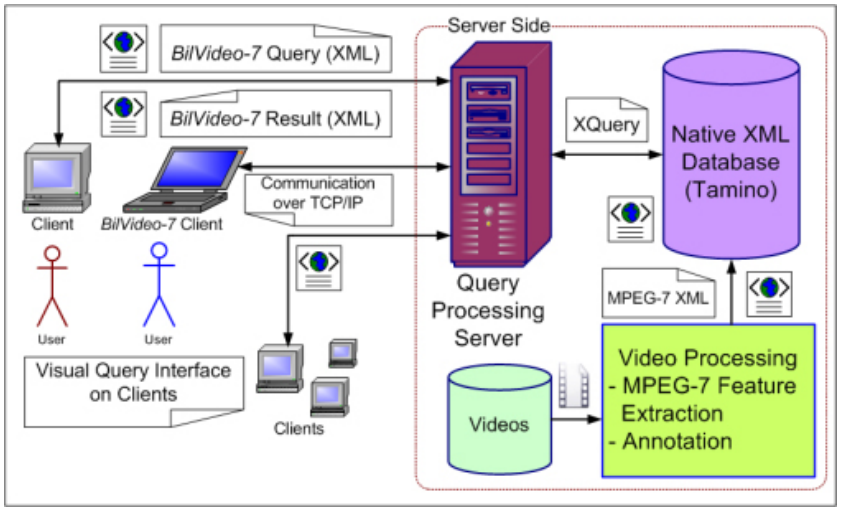


Figura 2.2: La característica principal de este proyecto es que proporciona un soporte espacio-temporal (combina consultas de tiempo, espacio, apariencia de objetos y trayectorias) y semántica para consultas de vídeos.

mecanismo de funcionamiento de esta aplicación.
Para la creación de consultas, la aplicación incorpora toda una serie de herramientas para hacer más fácil su desarrollo. Además de herramientas para la creación de consultas, también necesita herramientas para la extracción de objetos de videos y para la anotación de estos.

Una vez definidas las diferentes aplicaciones, se encuentran varias similitudes con la aplicación creada, como puede ser la generación dinámica de consultas SQL del sistema Semantra, o la necesidad de interactuar con una base de datos para la extracción de información. El sistema más parecido al que hemos desarrollado, es el proyecto *BilVideo*. En este se pueden destacar detalles como, por ejemplo, el trabajo realizado para poder especificar trayectorias, situaciones en el espacio, direcciones de objetos, etc.

También destaca la anotación de videos, pudiendo realizar una etiquetación muy detallada, especificando los objetos que aparecen, los tiempos que dura cada acción, etc., aunque se puede hacer algo complicado, al ha-

ber tanta información a rellenar y relacionar entre los elementos etiquetados, en comparación con nuestra herramienta de anotación que es más simple e intuitiva. Una de las desventajas de este proyecto, es la forma de realizar las consultas, ya que utiliza un sistema propio, utilizando Prolog. Si no se conoce bien como funciona, puede hacerse muy difícil crear una consulta propia, aunque se utilicen las herramientas que incorpora la aplicación. En este punto, en nuestro caso hemos acertado utilizando el lenguaje SQL, ya que es mucho más simple y hay multitud de herramientas que te permiten crear estas consultas fácilmente.

Otro punto a comentar, es la presentación de resultados. Mientras que BilVideo presenta una simple ventana con el número de casos que ocurren y en qué fotograma ocurren, en nuestra aplicación es un avatar 3D quien, mediante la voz, muestra toda la información solicitada o la que cree conveniente mostrar en cada caso.

En definitiva, el objetivo de estas aplicaciones anteriormente comentadas, es el de facilitar, en gran medida, el trabajo y el esfuerzo al ser humano en diferentes tareas en función de sus necesidades. Se ha podido comprobar que las aplicaciones basadas en el tratamiento del lenguaje natural son muy necesarias y de gran utilidad para resolver problemas en ciertas situaciones cotidianas. Por lo tanto, la comunicación es y seguirá siendo el medio para relacionarnos con los demás más esencial de nuestras vidas, ya sea entre humanos o entre máquinas y humanos.

Capítulo 3

Base del proyecto

3.1. Análisis y requerimientos

Las necesidades de este proyecto son las de poder comunicarnos con una cabeza parlante 3D, ya sea de forma escrita o hablada. La comunicación entre el Avatar y nosotros, se basará en que éste ha de ir comentando aquello que sucede en el exterior y además, ser capaz de responder a preguntas sobre ciertos temas relacionados con lo ocurrido.

Para ello, necesitaremos diseñar una aplicación que pueda tomar toda la información de un fichero FDP, ver apéndice B, y que pueda reproducir mensajes hablados. Además, deberá mostrar un estado de ánimo con cada mensaje, en función del contenido de éste. A medida que vaya observando los acontecimientos ocurridos, deberá poder hacer una reflexión sobre aquellos sucesos con la mayor probabilidad a que vuelvan a repetirse en un momento dado.

Para poder realizar este diseño necesitaremos crear una aplicación que acepte ficheros FDP, también se necesitará un *speech to text* para poder hacer preguntas mediante la voz, un *text to speech* para convertir el texto de los mensajes a sonido, una lista de consultas que el Avatar pueda comprender

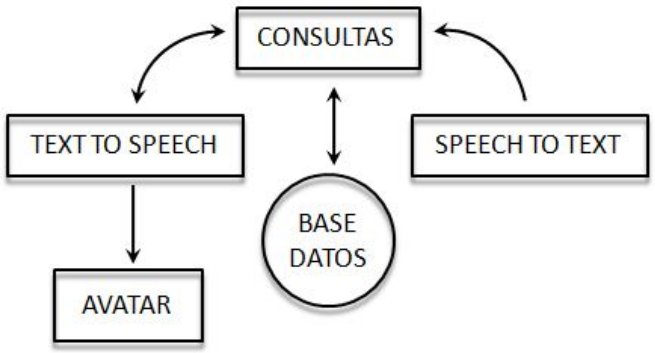


Figura 3.1: Detalle de las tareas realizadas

y tablas en una base de datos donde almacenar y mover toda la información de sucesos, actores, objetos y lugares.

3.2. Xface

En primer lugar, se estudia el software elegido como punto de partida de este proyecto, Xface, con el fin de entenderlo bien y saber donde trabajar para poder integrar al Avatar en él. En este apartado se procederá a su descripción.

3.2.1. Introducción

Desarrollado por la división de Tecnologías Cognitivas y de la Comunicación (Cognitive and Communication Technologies, TCC) [16] del ITC-irst (Instituto Trentino di Cultura, Il centro per la Recerca Scientifica e Tecnologica) [5], XFace es un kit de herramientas *open source* que pretende cubrir la falta de éstas respecto a los llamados Agentes Conversacionales Personificados (Embodied Conversational Agents, ECA). Concretamente, quiere ser una herramienta que permita generar y animar fácilmente agentes parlantes

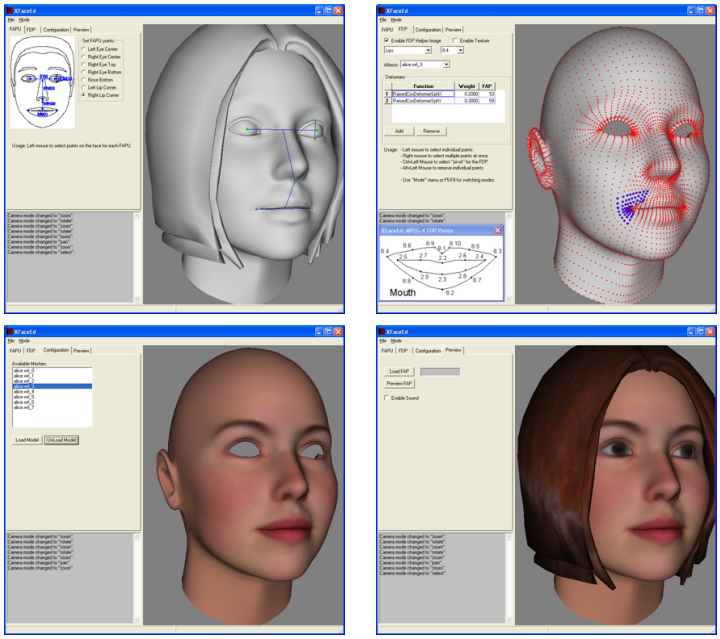


Figura 3.2: Alternativamente, también permite hacer animaciones basadas en *keyframes*, modelos 3D estáticos para gestos concretos de la cara previamente definidos, de los cuales XFace ya incluye algunos básicos, pero se pueden añadir más.

3D, también conocidos como Avatares, *caras* o *cabezas parlantes*.

Actualmente, en su versión 1.0 release del mes de enero de 2008, este kit está basado en el estándard FA (Facial Animation, animación facial) de MPEG-4, explicado en detalle en el apéndice B, de manera que pueda reproducir *streams* de FAP's (MPEG-4 Facial Animation Parameters, Parámetros de Animación Facial), los cuales determinan el grado de deformación y/o desplazamiento de zonas concretas de la cara, como por ejemplo el labio superior, el lado izquierdo del ojo derecho o el párpado inferior izquierdo.

Debido al amplio colectivo de investigación al que va dirigido, la arquitectura de XFace está diseñada para que sea configurable y fácil de extender. Todas las partes de éste son independientes del sistema operativo y pueden ser compiladas con cualquier compilador de ANSI C++ estándard (el código

viene preparado para ser compilado bajo Windows, en Visual C++ .Net). Para la animación de los objectos 3D en sí, el kit está basado en OpenGL y está bastante optimitzado para conseguir tasas de *frames* por segundo satisfactorias (para la generación de FAPs se requieren como mínim 25 FPS) con modelos con un número elevado de polígonos (12.000 polígonos) en un hardware modesto. En la figura 3.2 se pueden observar la evolución del modelado de una cabeza 3D.

Concretamente, hoy en dia el proyecto XFace consta de 5 programas: 2 librerías, la librería central, independente del resto XFace y una librería que interpreta el lenguaje de markup SMIL-Agent (XSmilAgent); y 3 aplicaciones finales que las hacen servir: un editor para configurar las cabezas para que XFace pueda trabajar y probarlos XFaceEd, un reproductor de animaciones faciales FAPs de ejemplo XFacePlayer y un cliente para trabajar con el anterior remotamente XFaceClient.

3.2.2. Funcionamiento

En este apartado se explica en detalle los 5 elementos que componen XFace, cual es la función de cada uno de ellos y como interactúan entre sí [17] [18].

Xface Librería central

La librería central de XFace se encarga de cargar los modelos de cara y la información MPEG-4 corresponente, recibir los *streams* de FAP's, normalmente leídos de un archivo de texto de tipo ″.fap″ que contiene una lista de diferentes valores de éstos y calcular las deformaciones y renderizaciones correspondientes sobre el modelo para crear la animación facial y reproduir sincronizadamente, el archivo de audio WAV con la voz, en el caso de que se proporcione. Está dividida en cuatro módulos o *namespaces* como podemos observar en la figura 3.3:

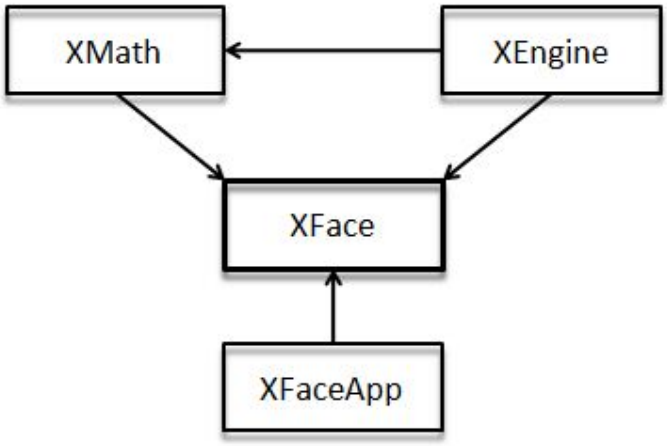


Figura 3.3: Distribución de los módulos de la librería central de Xface.

XEngine: Clase del motor gráfico 3D que utiliza XFace. Es una implementación simple, basada en OpenGL, especializada para XFace e implementada lo más genéricamente posible.
XMath: Clase para cálculos matemáticos complejos, como las funciones de deformación.
XFaceApp: Clase para interactuar con otras aplicaciones, como funciones para XML, interficies para el sonido, por temporizadores, etc.
XFace: Resto de clases de la librería que tratan la cara parlante en sí.

Esta librería necesita, como entrada básica para cargar un Avatar, un archivo de configuración XML llamado FDP, el cual se define para cada cabeza que se quiera utilizar en XFace y que describen los modelos 3D que se han de utilizar para definir cada parte de la cabeza, los FPs (Feature Points, Puntos Característicos), que sirven para adaptar un modelo 3D de una cabeza al estándard de MPEG-4, y sus zonas de influencia: conjunto de vértices alrededor del FP sobre los cuales se aplicará el cálculo de la función de deformación.

También da las 5 FAPUs (Facial Animation Parameter Units, Unidades del Parámetro de Animación Facial), que sirven para calibrar los FAPs genéri-

cos que llegan sobre el modelo en concreto al que se aplican y la función de deformación que se aplicará a cada zona, la cual ha de estar definida dentro del código de la librería, entre otras. También necesita, obviamente, los modelos estáticos 3D requeridos para el archivo FDP. Respecto a la función de deformación, en la implementación actual, ésta es por defecto una función coseno aumentada (raised cosinus), basada en la distancia euclídea que un punto se debería desplazar dentro de la región del FP en la dirección dada por el FAP corresponente. Si bien esta función da resultados satisfactorios, la librería permite que se pueda extender fácilmente con el fin de incluir y utilizar estrategias de deformación diferentes, como pueden ser *Radial Basis Functions* [18] o *Free Form Deformations* [19].

También, como ya se ha dicho previamente, se pueden definir y utilizar *keyframes* (otros modelos estáticos 3D de la misma cabeza mostrando gestos diferentes) que representen ciertos visemes (posición de la cara en general y de la boca en particular al pronunciar un fonema) y ciertas emociones concretas, como método alternativo de animación, los cuales también vendrán indicados en el archivo FDP. Esto permite crear un Avatar utilizando simplemente la combinación de los *keyframes* en tiempo real mediante alguna técnica de interpolación, en lugar de los FAP's de MPEG-4. En este caso, la entrada que se necesita es, además del archivo de audio ".wav", el llamado archivo de fonemas de tipo ″.pho″ y el archivo de animaciones de tipo ″.anim″.

En la siguiente sección se explica con mucho más detalle el funcionamiento y la generación de este tipo de ficheros.

3.3. Flite Text to Speech

A continuación, se explica el tipo de aplicación utilizada en el proyecto para el tratamiento de textos con la finalidad de convertirlos a voz.

3.3.1. Introducción

Flite fue desarrollado, principalmente, para hacer frente a una de las quejas más comunes acerca del sitema *Festival* de síntesis de voz, debido a que éste es grande y lento. *Festival* está escrito en C++ . Algunos compiladores de C++ son muy diferentes entre sí, lo que conlleva una gran cantidad de trabajo para garantizar la compatibilidad de la base del código a través de múltiples compiladores.

Otro problema con C++ es el tamaño y la eficiencia del código generado. *Festival* no es seguro para los subprocesos, y aunque se ejecuta bajo Windows, el modo servidor que está basado en Unix se centra en la rapidez de copia por escritura de memoria compartida para los clientes de servicios. Esta solución es perfectamente segura y práctica para los sistemas Unix, pero en Windows, donde la característica más común es utilizar el servicio para varios eventos, es caro y un programa inseguro para subprocesos, con lo que no puede ser eficaz.

Después de tomar en consideración los problemas citados anteriormente, se decidió desarrollar *Flite* como un nuevo sistema escrito en ANSI C. C es mucho más portátil que C++ , y ofrece más control sobre el tamaño de los objetos y la estructura de datos que utiliza. *Flite* no pretende ser una plataforma de desarrollo de investigación y de síntesis de voz, *Festival* es y seguirá siendo la mejor plataforma para ello. *Flite* va dirigido a dos tipos de comunidades bien diferenciadas. En la primera comunidad, *Flite* se puede utilizar como motor para pequeños dispositivos como PDAs y teléfonos con memoria y en algunos casos, ni siquiera en dispositivos con un sistema operativo convencional. *Flite* ofrece un motor amigable en tiempo de ejecución. Actualmente, se intenta desarrollar la construcción de nuevas voces en *Festvox* y depurar y ajustar *Festival*.

En la segunda comunidad, *Flite*, se dirige a la síntesis de los servidores para muchos clientes y a pesar de poseer grandes bases de datos fijos, son

aceptables tanto el tamaño de memoria requerida como la velocidad de sintetización.

Sin embargo, a pesar de la decisión de construir un motor de nueva síntesis, vemos esto estrechamente unido a las herramientas existentes de software libre de la síntesis de construcción de voz en *Festival* y *FestVox*.

Los desarrolladores, además, son partidarios de la tecnología de voz dentro de las áreas de investigación y de ser capaces de ofrecer apoyo en las nuevas plataformas tales como PDAs, para permitir llevar encima más aplicaciones interesantes de voz, tales como traducción de voz a voz, robots y/o asistentes interactivos personales digitales, que demuestran las nuevas áreas de interés de la investigación, teniendo así una plataforma que se pueda integrar más fácilmente en su investigación y hacer que nuestros trabajos sean más satisfactorios [20].

3.3.2. Estructura

La distribución *Flite* consta de dos partes bien diferenciadas:

La biblioteca que contiene el núcleo de la síntesis del código y las voces para *Flite*. Éstos contienen tres sub-partes:

- Idioma-modelos: procesamiento de textos, etc...
- Léxico y reglas de sonido
- Unidad de base de datos y definición de voz

CST val

Se trata de una estructura básica y simple que puede contener enteros, floats, strings y otras estructuras. También permite otro tipo de estructuras más complejas como listas y árboles.

La estructura *cst-val* está cuidadosamente diseñada para ocupar sólo 8 bytes (o 16 bytes en máquinas de 64-bits). La unión de múltiples estructuras está diseñada para que no hayan conflictos. Sin embargo, depende del hecho de que un puntero a una estructura *cst-val* garantice que se encuentre dentro de los límites de la dirección.

Como en *Festival*, los tipo usuario se pueden añadir a la estructura *cst-val*. En *Festival* se puede hacer al vuelo, pero se requiere la actualización de alguna lista cuando se ha añadido un nuevo tipo, esto no sería seguro para subprocesos.

API

Flite es una biblioteca diseñada para ser integrada en otras aplicaciones. Se incluye con la distribución un pequeño ejemplo ejecutable que permite la síntesis de cadenas y archivos de texto desde la línea de comandos.

Flite Binary: *Flite* puede ser adecuado para aplicaciones muy sencillas. En cambio, en *Festival* donde el tiempo de inicio es muy corto (menos de 25ms en un PIII a 500 MHz), lo que hace que deba ser llamada cada vez que se necesita para sintetizar algo.

Flite OutputType text: Si el texto contiene un espacio, se trata como una cadena de texto y se convierte en un fichero de voz, sin embargo, si no contiene un espacio de texto se trata como un nombre de archivo y el contenido de éste se convierte en un fichero de voz.
La opción -t especifica el texto que se va a tratar como tal (no como un nombre de archivo) y -f fuerza la salida como un archivo. Por lo tanto, *Flite-t hola* va a decir la palabra "hola", mientras que *Flite hola* dirá el contenido del archivo "hola". Del mismo modo *Flite "hola mundo"* dirá las palabras "hola mundo", mientras que *Flite-f "hola mundo"* dirá el contenido de "hola mundo" en un archivo. El *OutputType* que es el segundo argumento, es el

nombre de un archivo que se escribirá como salida, o se reproducirá en el dispositivo de audio directamente. Si no existe, entonces el audio se creará.

Voz de selección

Todas las voces en la distribución se recogen en una simple lista única en la variable global *flite-voice-list*. Se puede seleccionar una voz de la lista de la línea de comandos voz *Flite-AWB-f doc / alice-alice.wav* o si la lista de voces admitidas en el binario con *Flite-lv*.

3.4. Julius Speech to Text

Julius está desarrollado bajo Linux y Windows. También puede funcionar en algunos sistemas operativos como Solaris, FreeBSD y MacOsX. Desde que Julius se escribió en lenguaje C y tiene cierta dependencia en librerías externas, posibilita que funcione en otras plataformas. Los desarrolladores lo han llevado también a Windows Mobile, iPhone y entornos hardware. Julius reconoce voz en vivo mediante un dispositivo de capturación de audio en todos los sistemas operativos.

Para realizar el reconocimiento de voz con Julius, deberemos preparar modelos para dicha tarea y el lenguaje a seleccionar. Los modelos deben definir las propiedades lingüísticas del lenguaje en cuestión:

Unidad de reconocimiento, propiedades del audio de cada unidad y las restricciones lingüísticas para cada conexión entre las unidades. Normalmente, la unidad debería ser una palabra y se debe dar a la aplicación los siguientes modelos:

Diccionario, el cual determina el vocabulario. Esto define las palabras a ser reconocidas y sus pronunciaciones como una secuencia de fonemas.

Modelo lingüístico, que define las reglas de los niveles de sintaxis, los que determina las restricciones de conexión entre palabras. Esto debería indicar la restricción para el patrón de frases correctas.
También, esto puede ser una regla gramatical básica o un modelo probabilístico. El modelo lingüístico no es necesario para reconocer palabras aisladas.

Modelo acústico, el cual es un modelo estocástico de patrones de frecuencias de entrada por fonema. Julius toma el Modelo Oculto de Markov(HMM) para el modelado acústico.

Julius está distribuido, básicamente, como archivo fuente, y paquetes binarios para Linux y Windows también están disponibles. La fuente del archivo contiene el programa completo de Julius y herramientas relacionadas, información, muestras de archivos de configuración, muestras de código fuente y manuales online para Linux. El paquete binario está basado en la fuente del archivo, contiene ejecutables precompilados y relaciona archivos extraidos de la fuente del archivo. También se puede obtener como desarrollador capturas via CVS [21].

Las siguientes herramientas están incluidas:
• julius — Software principal de reconocimiento de voz de Julius
• adinrec — Detección de audio y herramienta de chequeo de grabación
• adintool — Herramienta de grabación / división / envío / receptor de flujo de audio
• jcontrol — Muestra de un módulo cliente escrito en C
• jclient.pl — Muestra de un módulo cliente escrito en Perl
• mkbingram — Conversor de archivos ARPA N-gram en formato binario
• mkbinhmm — Conversor de archivos HTK ASCII hmmdefs en formato binario
• mkbinhmmlist — Conversor de archivos HMMList en formato binario
• mkgshmm — Conversor de mono HMM a GS HMM para Julius

• mkss — Calculo de la media del espectro
• Herramientas para el modelado del lenguaje — mkdfa.pl, mkfa, dfadeterminize, dfaminimize, acceptcheck, nextword, generate, generate-ngram, gram2sapixml.pl, yomi2voca.pl
En Linux, las librerias, cabeceras de ficheros y algunos scripts que serán instalados por el desarrollador.
• libsent.a — Librería a bajo nivel de Julius
• libjulius.a — Librería principal de Julius
• include/sent/* — Cabecera para libsent
• include/julius/* — Cabecera para libjulius
• libsent-config,libjulius-config — Scripts para obtener los requisitos para la compilación con la librería libsent y la librería libjulius

3.5. NL Natural Language

El NL, lenguaje natural, se convierte en fundamental cuando se habla de comunicación con los usuarios finales. La comunicación lingüística natural implica dos capacidades principales:

Poner palabras a nuestros pensamientos e identificar los pensamientos de las palabras que percibimos. Estos son los objetivos que trasladaremos al sistema por medio del generador de lenguaje natural *NLG* y el entendedor de lenguaje natural *NLU*. Ambas tareas son subcampos del procesamiento del lenguaje natural, que a su vez puede ser visto como un subcampo de la informática y las ciencias cognitivas.

Los sistemas informáticos que automáticamente producen textos comprensibles para el ser humano y los que entienden estos idiomas, tienen mucho que ver con los modelos computacionales del lenguaje y su uso. En términos generales, los dos procesos tienen la misma finalidad, pero direcciones opuestas.

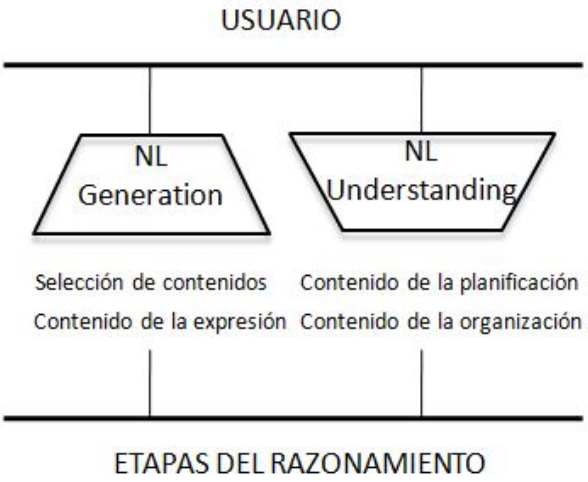


Figura 3.4: Esquema del tratamiento del Lenguaje Natural.

A pesar de que NLG y NLU puedan parecer estar estrechamente relacionadas, surgen diferentes problemas que requieren estrategias independientes.

3.5.1. NLG Natural Language Generation

La generación de lenguajes naturales NLG es el proceso de la construcción de un texto en lenguaje natural para la comunicación con fines específicos. Texto se refiere aquí a un término general y repetitivo aplicable a expresiones, o partes de ellas, de cualquier tamaño, tanto habladas como escritas. En el ser humano, el que sea hablado o escrito tiene consecuencias en el nivel deliberativo y de edición que ha tenido lugar; si el lenguaje es hablado puede faltar revisión ya que la mayoría de los programas actuales pueden hablar, si bien casi todos sólo presentan palabras en una pantalla. La decisión de revisar o usar la palabra escrita o hablada no es una opción para la generación del programa en la actualidad; pero se debe abordar el tema en el diseño de un programa en particular.

El principal énfasis de la generación de lenguajes naturales no es solo el

facilitar el uso del ordenador sino también el desarrollar una teoría computacional de la capacidad del lenguaje humano. En este sentido constituye una herramienta para extender, aclarar y verificar teorías que se han formulado en lingüística, psicología y sociología acerca de la comunicación entre humanos.

Un generador de lenguaje natural típicamente tiene acceso a un gran conjunto de conocimiento del cual ha de seleccionar información para presentar a los usuarios en varias formas. El generar texto es, pues, un problema de toma de decisiones con múltiples restricciones: de conocimiento proposicional, de herramientas lingüísticas disponibles, de los objetivos de la comunicación del usuario a quien se dirige el texto, y de la situación y del discurso pasado. Se trata de identificar los factores involucrados en este proceso y de determinar la mejor forma de representar estos factores y sus dependencias [23].

Proceso de generación

Para generar un texto, un sistema debe escoger cierta información de la base de conocimiento, decidir cómo organizarla, y determinar cómo producir el texto en lenguaje natural, lo cual incluye el decidir acerca de la entrada del léxico y de las estructuras sintácticas. Esto hace que el proceso de generación se divida en dos partes: una componente de planificación del texto y una componente de generación propiamente dicha. Ésta a su vez se divide en dos tareas: la de escoger los objetos del léxico y la de efectuar selecciones gramaticales. En todo esto se trata de escoger apropiadamente para expresar lo mejor posible el significado deseado.

Generación de textos

Elección de léxico En este campo se trabaja desde los puntos de vista lingüístico y computacionales. Este problema es difícil. En algunos casos se trata al nivel de la representación conceptual sobre cuya base opera el ge-

nerador. Este método puede resultar en una arquitectura de procesamiento más simple. En otros casos se trata de que la elección de léxico no ocurra de una forma aislada sino como parte del problema de la elección lexicogramatical. En otros casos se han desarrollado generadores basados en la teoría significado- texto, donde el léxico desempeña un papel central, que influye en el proceso de generación.

Recursos gramaticales En este campo la gramática a utilizar es una componente importante del sistema, y toma decisiones de cómo expresar sintácticamente la información deseada. En algunos casos se trata de una gramática que tenga en cuenta las correferencias del discurso. En otros casos la gramática se diseña para manejar las necesidades de la generación de oraciones incrementalmente.

Morfología Este campo trata de la formación de la palabra (inflexión, palabras derivadas, palabras compuestas). Se basa en un léxico que contiene entradas para un conjunto de palabras y reglas para las inflexiones.

3.5.2. NLU Natural Language Understanding

El módulo NLU elige la más apropiada interpretación de un conjunto de posibles interpretaciones, dando una salida en forma de texto en NL lenguaje natural.

En nuestro caso, la ontología especifica el dominio de validaciones de una gran varieda de posibles consultas por el usuario y hace esto posibilita la reducción a un espacio de situaciones más manejables. Además, para evitar la ambigüedad excesiva al resolver el significado de las entradas, este módulo acepta únicamente una sola frase de los usuarios finales.

Las operaciones generales conducidas por el módulo NLU son las vistas en la figura enterior, cada sentencia del usuario es procesada por un algoritmo de

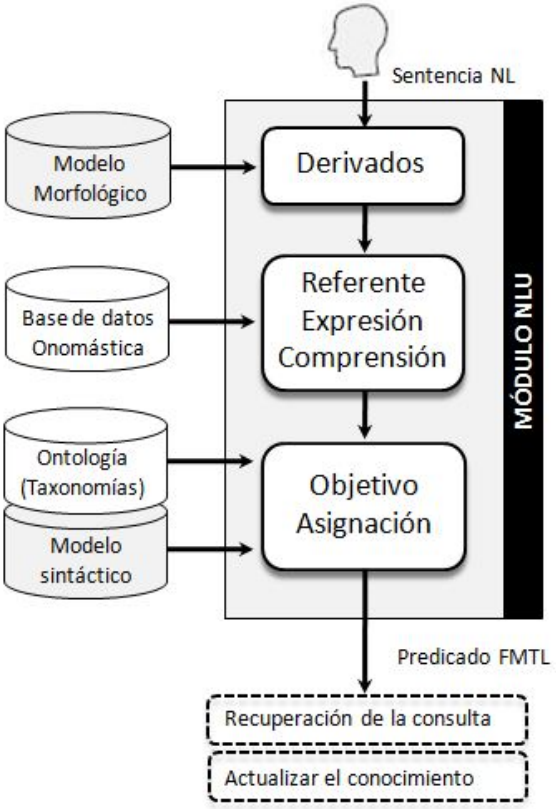


Figura 3.5: Las sentencias escritas por el usuario son convertidas individualmente en los predicados conceptuales.

división y el contenido de estos son ligados a conceptos de la globalidad de la ontología. Después de esto, el contexto específico de la sentencia se encuentra por sobre todas las expresiones referidas a la entidad correspondiente, con la ayuda de la llamada onomástica. Por último, la sentencia interpretada es analizada a un nivel sintáctico y semántico, y estos contenidos son asignados al predicado más adecuado a fin de generar agentes virtuales en la escena [22].

3.6. Base de Datos

Las tablas utilizadas en este proyecto se organizan de la siguiente manera:

La tabla **Inference** es la tabla más importante, ya que es la que almacena los predicados generados por el sistema de visión. Ésta, guarda el tipo de agente y su número identificativo, la acción que realiza y el lugar dentro de la escena donde actúa.

La tabla **Nlu** es la responsable de almacenar todas las consultas realizadas por el usuario en modo texto. Esta tabla necesita información de la tabla *Inference*, la cual usará para obtener la información requerida. Existe una consulta especial, *query 7*, que buscará información en la tabla *Consciousness*, explicada a continuación.

La tabla **Consciousness** se encarga de almacenar aquellas acciones realizadas en un fotograma en concreto, dontándola de una prioridad. Esta prioridad es importante ya que ordenará las respuestas en función de ésta.

La tabla **Learn** guarda aquellas acciones que más se repiten en un intervalo entre fotogramas. Esto permitirá al sistema hacer una previsión de que acción y en que momento tiene más posibilidad de producirse.

La tabla **Nltgresults** es la tabla encargada de guardar los mensajes, en lenguaje natural, que el Avatar deberá decir. Esta tabla recibe información de las tablas *Inference*, *Consciousness*, *Learn*.

En la figura 3.6 podemos observar la relación que existe entre las tablas de la base de datos.

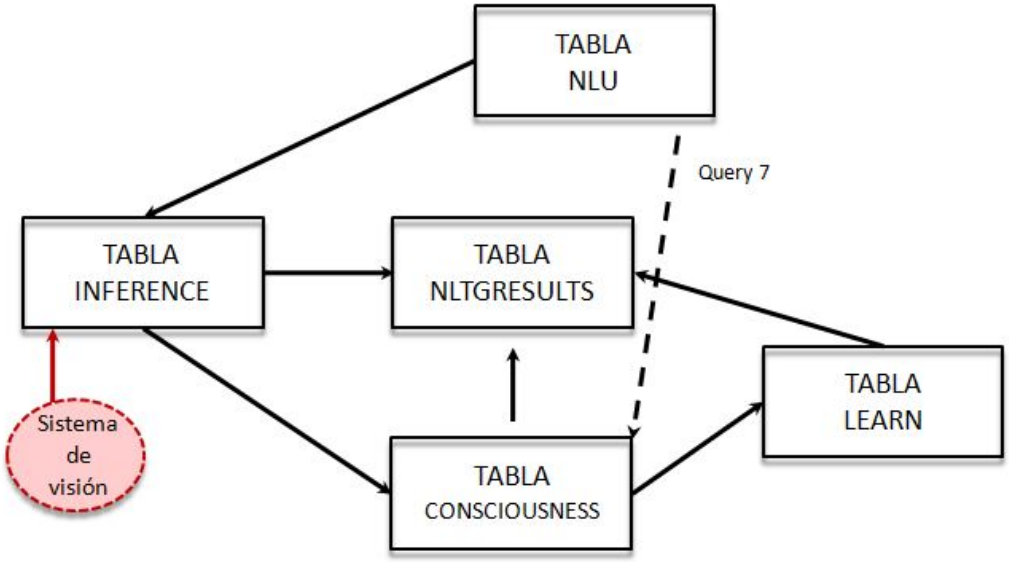


Figura 3.6: Relación entre las tablas utilizadas en la aplicación.

Capítulo 4

Diseño y desarrollo

4.1. Entorno gráfico

La base del entorno gráfico del Avatar ha sido programada bajo el entorno de programación Eclipse en lenguaje JAVA. Gracias a las librerías para gráficos 3D, OpenGl, se ha podido representar gráficamente una cabeza parlante 3D. Como se ha explicado en el capítulo anterior, XFace es capaz de construir modelos de Avatar. Éstos se guardan en el fichero de salida FDP, el cual se puede ver con más detalle en el Apéndice B.

Inicialmente, se ha realizado la carga del fichero FDP que contiene información de la forma de la cabeza, cabello, ojos, boca, etc..., del Avatar. Éste, aparece inicialmente en estado de reposo por lo que su expresión facial será neutral. Un complemento a esta cabeza parlante 3D es un cubo giratorio que lo envuelve. Dicho cubo también representa, con sus formas, el mismo estado de ánimo que el Avatar.

La Clase XFaceAppletDriver es la encargada de gestionar todo el proceso de carga y animación del Avatar. En un estado inicial o base, éste no reproducirá mensaje alguno. Simplemente estará a la espera de que se genere uno nuevo para ser tratado.

Una vez generado un mensaje, pasaremos a un estado general donde se irán reproduciendo mensajes al mismo tiempo que se irán almacenando aquellos nuevos que se generen y puedan ser reproducidos en ese momento.

Cada mensaje trae consigo tres ficheros que serán tratados de diferente manera. Los ficheros de sonido ″.wav″ son tratados por la Clase XFaceSound responsable de generar el efecto sonoro. Los ficheros de estado de ánimo ″.anim″ son tratados por la Clase Background, la cual carga el tipo de cubo adecuado para cada estado, y la Clase MorphChannel, encargada de generar las deformaciones en el Avatar para simular un cierto estado de ánimo. Por último, los ficheros para los fonemas y visemes ″.pho″ que también son tratados por la Clase MorphChannel y se encargará de producir la deformación necesaria de la cara del Avatar, para generar las expresiones faciales.

4.1.1. Background

Del mismo modo que en el Avatar simulado Max Headroom, se ha insertado un cubo giratorio que envuelve a nuestro Avatar. Este cubo, cambia su aspecto en función del estado de ánimo de la cabeza parlante 3D. Para poder implementar los diferentes cubos giratorios, se ha aprovechado las mismas librerías utilizadas para representar gráficamente al Avatar. El fichero ″.anim″ contiene el estado de ánimo a utilizar en cada mensaje. Una vez obtenido el estado, la Clase Background se encargará de seleccionar el cubo a mostrar para éste. En la figura 4.1 podemos observar el cubo en cuatro de sus siete estados. Estado neutral, consciencia, disgustado y contento respectivamente.

4.1.2. Los mensajes

Comprobar si existen mensajes nuevos es una parte fundamental de esta aplicación, ya que uno de los objetivos principales es que el Avatar los vaya diciendo. Para ello, se debe detectar que se ha generado e insertado un

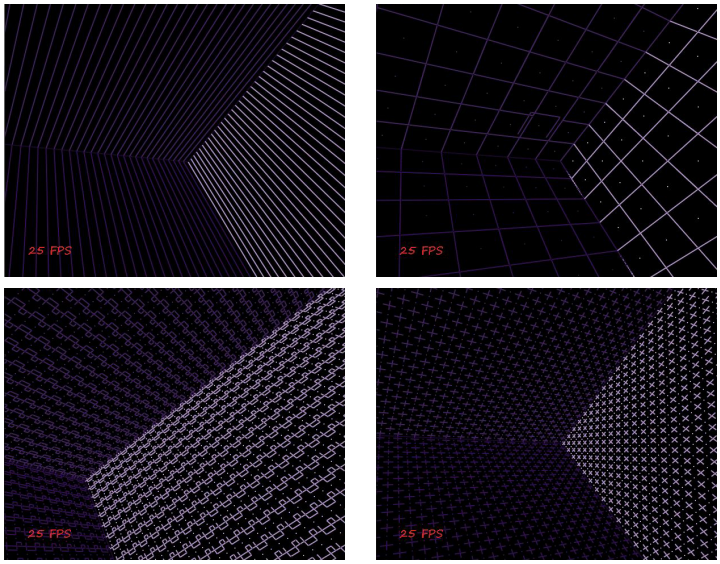


Figura 4.1: Las formas a representar, dan apoyo a la expresión del Avatar en cada uno de sus estados. La rotación del cubo es en sus tres ejes de representación, con movimientos ya predefinidos siguen una trayectoria coherente, manteniendo al Avatar dentro de él.

mensaje nuevo. Existen tres tipos de ficheros que genera el *Text to Speech*, pero solo nos basaremos en uno de ellos para hacer dicha comprobación. El fichero ″.pho″, que es el encargado de guardar los fonemas con sus tiempos de duración correspondientes, es el que el sistema estará comprobando constantemente. Para ello, mirará la fecha de su modificación y si ésta es diferente a la guardada la última vez que se generó el mensaje nuevo, se considerará que se ha actualizado y por lo tanto deberá ser leído o en este caso reproducido.

Un problema inherente a la generación de mensajes surge cuando se genera uno antes de que sea leído el anterior. Principalmente, esto producía el problema de sobreescritura de uno de los mensajes nuevos no leídos o detectados. Para solucionar este problema se ha creado la estructura *queue*, cola, que nos permite almacenar mensajes y evitar la sobreescritura.
La cola funciona de la siguiente manera, todos los ficheros que se van creando se van guardando en esta cola, así pues, por muy rápido que se genere un mensaje jamás sobreescribirá a otro ya existente que aún no se haya repro-

ducido. Se pueden guardar hasta un máximo de 100 mensajes sin ser leídos, con lo que nos aseguramos que no perderemos ninguno.

4.1.3. Sincronismo

La sincronización es parte fundamental para conseguir el máximo realismo, de nuestra cabeza parlante 3D, cuando nos está reproduciendo cualquier mensaje. Un concepto aún no explicado es el del viseme, el cual se encarga de situar el rostro del Avatar en cierta posición para reproducir un cierto fonema. Es decir, si queremos que nos diga ″hola″, el viseme que representa la ″ho″ hará que la cara se coloque en esta posición.

Dicho esto, uno de los problemas más engorrosos de solucionar fué la sincronización entre los visemes y los fonemas. No quedaba nada bien ver como el Avatar se posicionaba con una cierta expresión y que ese fonema se decía unas milésimas después. La utilización de *threads* solucionó este problema, ya que cuando un *thread* se encarga de gestionar los visemes, otro, lo hacía para gestionar el sonido y poder reproducir un fonema sincronizadamente.

4.1.4. Entorno web

La aplicación es su estado final, es lanzada desde un entorno web utilizando la máquina virtual de JAVA. Como se ha comentado en la sección anterior, se ha utilizado la estructura cola para almacenar todos los mensajes que se van generando. Esto nos lleva a la necesidad de guardar físicamente estos ficheros a disco. Para ello, al lanzarse la aplicación, se crea una carpeta temporal en el directorio caché para aplicaciones JAVA. A medida que los mensajes se van generando, se van guardando en dicha carpeta y cuando estos han sido reproducidos se van borrando progresivamente.

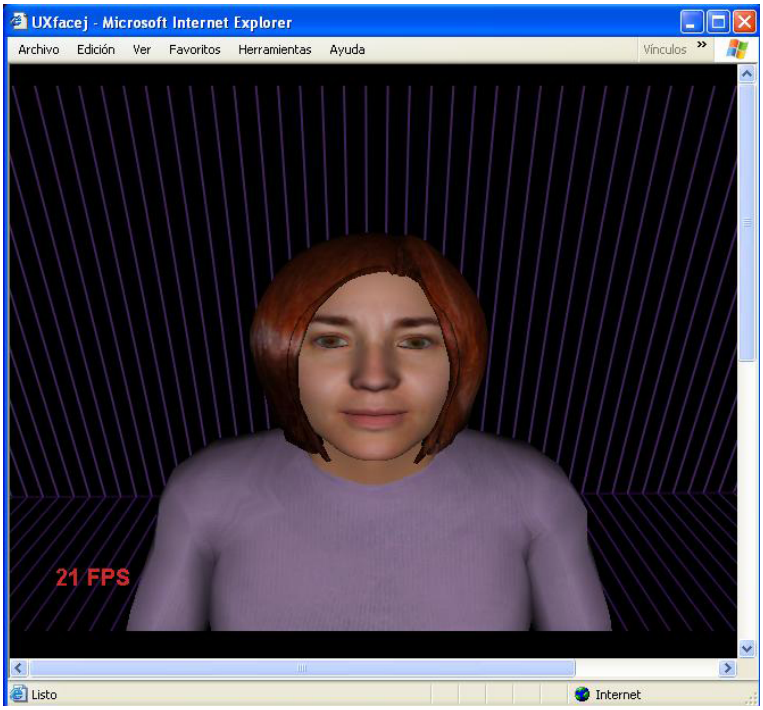


Figura 4.2: Aplicación ejecutada desde un navegador web.

4.2. Text to Speech

Como se ha explicado con anterioridad, la aplicación utilizada para convertir texto a sonido es *Flite* en su versión 1.4. Esta versión no era del todo adecuada para nuestros intereses, por lo que se han hecho algunas modificaciones y adaptaciones. La principal modificación y la más costosa en tiempo de realización, fue la selección del tipo de voz a utilizar. *Flite* dispone de diferentes fuentes de sonido que genera tonos de voz a 8 bits o a 16 bits. Para nuestro caso, nos era necesario una fuente de voz femenina y a ser posible con la mayor calidad de sonido. A primera vista, dicha selección no era difícil, ya que *Flite* dispone de una fuente de voz femenina de 16 bits, pero surgió un contratiempo.

A diferencia de las otras fuentes de voz, ésta no tenía en cuenta el tiempo de los espacios en blanco producidos entre palabra y palabra. Tal cosa

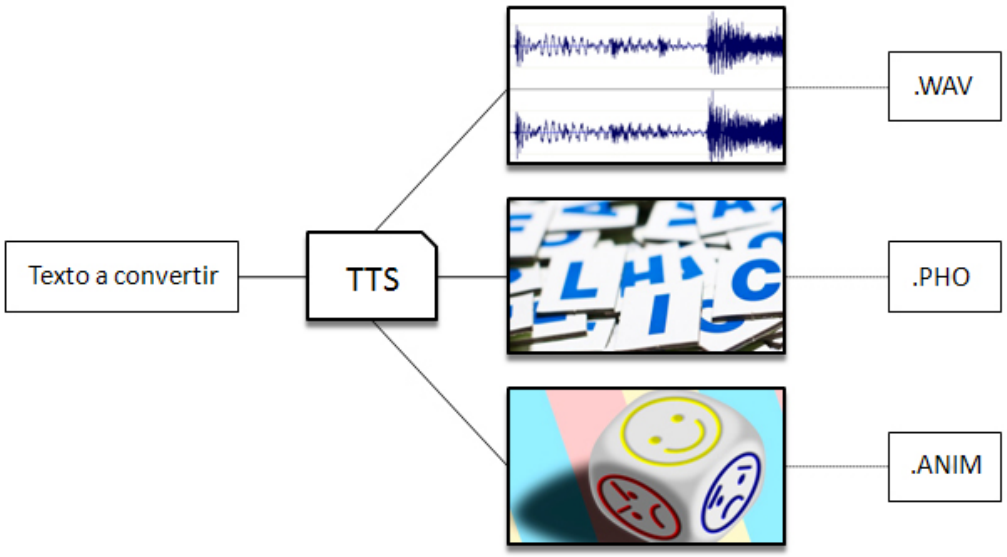


Figura 4.3: Al convertir el texto a sonido, se crean tres ficheros. Un primer fichero con extensión ".wav" que contiene el sonido de la frase a reproducir, un segundo fichero que contiene todos los fonemas y los tiempos de duración de éstos y un tercer fichero que contiene el estado de ánimo que deberá representar el Avatar al reproducir la frase.

provocaba falta de sincronismo entre la vocalización y sonido, a medida que la frase a reproducir se hacía más larga. Este problema solo se producía con esta fuente de voz y sin embargo con las otras no existía dicho problema.

Para solucionar esto, se tomó la voz femenina de 16 bits pero con los tiempos para los fonemas de otra de las fuentes de voz. De esta manera se consiguió el resultado deseado. Se cortó el flujo de datos que calcula los tiempos para la voz de mujer y se forzó el calculo de los tiempos para otra de las voces. Estos valores se guardan en un fichero temporal al que se le irá concatenando el valor numérico que representa a un cierto fonema. El fichero resultante es *message.pho* para los fonemas y *message.wav* para el fichero de sonido.

4.2.1. Diccionario

El diccionario es imprescindible para poder tratar fonéticamente cualquier palabra y con ello cualquier frase que se quiera construir. Mediante éste, se consigue relacionar un fonema con un valor numérico que posteriormente será tratado por nuestra aplicación final basada en XFace, como se ha comentando en el apartado anterior.

El diccionario funciona de la siguiente manera, una vez introducido la frase a tratar, *Flite* separa fonéticamente cada palabra de la frase, posteriormente, estos fonemas son filtrados por el diccionario el cual asocia un número entero a cada fonema.

Como podemos ver en la tabla 4.1, si el diccionario recibe el fonema ″ae″, retornará el valor ″1″ y así sucesivamente a medida que vaya recibiendo fonemas.

El resultado del fonema se irá concatenando con un tiempo de duración calculado previamente por la aplicación *Flite*

4.2.2. Emoticonos

Además, de los ficheros de fonemas y de sonidos, también se genera otro tipo de fichero, éste determinará el estado de ánimo del Avatar.
Para ello, al propio *Flite* se le ha añadido un pequeño *parser* capaz, en función de unos carácteres que debe llevar el texto a convertir, de determinar el estado de ánimo requerido.

Una vez determinado este estado, se generá un fichero, llamado *message.anim*, que contendrá el nombre del mismo para que posteriormente nuestro aplicativo determine el estado de ánimo. Como podemos ver en la figura 4.4,

Número	Inicio	Fonema	Número	Inicio	Fonema
17	0.254	th	15	1.792	s
1	0.314	ae	13	1.851	r
15	0.465	s	2	1.926	aa
4	0.594	eh	19	1.989	n
20	0.715	k	6	2.039	ih
1	0.751	ae	20	2.101	ng
19	0.804	n	6	2.163	ih
19	0.842	d	19	2.224	n
21	0.953	p	17	2.245	th
1	1.019	ae	1	2.289	ae
19	1.093	t	21	2.369	m
4	1.191	ey	4	2.508	ey
15	1.284	s	19	2.560	n
19	1.374	t	4	2.650	eh
13	1.411	r	19	2.712	n
6	1.554	ih	19	2.802	t
1	1.591	ah	13	2.840	r
19	1.643	n	1	2.866	ah
6	1.717	ih	19	2.936	n
			15	3.013	s

Cuadro 4.1: Desglose de los fonemas de la frase: ″the second pedestrian is running in the main entrance″

cada emoticono tiene asignado uno.

4.3. Speech to Text

4.3.1. Gramática de reconocimiento

En Julius, el reconocimiento de la gramática se hace mediante dos archivos diferenciados, tales como, ″.grammar file″ y ″.voca file″. El primer fichero define el nivel de sintaxis de la categoría, es decir, permite la conexión de palabras con el nombre de su categoría. El segundo fichero define la palabra y las candidatas en cada categoría, con la informaición de la pronunciación.

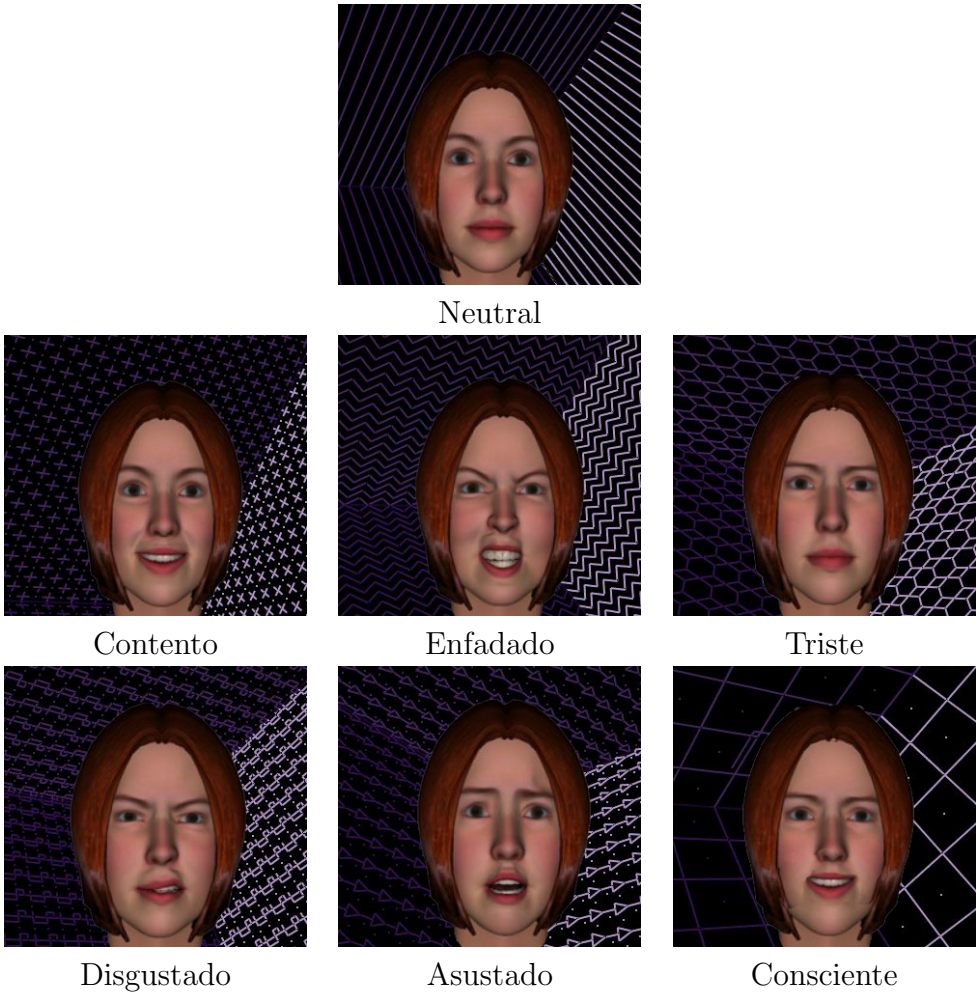


Figura 4.4: El Avatar puede representar hasta siete estados de ánimo. Un emoticono al inicio de cada frase determinará qué estado representará.

.grammar file La conexión de las palabras debe ser definida en '.grammar file', utilizando el nombre de las categorías de las palabras como símbolos terminales. Es muy similiar a la notación BNF[1], y el símbolo inicial para empezar debe ser 'S'.

[1]La notación de Backus-Naur, también conocida por sus denominaciones inglesas Backus-Naur form (BNF), Backus-Naur formalism o Backus normal form, es una metasintaxis usada para expresar gramáticas libres de contexto: es decir, una manera formal de describir lenguajes formales

S :	NSB SENT NSE
S :	NSB SENT NSE
QUERY :	List agents meeting
QUERY :	List agents running

Cuadro 4.2: En este ejemplo, (NSB, NSE, QUERY, AGENTS, ACTION) son categorías de palabras, y sus contenidos deben estar especificado en .voca file. NSB y NSE corresponde al silencio inicial y al silencio final de la entrada de la voz, y debe estar definido en toda la gramática para Julius.

Token	Fonema
<s>	sil
</s>	sil
List	l iy s th
agents	ae y eh n t s
meeting	m ih t ih ng
running	r aa n ih ng

Cuadro 4.3: .voca file contiene la definición de las palabras para cada categoría definida en .grammar file.

Las reglas de escritura deberán ser definidas en cada línea, utilizando ":" como delimitador. Carácteres del alfabeto ASCII, números, guiones bajos, mayúsculas y minúsculas, son permitidos como nombres para los símbolos. Por ejemplo, la expresiones "List agents meeting", o "Show me situation agent one, agent two and agent three" son aceptadas.

La primera columna es la cadena que se emitirá una vez reconocidos, y el resto es la pronunciación. Espacio y tabulación son el separador de campo. En la figura 4.2 observamos el contenido del fichero .grammar y en la figura 4.3 observamos el contenido del fichero .voca.

4.4. Interacción

La interacción entre el Avatar y nosotros es la parte del proyecto más interesante, ya que el objeto de éste no es otro que comunicarse con un Avatar y que éste reaccione a nuestras peticiones o necesidades. La interacción se puede realizar de dos modos, tanto escrita como hablada.

En la modalidad de escritura, introducimos peticiones y consultas a través de una consola desde una aplicación web, sin embargo, mediante la modalidad de voz, la petición o solicitud hablada se transforma en texto y al igual que en la modalidad anterior el Avatar reaccionará a dichos estímulos.

Los elementos de la escena, ya sean agentes que participan o lugares donde se desarrollan las acciones, están etiquetadas y clasificadas mediante una estructura de árbol como se muestra a continuación.

4.4.1. Mensajes

El primer paso que realiza la aplicación es el de comprobar el valor numérico *id* del último mensaje insertado en la tabla de mensajes de texto *nltgresults*. Para ello, se realiza una consulta en SQL a la base de datos. Esto es importante, ya que podemos saber en cada instante si se ha generado un nuevo mensaje en ésta. A continuación, una vez dentro del bucle general, se va comprobando si el valor número del identificador cambia con respecto a la última vez que se consultó. Si existe diferencia entre el valor obtenido recientemente y el anterior, pasaremos por un bucle interno tantas veces como la diferencia existente entre estos dos valores. Dentro del bucle ejecutaremos el programa de Text to Speech *Flite* pasándole como parámetro el mensaje de texto que se insertó en la tabla *nltgresults*. Como se ha explicado en apartados anteriores, *Flite* recibe el texto y generará un fichero de sonido en formato *wav*, con el mensaje a decir, un fichero con los fonemas

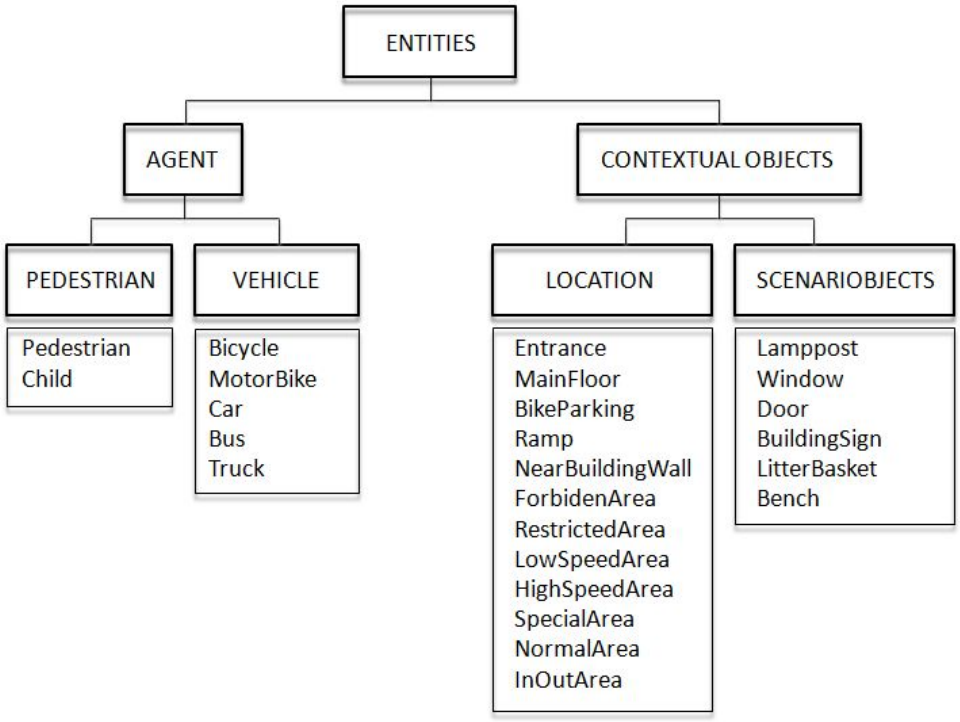


Figura 4.5: Entidades de la escena.

y los tiempos para que el Avatar simule la pronunciación del mensaje y por último, generará un fichero con el correspondiente token indicando el estado de ánimo que deberá adoptar el Avatar al decir dicho mensaje. Este proceso se repetirá tantas veces como mensajes nuevos se hayan generado e insertado en la tabla correspondiente de la base de datos.

En la figura 4.7 podemos observar el procedimiento a seguir cuando se detecta la inserción de un nuevo mensaje en la base de datos.

4.4.2. Acciones

Cuando el sistema de visión ha detectado una acción, dentro de los límites donde éste trabaja, generará un predicado donde saldrá reflejado el actor, el número de éste, la acción realizada por éste y el lugar donde se produce el suceso. Esta información se guardará en la tabla *Inference* de la base de datos y es necesaria para poder generar el contenido del nuevo mensaje.

Figura 4.6: La aplicación entra en un bucle infinito a la espera de encontrar nuevos mensajes, nuevas acciones o nuevas consultas.

La aplicación, al igual que en el caso de los mensajes, obtendrá el valor numérico *Id inference* de la última acción insertada en la tabla. De esta forma, comparando este valor con las siguientes consultas sobre este identificador, se podrá saber si se ha generado una nueva acción.

En la figura 4.8 podemos observar el procedimiento a seguir cuando se detecta la inserción de una nueva acción en la base de datos.

De haberse producido una nueva acción, se generará un nuevo mensaje que se insertará en la tabla *nltgresults* (hay que recordar que es la tabla que almacena todos los mensajes en su estado final) y también se guardarán cierto datos en la tabla *Consciousness*, la cual se explicará en la siguiente sección.

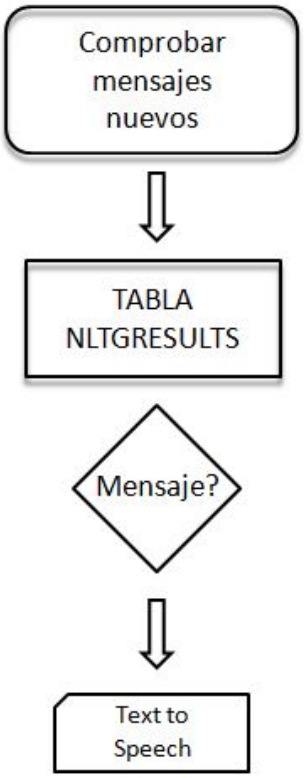


Figura 4.7: Lector de nuevos mensajes.

4.4.3. Consultas

Mediante este apartado podemos interactuar con el Avatar de una forma directa, donde formularemos una pregunta que éste deberá responder. El primer paso, pues, es saber el valor numérico de la última consulta realizada. Al igual que en los apartados anteriores, realizaremos una consulta a la base de datos, más concretamente a la tabla *Nlu* para obtener dicho valor, que es el identificador de la consulta. De existir una nueva consulta, se procederá a analizar sintácticamente la frase para determinar si es una consulta válida o no. Para ello, se ha creado un parser que irá recorriendo palabra a palabra la frase de la consulta. De ser correcta, se le asignará un valor numérico a una etiqueta que nos valdrá posteriormente para determinar las condiciones a filtrar y así delimitar los parámetros de la respuesta a la consulta. Reco-

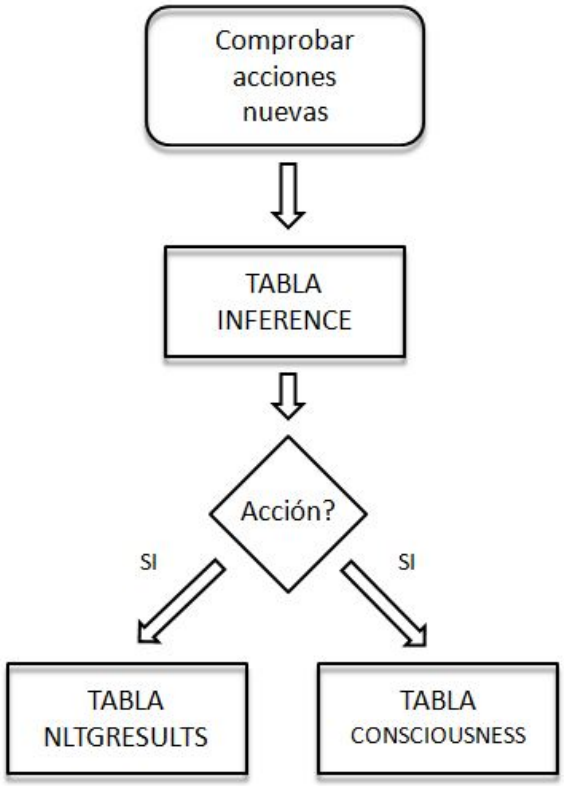


Figura 4.8: Genera el texto de la nueva acción.

rreremos fila a fila la tabla *Inference* e iremos guardando aquella información necesaria a filtrar, como el tipo de agente y su número, la acción del verbo, el fotograma y el lugar donde se ha producido la acción. A cada verbo se le ha asignado un estado de ánimo que el Avatar deberá representar cuando reproduzca el mensaje.

Para *run, running, leave, leaving* se ha asignado el emoticono :V
Para *hit, kick, punch, escape, shove* se ha asignado el emoticono :@
Para *theft, chase, chasing* se ha asignado el emoticono :(
Para *stole, stealing, cross, crossing* se ha asignado el emoticono :S
Para *meet, meeting, pick up, picking up* se ha asignado el emoticono :D
Para cualquier otro que no esté en la lista, se ha asignado el emoticono :|

El siguiente paso es determinar que resultado queremos obtener. Por ejemplo, si queremos saber ″Cuantos agentes se han encontrado en la entrada entre los fotogramas 10 y 15″, la condición para que se cumpla esto ha de ser que la acción sea *meet* (encontrarse), para cualquier fotograma que se encuentre entre [10,15], que el lugar sea la entrada (Entrance equivale a Location1) y para cualquier tipo de agentes. Esto generará las siguientes frases: Si no hemos encontrado ninguna coincidencia ″I haven't found anybody″, sin embargo, si se ha encontrado a un agente la respuesta será ″I have found one agent″ y si se ha encontrado a más de un agente en estas condiciones, la respuesta será ″I have found (número de agentes encontrados) agents″. Este mensaje es enviado a la tabla *nltgresults* de la base de datos. Esto hace que se genere un mensaje nuevo, con lo que la parte encargada de leer mensajes, anteriormente explicada, lo detectará y procederá a tratarlo.

En la figura 4.9 podemos observar el procedimiento a seguir cuando se detecta la inserción de una nueva consulta en la base de datos. Estas consultas pueden ser insertadas de forma escrita desde una aplicación web o a través del programa Speech to text.

4.5. Consciencia

Durante las últimas décadas, la tecnología de computadores electrónicos ha hecho enormes progresos y existen pocas dudas de que en las próximas décadas tengan lugar nuevos avances en velocidad, capacidad y diseño lógico. Hay algo casi estremecedor en el ritmo del progreso. Pero el ″pensar″, eso sí ha sido siempre un prerrogativa humana. La pregunta afecta a temas profundos de la filosofía. ¿Qué significa pensar o sentir? ¿Qué es una mente? ¿Existen realmente las mentes? Y sobre todo, ¿podría darse que el placer y el dolor, la estimación de la belleza y el humor, la consciencia y el libre albedrío fueran capacidades que emerjan de modo natural cuando el comportamiento algorítmico de los robots electrónicos llegue a ser suficiente complejo?

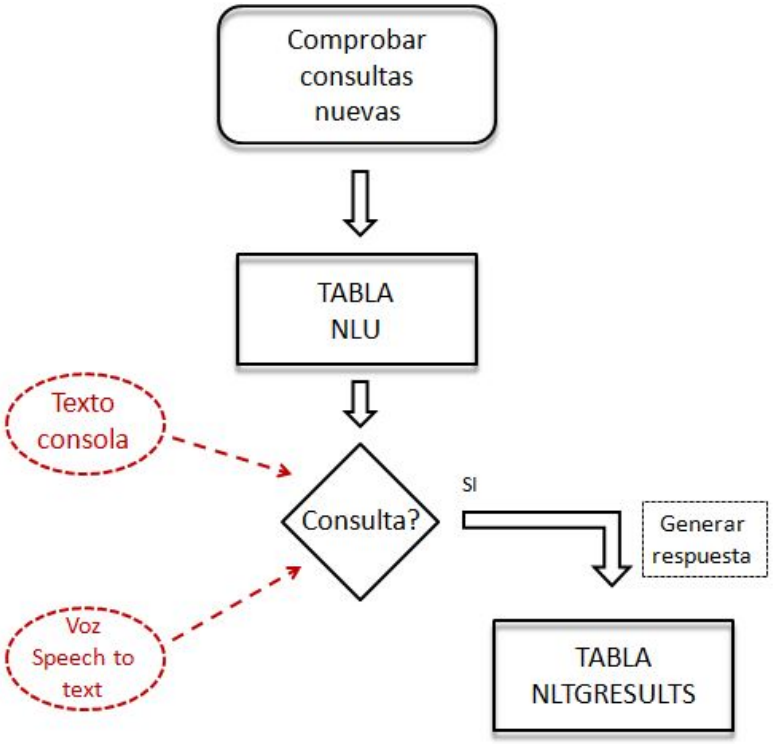


Figura 4.9: Lector de nuevas consultas.

En este proyecto se ha querido dotar de un cierto nivel de ″consciencia″ para que se puedan tener en cuenta ciertos acontecimientos que irán sucediendo en la zona de acción del sistema de visión. Por eso, cuando se produce un nuevo acontecimiento, en la tabla *Consciousness* de la base de datos, se irá guardando cierta información relevante para poder simular el estado de consciencia.

En esta tabla, se guardará el verbo de la acción *Action*, una referencia al mensaje generado *Refaction*, el fotograma en que se produjo dicha acción *Frame*, una referencia a la tabla de los mensajes *nltgresults* para recuperar el mensaje que se generó al producirse la acción y por último, el nivel de prioridad *Priority* de la acción. El nivel de prioridad es calculado en función

del tipo de verbo dentro del predicado.
Por ejemplo:

Prioridad 1, para *theft, stole, stolen, steal, stealing*
Prioridad 2, para *hit, kick, punch, shove*
Prioridad 3, para *chase, chasing, run, running, escape*
Prioridad 4, para *meet, meeting, met, pick up,picking up*
Prioridad 5, para *leave, living, enter, entered*
Prioridad 6, para *walk, walking, walking with, walked*
Prioridad 7, para el resto de verbos

El estado de consciencia se pone en marcha cuando aparece la siguiente solicitud en la tabla de consultas *nlu*: ″What's the most important thing happened today?″, ″¿Qué es lo más importante ocurrido hoy?″. El sistema detecta esta solicitud como una consulta especial y por ello la tratará de diferente forma que las demás.
Una vez que el parser ha analizado la estructura sintáctica de la consulta y la da como válida, se accederá a la tabla *Consciousness* de la base de datos y retornaremos a la aplicación todas las filas de la tabla ordenadas por prioridad, es decir, las de mayor prioridad primero. Tanto si la prioridad es 1 como si la prioridad es 2, se generará una nueva frase que será ejecutada por el text to speech *Flite* de la siguiente forma: There was a *Action* in the frame *Frame*. A continuación, mediante la referencia al mensaje generado en la tabla de mensajes *Refaction*, el mensaje en cuestión será ejecutado de nuevo por el text to speech *Flite*.

También se le ha dotado de una cierta capacidad de aprendizaje que consiste en ir guardando en una tabla llamada *Learn* dentro de la base de datos, aquella cantidad de acciones repetidas entre un intervalo de fotograma, es decir, guardar el número de veces que se produce una acción en un mismo intervalo.

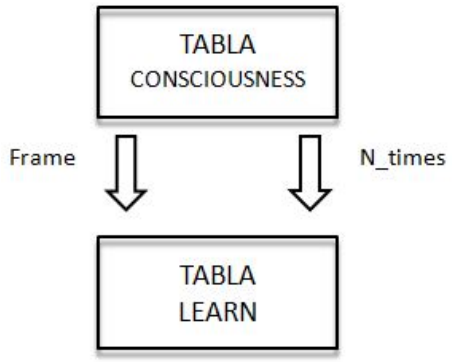


Figura 4.10: Aprendizaje de sucesos.

Tabla 'Learn'		
Action	Frame	Ntimes
Theft	6000	6
Stole	7300	10
Chase	240	3
Meet	2000	13
Walking	7300	96

Cuadro 4.4: Ejemplo de la Tabla ″Learn″ de la base de datos

En un momento dado se toman los resultados obtenidos en la tabla ″Consciousness″ y se van insertando en la tabla ″Learn″ el fotograma con el mayor número de repeticiones sobre una acción.
Por ejemplo: Supongamos que la acción ″theft″ se produce 6 veces en diferentes días entre los fotogramas 5000 y 10000, la tabla ″Learn″ guardará en una de sus filas la siguiente información:

Esta información se irá actualizando día a día o incluso podría cambiar si el número de veces que se realiza dicha acción en otro fotograma es mayor al actual registrado en la base de datos. Esto nos será de mucha utilidad para determinar qué probabilidad existe de que suceda una acción en un momento del día en concreto.

Capítulo 5

Resultados

5.1. Introducción

En este capítulo se muestran los resultados obtenidos a partir de un conjunto de consultas realizadas al Avatar, en los que éste deberá responder en función de sus conocimientos adquiridos. Al realizarse la consulta, el sistema compara la estructura sintáctica de ésta para comprobar que sea correcta. En el caso que no cumpla dicha estructura, el sistema devolverá un mensaje de error indicando que la consulta no es correcta.

5.2. Especificaciones del software

Lenguaje de programación : Para crear el entorno donde se va a mover el Avatar, se ha utilitado el lenguaje de programación Orientado a Objetos JAVA. La razón principal de utilizar JAVA en lugar de otros lenguajes de programación es debido a su fácil portabilidad a diferentes Sistemas Operativos. Además, la gran mayoria de API's disponibles y módulos integrados están programados en JAVA.

Sin embargo, para el módulo conversor de texto a voz, de voz a texto y

el generador de consultas, se ha utilizado el lenguaje de programación C++ . Esto es así, debido a que los dos primeros módulos ya están programados en este lenguaje y el tercero, creado desde cero, necesita integrarlos, por lo que se pensó que lo más indicado era programarlo también en C++ .

Entorno de Programación : Eclipse. Es un entorno de desarrollo integrado de código abierto multiplataforma para desarrollar y testear aplicaciones en lenguaje JAVA, entre otros.

Acceso a la Base de Datos : APACHE + MYSQL, para el acceso a la base de datos, ya sea para realizar consultas o para recoger información sobre sucesos.

5.3. Test

El Avatar tiene cuatro tipos de comportamiento diferenciados, pero relacionados entre sí por la base del conocimiento. El primer comportamiento viene dado por aquello que reconoce en el exterior. A medida que va visualizando acciones, el Avatar las va citando en tiempo real. Como se puede observar en la figura 5.1, las frases generadas se van guardando en la tabla destinada a almacenar mensajes de la base datos.

El segundo comportamiento va relacionado con las consultas que se le puede realizar al Avatar, ver Apéndice B. Se han preparado 60 consultas que la aplicación es capaz de entender y responder. Por ejemplo, a la pregunta " Lista de agentes que están en la entrada", "List agents in the entrance", el sistema nos responderá con la lista de agentes con su número identificador correspondiente que están en la entrada. En la figura 5.2 observamos el resultado de la consulta,

El tercer comportamiento es muy parecido al segundo comportamiento,

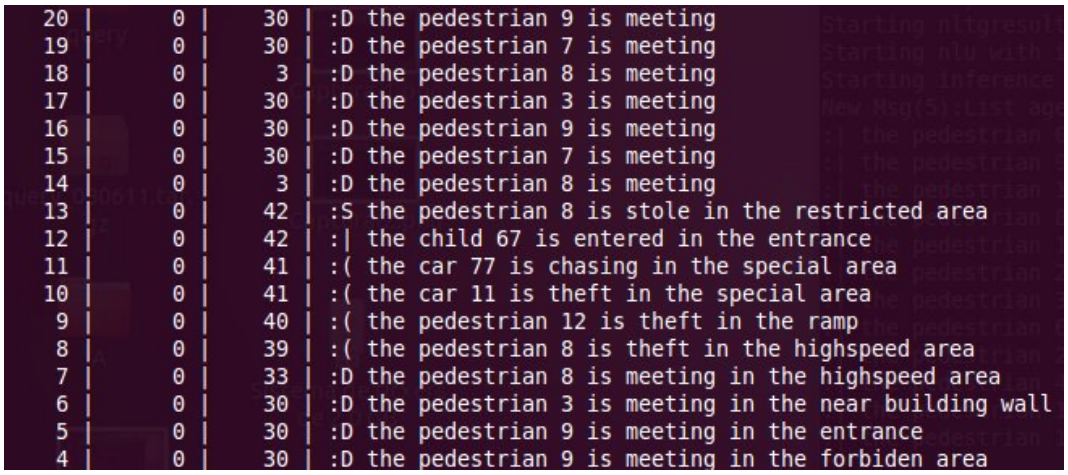


Figura 5.1: Las frases en lenguaje natural que el Avatar va diciendo, se van guardando en la base de datos.

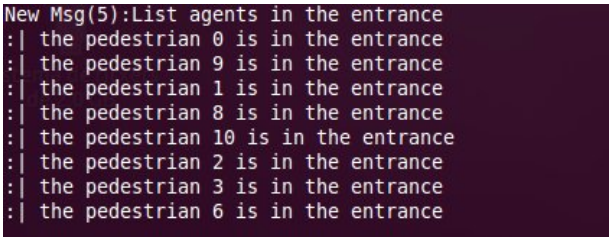
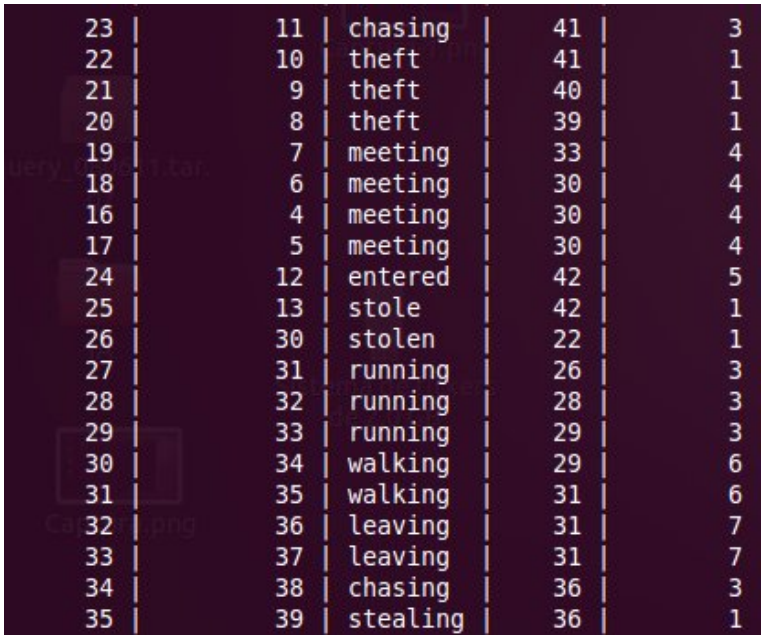


Figura 5.2: La aplicación recorre fila a fila la tabla donde se guardan las acciones y va mirando en ella cual cumple las condiciones solicitadas.

ya que se trata de otra consulta, pero en este caso es algo especial. Esto es debido a que la consulta pide que se haga un resumen de aquellas acciones o sucesos más importantes ocurridos en el día. Esto pone en marcha un sistema de reconocimiento de acciones y sobre todo del entendimiento de éstas. A la pregunta " ¿Qué es lo más importante sucedido hoy?, "What is the most important thing happened today?", el sistema deberá mirar en la base de datos, más en concreto en la tabla llamada de consciencia, donde las acciones están enumeradas por prioridades para determinar el grado de importancia de éstas.

El cuarto y último comportamiento es el denominado de consciencia, ya



23	11	chasing	41	3
22	10	theft	41	1
21	9	theft	40	1
20	8	theft	39	1
19	7	meeting	33	4
18	6	meeting	30	4
16	4	meeting	30	4
17	5	meeting	30	4
24	12	entered	42	5
25	13	stole	42	1
26	30	stolen	22	1
27	31	running	26	3
28	32	running	28	3
29	33	running	29	3
30	34	walking	29	6
31	35	walking	31	6
32	36	leaving	31	7
33	37	leaving	31	7
34	38	chasing	36	3
35	39	stealing	36	1

Figura 5.3: Las acciones númeradas con prioridad más altas serán tomadas como respuesta, hasta un máximo de diez.

```
New Msg(6):What is the most important thing happened today?
:8 There was a stolen in the frame 22
:S the pedestrian 3 is stolen in the forbiden area
:8 There was a stole in the frame 42
:S the pedestrian 8 is stole in the restricted area
:8 There was a stealing in the frame 36
:S the pedestrian 8 is stealing in the lowspeed area
:8 There was a theft in the frame 39
:( the pedestrian 8 is theft in the highspeed area
:8 There was a theft in the frame 40
:( the pedestrian 12 is theft in the ramp
:8 There was a theft in the frame 41
:( the car 11 is theft in the special area
:8 There was a running in the frame 29
:V the pedestrian 3 is running in the entrance
:8 There was a running in the frame 28
:V the pedestrian 8 is running in the special area
:8 There was a running in the frame 26
:V the pedestrian 3 is running in the lowspeed area
:8 There was a chasing in the frame 36
:( the pedestrian 90 is chasing in the forbiden area
```

Figura 5.4: Se observa como el sistema toma la acción que considera importante y el momento en que sucedió. Después, hace un reproducción del suceso al que hace referencia.

que se basa en el número de veces que se repite una acción durante un cierto momento del día. Así pues, si por ejemplo, la acción de robar se produce cada día en el mismo intervalo de tiempo, el sistema tomará como probable que esta acción se vuelva a repetir el siguiente día en el mismo momento aproximadamente. Por lo que, si esta iteración de acciones en el mismo intervalo supera un cierto número, el sistema nos alertará de la probabilidad del suceso.

Capítulo 6

Conclusión y trabajo futuro

6.1. Conclusión

Una vez finalizadas todas la tareas de búsqueda, programación y testeo de este trabajo, se da por completado el objetivo principal: Realizar un Avatar virtual con el que podamos comunicarnos mediante voz y texto. Además, de que éste sea capaz de ir diciéndonos aquello que va sucediendo en la calle, que sea capaz de atender a nuestas peticiones y realizarlas.

Hay que remarcar el hecho de haber utilizado herramientas de software libre, como XFace, Flite y Julius han facilitado mucho la realización de este proyecto, al margen de que se hayan tenido hacer modificaciones y adaptaciones en el código de éstas. Dichas modificaciones podrán ser usadas por otras instituciones y beneficiarse por ello.

Hemos visto como cabezas 3D son capaces de mostrar emociones y hablar utilizando el estándar MPEG-4 FA, que nos permite una independencia del modelo a la hora de reproducirlos en distintos reproductores MPEG-4. En el caso de XFace, es necesario definir las distintas emociones y visemes para cada modelo.

La limitación principal de este trabajo reside en el tiempo. Al ser un proyecto muy amplio es fácil desviarse del objetivo principal y centrarse únicamente en alguna de sus partes, intentando perfecionarla. La división del proyecto en módulos y a su vez en submódulos ha sido la mejor estrategia a seguir, ya que se puede fijar objetivos y ver sus resultados en un espacio de tiempo relativamente corto.

De todas maneras, al tener que tratar con diferentes lenguajes de programación, con diferentes formas de resolución y planteamientos, obliga a estar muy concentrado y tener una dedicación contínua para conseguir realizar este proyecto con éxito.

6.2. Trabajo futuro

El trabajar en este proyecto me ha permitido pensar sobre temas todavía desconocidos para la investigación. Temas que me han rondado durante mucho tiempo en la cabeza y que de una forma u otra he podido dar a luz. La mente humana es el gran desconocido para la humanidad, donde diferentes ciencias intentan descrifrar y desvelar sus interioridades. ¿Somos conscientes por qué pensamos? o ¿Pensamos por qué somos conscientes?. Otra pregunta que surge a raíz de las anteriores es, ¿únicamente piensan los humanos? y de ser así, ¿sería debido a que nuestros cerebros están más evolucionados?, por lo que nos haría pensar que ¿los seres menos evolucionados también piensan, pero a un nivel inferior?

Todo esto me lleva a la conclusión de que cuanto más evolucionados más nivel de pensamiento, por lo tanto, más consciencia. Entonces, esto implica que los seres con los algoritmos más complejos son lo más evolucionados. Si pudiéramos agrupar todos esos algoritmos que simulan el comportamiento humano, ¿estaríamos creando una mente digital?

Realmente, sería muy interesante, además de beneficioso para la humanidad, trabajar en esta linea. Continuar con este proyecto a un nivel más completo, con más tiempo y con más recursos, haría de éste un trabajo satisfactorio al mismo tiempo que estimulante y misterioso para uno mismo.

6.2.1. Otras vías de investigación

Quizá, la inteligencia humana pueda ser simulada e insertada en algún dispositivos de uso cotidiano, que actualmente nos rodean, dotándolos de ″vida″. ¿Y si en lugar de hacer una simulación, fuéramos capaces de acceder a los datos almacenados en el cerebro y transferirlos a un tipo de almacén digital? Realmente, el problema reside principalmente en como acceder a esos datos, más que en como almacenarlos.

La personalidad nos convierte en quienes somos, y de poder transferirla a otro lugar, ¿no estaríamos de alguna forma engañando a la muerte? Cierto es que este hecho nos convertiría en copia, pero, sería una copia perfecta en todos los aspectos. Si esa copia hablara como nosotros, pensara como nosotros, con los mismos gustos, con los mismos defectos, en definitiva que fuera como nosotros, ¿habría alguna diferencia con el ″yo″ original?. Una diferencia que no hace la diferencia, no es una diferencia.

Apéndice A

¿Sueñan los Avatares Virtuales con Ovejas Eléctricas ?

A.1. Planteamiento

¿Por qué los seres humanos rara vez son conscientes del increible ″milagro″ que es tener la capacidad de percibir el mundo que nos rodea? Podemos definir como consciencia, al conocimiento inmediato que el sujeto tiene de sí mismo, de sus actos y reflexiones. También como la capacidad de los ″seres humanos″ de verse y reconocerse a sí mismos y de juzgar sobre esa visión y reconocimiento.

¿Es consciente un bebé cuando nace? Puesto que un bebé humano nace con casi todas las conexiones neuronales en su corteza (es decir, el cerebro sigue siendo en gran medida una masa de células sin cables) ¿significa esto que un bebé no es consciente, ya que aún no puede crear significado de los estímulos sensoriales que recibe? La respuesta es no. Mientras que la mayor parte del cerebro está aún por comenzar el desarrollo, el cerebro inferior de un niño recién nacido ya está bien desarrollado y capaz de producir los comportamientos instintivos, tales como lactantes, el llanto e incluso el seguimiento de objetos con sus ojos, en respuesta a estímulos externos.

La base firme de la evolución de los vertebrados, durante millones de años, es la de transmitir los genes de la propia especie y asegurar que el tronco del encéfalo viene ″pre-cableado″ para ciertas funciones necesarias tales como la respiración. Así que el cerebro inferior proporciona una base estrecha, pero sólida, de supervivencia. ¿Cómo puedo demostrar que soy ″yo″ simplemente confiando en mis propias experiencias y recuerdos de ser yo mismo? Es evidente que la hipótesis se basa en las perspectiva de la consciencia en primera persona, es decir, lo que sé y la sensación y la experiencia acerca de mí mismo. Sin embargo, una máquina programada también tendría este mismo sentido de ser ″yo″ si tuviera ciertas experiencias añadidas en su memoria. Los enfoques subjetivos a los distintos estados de la conciencia se basan fundamentalmente en la interacción en primera persona con el conocimiento que viene desde el "yo". Por ejemplo, "Yo sé que soy yo, porque tengo mis recuerdos", o, "Sé que soy yo porque me siento como yo″ [24].

Normalmente, la sociedad no suele cuestionarse tales declaraciones como pruebas verificables de la identidad. Por ejemplo, si dos personas se reúnen después de muchos años, seguramente hayan cambiado mucho físicamente y esto hace que deban basarse en una memoria compartida de un suceso pasado para validar la verdad de la demanda de identidad de cada uno. Dado que esta información viene desde dentro de las personas (es decir, se basa en la memoria), y no puede ser observado abiertamente ni compartida por otras personas, es tomada como evidencia precisa y verificable de la identidad: "Yo sé que soy su amigo de la infancia, porque recuerdo que solía esconderse en un compartimiento secreto en el armario de la habitación de su madre".

De todas maneras, esta idea no es suficiente para verificar dicha identidad, ya que tanto el amigo o una versión emulada de él sostendrían en la memoria la misma experiencia de haberse escondido en el compartimiento secreto del armario y por tanto se sienten justificados al decir que ellos son

el amigo de la infancia de la persona que encontró pasados los años.

Como conclusión a esto, los enfoques subjetivistas basados en el conocimiento de la experiencia, emoción, memoria, etc, no son del todo adecuados para resolver la hipotética crisis de identidad.

A.2. Consciencia

¿Qué ventaja selectiva confiere la consciencia a quienes realmente la poseen, si es que únicamente la poseen ciertas especies? Se supone que esta «cosa» realmente «hace algo», y, además, que lo que hace es útil para la criatura que la posee. En el extremo opuesto, se podría pensar que la consciencia es simplemente un «algo» pasivo de la posesión de un sistema de control suficientemente elaborado , y que no «hace» nada realmente por sí sola. Aceptemos que la presencia de la consciencia en una criatura supone realmente una ventaja selectiva para la misma. ¿Cuál sería en concreto esta ventaja? Podría ser ventajoso para un predador que trata de imaginar lo que probablemente haría su presa a continuación. Imaginarse que él mismo es la presa podría darle una ventaja sobre ella. Si suponemos que la acción del cerebro humano, consciente o no, consiste simplemente en la ejecución de algún algoritmo muy complicado, entonces debemos preguntarnos cómo se formó un algoritmo de eficacia tan extraordinaria. Por lo que respecta a las criaturas con cerebros evolucionados, aquellos con los algoritmos más eficaces tendrían una mejor tendencia a sobrevivir [24].

A.3. Modelo Computacional del Cerebro Consciente

Hay muy pocas dudas sobre que el cerebro es el ″mecanismo de control″ del sistema nervioso humano. Controla la comunicación, la memoria, el movimiento, el aprendizaje y una serie de otras funciones que podrían ser análogas

en comparación con las funciones de un ordenador moderno. Por otra parte, no hay duda de que existe alguna similitud entre los complejos paquetes de neuronas de los seres humanos y las conexiones de los circuitos eléctricos de un ordenador, sobre todo ahora que las computadoras son capaces de complejos procesos paralelos. Las neuronas son también una forma de cableado eléctrico, donde la información pasa de dentro a fuera del cerebro por medio de impulsos electroquímicos que viajan a través de las uniones nerviosas, las sinapsis. O sea, que las neuronas cooperan de manera ordenada y mecánica, a raíz de todas las leyes deterministas de la física clásica. Estos flujos cerebrales interactúan para formar lo que se percibe como la realidad y la conciencia. Treisman[1] trabajó sobre la visión humana, en 1986, ilustrando el funcionamiento del cerebro en este campo.

Cuando vemos una manzana, uno es inmediatamente consciente de que el objeto que observamos es una manzana (que es de color rojo, redondo, que es bastante pequeño, etc.) Naturalmente, hay otros aspectos de nuestro funcionamiento consciente que comúnmente hacen referencia a las manzanas. Por ejemplo que puede satisfacer nuestro hambre, que una cayó en la cabeza de Newton, etc.

Sin embargo, esto no es parte de nuestra percepción visual. La percepción visual se relaciona con el tamaño de la manzana, la forma, orientación, color y posición.

Treisman señala que cada una de estas informaciones se registran por separado. En otras palabras, el cerebro no ve una manzana. Más bien, observa la redondez y enrojecimiento, etc, de ésta.
Cada característica observable es almacenada en un lugar diferente, a lo que se le llama las ″características del mapa″ y posteriormente en un ″mapa maestro de los lugares″.

[1] Anne Marie Treisman, es una psicóloga del Departamento de Psicología de la Universidad de Princeton. Ella investiga la atención visual, la percepción de objetos, y la memoria. Una de sus obras más influyentes es la teoría de la integración de funciones de atención, publicado por primera vez con G. Gelade en 1980.

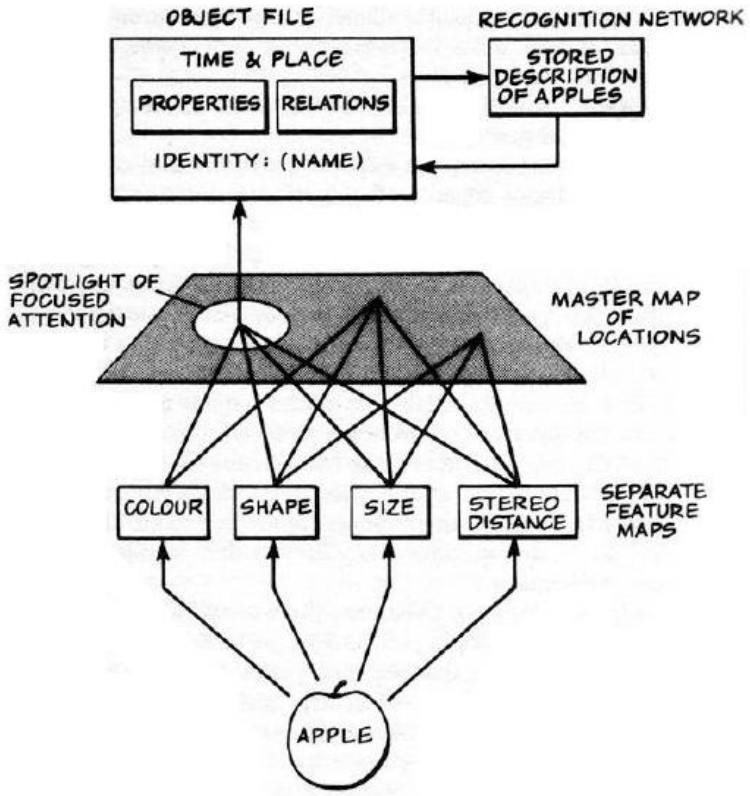


Figura A.1: Esquema del conocimiento.

Una vez el mapa maestro ha sido completado, es analizado y el cerebro ve una manzana a partir de todos estos elementos y características.

A.4. Inteligencia Artificial Fuerte

Con frecuencia utilizamos ciertos términos descriptivos que humanizan el comportamiento de las máquinas. «Mi coche no quería arrancar»; «El televisor ha decidido apagarse»; o « El ascensor no quería abrir las puertas»; Por supuesto, que no queremos decir que el coche quiera algo de verdad, que el televisor decidiera hacer o no tal cosa o que el ascensor esté enfadado y se niegue a hacer tal otra cosa. Es evidente que las reivindicaciones actuales sobre la Inteligencia Artificial, van mucho más allá de las modestas expecta-

tivas de los primeros teóricos de la IA, como Alan Turing[2] y Herbert Simon[3].

Tales reivindicaciones sugieren que las computadoras serán capaces de emular y recrear con precisión la inteligencia humana, e incluso ir más allá de una recreación o emulación de la misma. Este posicionamiento sobre estas cuestiones es lo que se conoce como IA fuerte, o Inteligencia Artificial fuerte. Según esta tendencia, no solo los dispositivos que acabamos de mencionar serían inteligentes y tendrían mentes, sino que se puede atribuir un cierto tipo de cualidades mentales al funcionamiento lógico de cualquier dispositivo computacional, incluso los dispositivos mecánicos simples tales como un termostato. La idea es que la actividad mental consiste simplemente en llevar a cabo alguna secuencia de operaciones, frecuentemente llamadas algoritmo. Para cualquier tipo importante de actividad mental de un cerebro humano, el algoritmo tendría que ser muchísimo más complicado, pero, según el punto de vista de la IA fuerte, un algoritmo en cualquier caso.

Por lo tanto, todas las cualidades mentales como el pensamiento, el sentimiento, la inteligencia, la comprensión, la consciencia, deben ser consideradas, según este punto de vista, simplemente como aspectos de este funcionamiento complicado; es decir, son simplemente características del algoritmo a ejecutar por el cerebro.

Se han expresado muchas opiniones diferentes con relación entre el estado del cerebro y el fenómeno de la consciencia. Es notable el escaso concenso de opinión para un fenómeno de tan obvia importancia. Es evidente que no todas las partes del cerebro están involucradas por igual en su manifesta-

[2]Alan Mathison Turing, fue un matemático, informático teórico, criptógrafo y filósofo inglés. Considerado uno de los padres de la Ciencia de la Computación siendo el precursor de la informática moderna.

[3]Herbert Alexander Simon, economista, politólogo y teórico de las ciencias sociales estadounidense. Premio Nobel de Economía por ser uno de los investigadores más importantes en el terreno interdisciplinario y porque su trabajo ha contribuido a racionalizar el proceso de toma de decisiones.

ción. Las acciones bajo control cerebelar parecen tener lugar casi de forma «autómata» sin que tengamos que «reflexionar» sobre ellas. Mientras que podemos decidir andar de un lugar a otro, no tenemos consciencia a menudo de los elaborados planes de los movimientos musculares que se deben realizar para tal desplazamiento.

Según los partidarios de la IA fuerte, si la «consciencia» es simplemente una característica de la complejidad de un algoritmo, entonces, los algoritmos complicados que ejecuta la corteza cerebral confirmaría a esta región como la más firme candidata a la capacidad de manifestar consciencia.

El credo de la IA fuerte se basa esencialmente en dos puntos: El primero es que con el tiempo es posible capturar todos los aspectos de la inteligencia humana en un soporte informático, y en segundo término que la mente humana es, a todos los efectos, como un gran programa de ordenador.

Esto nos sugiere que las funciones de la mente humana están contenidas dentro de un equipo bioquímico muy complejo. Por desgracia, esta afirmación no es científicamente verificable, ya que la tecnología para medir y entender el funcionamiento de la mente humana aún no existe.

Tales afirmaciones son teóricas y de naturaleza filosóficas siendo suposiciones acerca de la mente humana. Dichas afirmaciones son conocidas como ″Metafísica″, la cual se encarga de estudiar la naturaleza, estructura, componentes y principios fundamentales de la realidad. La IA fuerte afirma que en el futuro los computadores no sólo serán capaces de igualar la inteligencia humana, sino que la superará. Los supuestos básicos de Kurzweil[4] encajan

[4]Raymond Kurzweil, es un inventor estadounidense, además de músico, empresario, escritor y científico especializado en Ciencias de la Computación e Inteligencia Artificial. Experto tecnólogo de sistemas y de Inteligencia Artificial y eminente futurista. Actualmente presidente de la empresa informática Kurzweil Technologies, que se dedica a elaborar dispositivos electrónicos de conversación máquina-humano y aplicaciones para discapacitados y es canciller e impulsor de la Universidad de la Singularidad de Silicon Valley.

en lo anterior comentado.

En particular, Kurzweil señala el hecho común de que las computadoras ya superan la inteligencia humana en muchas áreas, tales como el cálculo matemático, la predicción basada en la consideración de una amplia gama de variables y escenarios (por ejemplo, la predicción del tiempo, y el comercio de acciones).

Sin embargo, reconoce que la verdadera inteligencia artificial no es posible todavía, pero sugiere que la evolución tecnológica del hardware y software significará que esta disparidad entre la inteligencia humana y la inteligencia computacional llegará a desaparecer.

A.5. Test de Turing y la Habitación China de Searle

¿Puede una máquina pensar en la forma que lo hacen los seres humanos? Con el fin de encontrar un resultado a su pregunta, Turing propuso que un computador debe ser capaz de interactuar con las personas de la misma manera que los humanos lo hacen entre sí.

Esta prueba recibe el nombre de *el Test de Turing* o *el juego de la imitación*. A un lado de una pared se coloca un interrogador humano delante de un terminal y al otro lado de ésta, en otro terminal se encuentra un humano, o una computadora. El interrogador humano realiza diferentes tipos de preguntas a quien está al otro lado de la pared. Las respuestas aparecen en el terminal frente al interrogador humano. Si las respuestas, del humano o del computador, no tienen errores bajo el punto de vista del interrogador, se puede decir que se ha pasado el Test de Turing.

El argumento filosófico anterior, muestra de una manera convincente que los seres humanos no sólo son inteligentes, sino que lo que hace que la inteli-

Figura A.2: Test de Turing.

gencia humana sea efectiva es su relación entre la conciencia y la comprensión. La cuestión que debemos considerar es si un éxito de este tipo indica realmente una auténtica comprensión por parte del computador, o quizá, por parte del propio programa.

Un punto de vista ampliamente más aceptado y más convincente, lo argumentó John Searle[5] en *La habitación China*.
En primer lugar, él imagina que las historias se cuentan en chino en lugar de castellano y que todas las operaciones del algoritmo del ordenador para este ejercicio concreto se suministran (en castellano) como conjunto de instrucciones para manipular fichas con símbolos chinos en ellas. Searle se imagina a sí mismo haciendo todas las manipulaciones en el interior de una habitación cerrada.

Las secuencias de símbolos que representan las historias, y luego las preguntas, se introducen en la habitación a través de una pequeña ranura. No se permite ninguna otra información del exterior. Finalmente, cuando se han

[5]John Rogers Searle, es Slusser Profesor de Filosofía en la Universidad de California, Berkeley, y es célebre por sus contribuciones a la filosofía del lenguaje, a la filosofía de la mente y de la consciencia, a las características de las realidades sociales frente a las realidades físicas, y al razonamiento práctico.

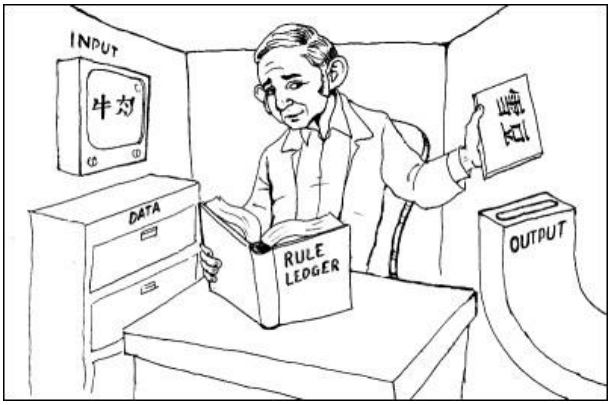


Figura A.3: La habitación China de Searle.

completado todas las manipulaciones, la secuencia resultante se extrae de nuevo a través de la ranura. Puesto que todas estas manipulaciones están simplemente ejecutando el algoritmo de Schank[6], debe ocurrir que esta secuencia final resultante es simplemente el equivalente chino para «si» o «no», según sea el caso, que da la respuesta correcta a la pregunta original en chino acerca de una historia en chino.

Entonces Searle deja claro que él no entiende ni una sola palabra de chino, de modo que no tiene la más remota idea de lo que cuentan las historias. De todas formas, llevando a cabo correctamente la serie de operaciones que constituyen el algoritmo de Schank (las instrucciones para este algoritmo le han sido dadas en castellano) él sería capaz de contestar tan bien como lo haría una persona china que realmente entendiera las historias. El punto importante de Searle es que la mera ejecución de un algoritmo correcto no implica en sí mismo que haya tenido lugar ninguna comprensión.

Decir que un ordenador no es realmente inteligente porque no entiende la entrada y la salida de las historias citadas anteriormente, es decir lo mismo

[6]Roger Schank, es el presidente y consejero delegado de Socratic Arts y ha trabajado en Inteligencia Artificial. En conjunto con la Universidad La Salle de Barcelona, Schank creó el instituto de Ciencias del Aprendizaje en conjunto con la Escuela de Negocios de Ingeniería.

sobre el hombre de la habitación china. Tampoco sería inteligente, en este caso, ya que él tampoco comprende las entrada ni las salidas en chino.

Apéndice B

Formato de los ficheros del toolkit XFace

B.1. Estándard FA de MPEG-4

En 1999, el Moving Pictures Experts Group (MPEG, Grupo de Expertos en Imágenes en Movimento) publicó el estándard MPEG-4 (ISO/IEC JTC1/WG11 N1901 y N1902), el cual abarcaba un amplio espectro de temas multimedia, como por ejemplo audio y vídeos naturales y sintetizados o gráfics en 2D y 3D. Al contrario que anteriores estándars de MPEG, que se centraban en la codificación eficiente de diferentes contenidos, MPEG-4 trata sobre todo de la comunicación y la integración de éstos. Hoy en día es el único estándard que contempla animación facial, y ha sido ampliamente aceptado en el entorno académico, al mismo tiempo que también comienza a llamar la atención de la indústria.

MPEG-4 Facial Animation (FA) describe los pasos para crear un agente parlante mediante la definición estandarizada de unos ciertos parámetros necesarios. La creación de éstos se divide principalmente en dos fases, tratadas por separado en el estándard: la localización de los puntos característicos en el modelo 3D estático, cosa que define las regiones de deformación en la cara,

y la generación y la interpretación de los parámetros que modificarán estos puntos característicos para generar la animación en sí.

Con el fin de crear una cara conforme al estándard, MPEG-4 define una serie de parámetres llamados FDP (Facial Definition Parameters, Parámetros de Definición Facial), de los cuales los más importantes son los 84 FPs (Feature Points, puntos característicos) localizados en el modelo de una cabeza. Estos se deberían de definir para cada modelo 3D estático (indicar a que vértices concretos corresponden) que se quiera hacer conforme al estándard y su función es describir la forma de una cabeza ″normal″ en ″posición de reposo″, la cual se asume siempre como la posición de partida de cualquier animación y está definida por el estándard como las siguientes características: todos los músculos relajados, párpados tangentes al iris, las pupilas abiertas hasta un tercio del diámetro del iris, los labios y los dientes en contacto y la lengua plana com la punta tocando los dientes. Estos puntos característicos se utilizan para definir parámetros de la animación así como para calibrar los modelos cuando se abren en reproductores diferentes.

B.2. Lista de comandas

1: List agents
2: List agents between frames XX and XX
3: List agents meeting
4: List agents meeting between frames XX and XX
5: List agents in the entrance
6: List agents in the entrance between frames XX and XX
7: What is the most important thing happened today? (Activa Consciousness)
8: List pedestrians in the scene
9: List pedestrians in the scene between frames XX and XX
10: List pedestrians meeting

11: List pedestrians meeting between frames XX and XX
12: List cars in the scene
13: List cars in the scene between frames XX and XX
14: List pedestrians running in the restricted area
15: List pedestrians running in the restricted area between frames XX and XX
16: Show me agents meeting in the ramp
17: Show me agents entering in the ramp between frames XX and XX
18: Have you seen any theft?
19: Have you seen any theft between frames XX and XX?
20: Have you seen pedestrians meeting in the in out area?
21: How many agents have you seen in the scene?
22: How many pedestrians have stolen in the scene?
23: How many pedestrians have entered by the forbiden area?
24: How many pedestrians have entered by the forbiden area between frames XX and XX?
25: How many pedestrians have crossed the lowspeed area?
26: How many pedestrians have walked by the main floor?
27: How many times does the agent X walk?
28: How many times does the agent X walk between frames XX and XX?
29: How many times does the agent X walk in the near building wall?
30: How many times does the agent X walk in the near building wall between frames XX and XX?
31: How many people are there in the scene between frames XX and XX?
32: How many people met in the normal area?
33: Have you seen somebody running by the highspeed area between frames XX and XX?
34: Has there been any theft?
35: Has there been any chase between frames XX and XX?
36: Has there been somebody running by the entrance?
37: Has there been somebody running by the entrance between frames XX

and XX?
38: Has there been anybody?
39: Has there been person between frames XX and XX?
40: Has somebody met in the bike parking?
41: List locations where a pedestrian has walked by
42: List pedestrians's actions between frames XX and XX
43: List agentX's actions in the main floor
44: List agentX's actions in the main floor between frames XX and XX
45: List actions that have happened in the bike parking
46: List actions that have happened in the bike parking between frames XX and XX
47: List the situations of a pedestrian in the ramp
48: List the situations of a car in the ramp
49: List locations in the scene
50: How many people chases after a theft?
51: How many people pick up abandoned objects?
52: How many people are after a theft?
53: How many people are there after a meeting in the entrance
54: How many people leave an object before the meeting?
55: How many people have left an object before the chase?
56: Which agents are there in the scene before a theft?
57: Which vehicles are there in the scene after a meeting?
58: What happens before a theft?
59: Which agents are there in the scene during a theft?
60: Has a chase happened after a theft

B.3. Ejemplo de fichero FDP de Xface

A continuación, se da como ejemplo un archivo de configuració de una cabeza para XFace que define al Avatar femenino llamado ″Alice″.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"? >
<xfdp>
<!–XFaceEd generated MPEG-4 FDP specification file–>
<!–Header information for the model–>
<head>
<!–Version of the FDP specification file (this file)–>
<file version="0.2/>
```

Cuadro B.1: Ejemplo de archivo de configuración de XFace FDP.

```
<!–FAPU (Facial Animation Parameter Units) for the model–>
<fapu ENS0="51" ES0="69" IRISD0="16" MNS0="30" MW0="50"/>
<!–Global Translation info for the whole face–>
<translation x="0" y="-1" z="-659"/>
<!–Global Rotation info for the whole face–>
<rotation axis angle="0.407589" axis x="-0.999632"
axis y="-0.0154467" axis z="0.0223226"/>
</head>
<!–3D model files (mesh) configuration–>
```

Cuadro B.2: Definición de las FAPUs.

```
<source>
<entity alias="Anger" category="Expression" >
<mesh file="alice ExpressionAnger.wrl" format="wrl"/>
</entity>
<entity alias="BlinkEyes" category="Modifier" >
<mesh file="alice ModifierBlinkEyes.wrl" format="wrl"/>
</entity>
<entity alias="Disgust" category="Expression" >
<mesh file="alice ExpressionDisgust.wrl" format="wrl"/>
</entity>
<entity alias="Fear" category="Expression" >
<mesh file="alice ExpressionFear.wrl" format="wrl"/>
</entity>
<entity alias="LookDown" category="Modifier" >
<mesh file="alice ModifierLookDown.wrl" format="wrl"/>
</entity>
<entity alias="LookLeft" category="Modifier" >
<mesh file="alice ModifierLookLeft.wrl" format="wrl"/>
</entity>
<entity alias="LookRight" category="Modifier" >
<mesh file="alice ModifierLookRight.wrl" format="wrl"/>
</entity>
<entity alias="LookUp" category="Modifier" >
<mesh file="alice ModifierLookUp.wrl" format="wrl"/>
</entity>
```

Cuadro B.3: Especificación de los modelos 3D para cada *keyframe*.

```
<entity alias="Rest" category="Expression" >
<mesh file="alice.wrl" format="wrl"/>
<bind item="Hair" submesh="alice.wrl 0"/>
<bind item="LeftEye" submesh="alice.wrl 1"/>
<bind item="RightEye" submesh="alice.wrl 2"/>
<bind item="LowerTeeth" submesh="alice.wrl 5"/>
<bind item="UpperTeeth" submesh="alice.wrl 6"/>
<bind item="Tongue" submesh="alice.wrl 7"/>
</entity>
```

Cuadro B.4: Especificación de los modelos 3D básicos para formar la cabeza en posición de descanso.

```
<entity alias="Sad" category="Expression" >
<mesh file="alice ExpressionSad.wrl" format="wrl"/>
</entity>
<entity alias="SmileClosed" category="Expression" >
<mesh file="alice ExpressionSmileClosed.wrl" format="wrl"/>
</entity>
<entity alias="SmileOpen" category="Expression" >
<mesh file="alice ExpressionSmileOpen.wrl" format="wrl"/>
</entity>
<entity alias="Surprise" category="Expression" >
<mesh file=.ªlice ExpressionSurprise.wrl" format="wrl"/>
</entity>
<entity alias="aah" category="Viseme" >
<mesh file="alice Phonemeaah.wrl" format="wrl"/>
</entity>
<entity alias="b" category="Viseme" >
<mesh file="alice PhonemeB,M,P.wrl" format="wrl"/>
</entity>
<entity alias="bigaah" category="Viseme" >
<mesh file="alice Phonemebigaah.wrl" format="wrl"/>
</entity>
<entity alias="ch" category="Viseme" >
<mesh file="alice Phonemech,J,sh.wrl" format="wrl"/>
</entity>
<entity alias="d" category="Viseme" >
<mesh file="alice PhonemeD,S,T.wrl" format="wrl"/>
</entity>
<entity alias="ee" category="Viseme" >
<mesh file="alice Phonemeee.wrl" format="wrl"/>
</entity>
<entity alias="eh" category="Viseme" >
<mesh file="alice Phonemeeh.wrl" format="wrl"/>
</entity>
<entity alias="f" category="Viseme" >
<mesh file="alice PhonemeF,V.wrl" format="wrl"/>
</entity>
<entity alias="i" category="Viseme" >
<mesh file="alice Phonemei.wrl" format="wrl"/>
</entity>
<entity alias="k" category="Viseme" >
<mesh file="alice PhonemeK.wrl" format="wrl"/>
</entity>
<entity alias="n" category="Viseme" >
<mesh file="alice PhonemeN.wrl" format="wrl"/>
</entity>
<entity alias="oh" category="Viseme" >
<mesh file="alice Phonemeoh.wrl" format="wrl"/>
</entity>
</source>
```

<fdp affects="alice.wrl 3" index="2395" name="2.1" >
<indices>0 1 2 78 104 176 177 178 259 282 407 408 409 410 411
412 413 414 415 416 417 442 445 446 447 449 478 499 506 782 783
788 976 977 978 979 980 981 982 983 984 1008 1009 1034 1053 1302
1303 1308 1439 1440 1441 1442 1609 1610 1611 1677 1678 1679 1835
1836 1837 1838 2024 2025 2026 2090 2091 2092 2304 2330 2331 2332
2333 2334 2335 2336 2337 2338 2339 2340 2341 2342 2343 2344 2345
2346 2347 2348 2349 2350 2384 2386 2387 2388 2393 2394 2395 2396
2397 2398 2399 2400 2401 2402 2403 2453 2494 2496 2497 2510 2511
2512 3194 3196 3197 3326 3327 3328 3329 3330 3331 3332 3343 3344
3352 3358 3359 3360 3361 3362 3363 3396 3397 3398 3399 3406 3407
3603 3605 3612 3619 3620 3621 3647 4139 4140 4141 4142 4143 4144
4145 4146 4147 4148 4149 4150 4151 4152 4153 4154 4155 4156 4157
4158 4191 4192 4193 4199 4200 4201 4202 4203 4204 4205 4206 4207
4252 4294 4296 4297 4310 4312 4956 4957 4958 5087 5088 5089 5090
5091 5092 5093 5104 5109 5110 5112 5118 5119 5120 5121 5122 5123
5155 5156 5157 5158 5165 5360 5361 5369 5375 5376 5377 5403 </indices>
<influence fap="2" type="RaisedCosInfluenceSph" weight="0.3"/>
</fdp>

Cuadro B.6: Función de deformación a aplicar con los pesos. Los vértices del modelo se dan indexados.

Bibliografía

[1] M. Vallez, R. Pedraza-Jimenez , "El procesamiento del Lenguaje Natural en la Recuperación de Información Textual y áreas afines", Hipertext.net, no 5 (mayo 2007). Web: http://www.hipertext.net/web/pag277.htm

[2] World Wide Web Consortium. Oficina Española. Guía Breve de Interacción Multimodal. Darrera actualització: 13 de febrer del 2006.
http://www.w3c.es/Divulgacion/Guiasbreves/Multimodalidad

[3] Eric Keller (editor) i altres, Universitat de Lausanne. Fundamentals of Speech Synthesisand Speech Recognition. John Wiley and Sons, 1995.

[4] Centre de Visió per Computador. Human-Expressive Representations of Motion and their Evaluation in Sequences (HERMES Project). IST 027110. Darrera actualització: 2 de Juny del 2006.
http://www.hermes-project.eu/

[5] ITC-irst. XFace. Darrera actualització: 7 de octubre del 2005.
http://xface.itc.it/

[6] G. Erozel, N. K. Cicekli, and I. Cicekli. Natural language querying for video databases. Inf. Sci., 178(12):2534–2552, 2008.

[7] J. M. Soto Corzo, D. Díaz Portillo, J. A. Cruz Zamora , "Sistema de Consultas en Lenguaje Natural para Bases de Datos", Pro-

ceedings of the LoLaCOM06 Workshop, Instituto Tecnológico de Apizaco, Apizaco, Tlaxcala, México, 13th - 14th November 2006, volumen 220, 2006

[8] R. Beneyto, "Implantar el lenguaje Natural", Mètode: anuario, No. 2004, 2004, pags. 165-168. Web: http://www.uv.es/metode/anuario2004/165-2004.htm

[9] S. Balari, "El procesamiento del lenguaje natural como problema computacional", Quark: Ciencia, medicina, comunicación y cultura, pags. 35-43, No 19, 2000.

[10] ELF Software Co., Web: http://www.elfsoft.com/ns/index.htm

[11] Semantra, Web: http://www.semantra.com/

[12] S. Balari, "El procesamiento del lenguaje natural como problema computacional", Quark: Ciencia, medicina, comunicación y cultura, pags. 35-43, No 19, 2000.

[13] W. Wahlster, "Verbmobil: foundations of speech-to-speech translation", Wolfgang Wahlster (Ed.), 2000.

[14] ¿Qué es Procesamiento del Lenguaje Natural?, Web: http://www.cicling.org/ampln/NLP.htm

[15] Conversational Interaction and Spoken Dialogue Research Group, Web: http://www.cs.rochester.edu/research/cisd/

[16] Annalisa Guerzoni. Web de la divisió TCC de l'ITC-irst. http://tcc.itc.it/

[17] Koray Balci, Xface: MPEG4Based Open Source Toolkit for 3D Facial Animation. ITC-irst, 2004.

[18] Koray Balci, Xface: Open Source Toolkit for Creating 3D Faces of an Embodied Conversational Agent. ITC-irst, 2005.

[19] N. Kojekinem V. Savchenko, M. Senin, I. Hagiwara, Real-time 3D Deformations by Means of Compactly Supported Radial Basis Functions, "Proc. of Eurographics02", pàgines 35 a 43, setembre del 2002.

[20] Flite Home. Speech Software at CMU. http://www.speech.cs.cmu.edu/flite/

[21] Julius Home. http://sourceforge.jp/projects/julius/downloads/47534/Juliusbook-4.1.5.pdf/

[22] Carles Fernández Tena. Universitat Autònoma de Barcelona. Understanding Image Sequences: the Role of Ontologies in Cognitive Vision.

[23] http://en.wikipedia.org/wiki/Natural language generation.

[24] Roger Penrose. "La nueva mente del emperador", título original "The Emperor's New Mind", octubre 2006

Resum

Creació d'una plataforma que permeti comunicar-se amb una Avatar virtual. Aquest haurà d'informar sobre allò que succeeix en una zona del carrer, atenent a consultes relacionades amb allò que va observant. El coneixement de l'Avatar es basarà en les seves observacions que convertirà a llenguatge natural i enmagatzemarà en una base de dades. En el transcurs d'aquest document es descriuen els métodes i els móduls utilitzats i les característiques de les eines necessitades. Finalment, es presentan els resultats experimentals que demostren l'eficàcia de la plataforma final.

Resumen

Creación de una plataforma que permita comunicarse con un Avatar virtual. Éste deberá informar sobre aquello que sucede en una zona de la calle, atendiendo a consultas relacionadas con aquello que va observando. El conocimiento del Avatar se basará en sus observaciones que convertirá a lenguaje natural y almacenará en una base de datos. A lo largo del documento se describen los métodos y módulos utilizados y las características de las herramientas necesitadas. Finalmente, se presentan los resultados experimentales que demuestran la eficacia de la plataforma final.

Abstract

Creating a platform to communicate with a virtual Avatar. It should report on what happens in an area of the street response to questions related to what is observed. Knowledge of Avatar is based on observations that will convert natural language and stored in a database. Throughout the paper describes the methods and modules used and the characteristics of the tools needed. Finally, we present experimental results that demonstrate the effectiveness of the final platform.