17

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CONTADURÍA Y ADMINISTRACIÓN

MODELO JERÁRQUICO PARA INTEGRAR MÚLTIPLES FUENTES DE CONOCIMIENTO LINGÜÍSTICO EN UNA BASE DE DATOS SEMIESTRUCTURADA

TESIS PROFESIONAL QUE PARA OBTENER

EL TÍTULO DE:

LICENCIADO EN INFORMÁTICA

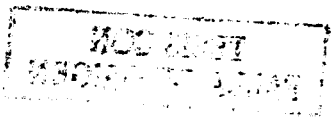
P R E S E N T A:

ROXANA IRENE PHILLIPS MACÍAS

ASESORA:

M.C. ESMERALDA URAGA SERRATOS

MÉXICO, D.F. 2002



Agradecimientos

A mi asesora de tesis M.C. Esmeralda Uraga por su apoyo y gran paciencia durante el desarrollo de este trabajo.

A los integrantes del área de inteligencia artificial por sus críticas, consejos y comentarios: Paulino Ochoa, César Gamboa, Javier Cuétara, Iván Meza, Arturo Espinosa y Luis Pineda.

A toda mi familia, especialmente a mi mamá por su apoyo.

A Juan por levantarme el ánimo y por estar a mi lado.

A mis amigos, por el gusto de saber que se preocupan por mí.

Al CONACYT (convenio 27948-A) por el apoyo en el desarrollo de este trabajo.

Con cariño a la Universadad Nacional Autónoma de México.

Índice General

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción

1.1 Antecedentes

La inteligencia artificial tiene por objetivo resolver problemas y tomar decisiones similares a las que los humanos afrontan cotidianamente; es decir, busca explicar y emular el comportamiento inteligente en términos de procesos computacionales. Para esto se han desarrollado los llamados sistemas inteligentes; estos sistemas tratan de modelar algunos procesos del pensamiento humano y pueden servir como una extensión de nuestras habilidades creativas y de resolución de problemas. Los sistemas inteligentes cuentan con dos características primordiales [Bielawski et al., 1991]: la capacidad de usar conocimiento para realizar ciertas tareas o para solucionar problemas, y la capacidad de explotar los poderes de la asociación y de la deducción; es decir, procuran llegar a un resultado a través del razonamiento con la intención de tratar problemas complejos que se parezcan al mundo real.

Algunos de los ejemplos de sistemas inteligentes que se encuentran hoy en día son los juegos de ajedrez de computadora, sistemas traductores de lenguaje natural, aplicaciones robóticas, sistemas de reconocimiento de voz y de visión, sistemas expertos, sistemas de síntesis de voz y sistemas de lenguaje hablado (sistemas conversacionales), etc. Todos estos sistemas ofrecen modelos de algunos procesos de la mente humana; por un lado intentan emular la inteligencia humana; por otro manipulan extensas cantidades de información, y finalmente ofrecen un cierto grado de capacidad para solucionar problemas.

La comunicación humano-computadora por medio del lenguaje hablado se ha convertido en el objetivo común de muchos trabajos de investigación hoy en día. Se han hecho grandes progresos en varias ramas de esta área de investigación en los últimos años, principalmente en síntesis de voz, reconocimiento de voz, entendimiento del lenguaje natural y en generación del habla [Mei-Ling, 1995]. Los sintetizadores de voz son aquellos sistemas que reproducen la voz del ser humano

por medio de una computadora; estos sistemas han alcanzado un alto grado de claridad y naturalidad. Los sistemas de reconocimiento de voz son capaces de reconocer vocabularios grandes independientemente del hablante. Los sistemas de entendimiento de lenguaje natural, pueden analizar y reconocer enunciados con la intención de obtener su significado; la recuperación de información interactiva por medio de voz, requiere del componente de generación de lenguaje para que pueda responder al sistema.

La combinación de las aplicaciones de estas cuatro áreas (síntesis de voz, reconocimiento de voz, generación de lenguaje y los sistemas para el entendimiento de lenguaje natural) ha permitido la creación de sistemas conversacionales. Los sistemas conversacionales permiten la comunicación y la interacción humano-computadora. Estos sistemas deben de ser capaces de reconocer la voz de una persona, interpretar la secuencia de palabras dentro de alguna aplicación para obtener un significado y proveer de esta forma una respuesta apropiada.

Los sistemas conversacionales pueden establecer un diálogo con un usuario de temas relacionados, por lo regular, en un dominio específico. Estos sistemas permiten al usuario resolver problemas dentro del dominio designado; realizar mensajes hablados en otro idioma a través de la traducción por computadora, y ayudar a aprender a leer. [Mei-Ling, 1995]. Cuando madure esta tecnología, la utilización de los sistemas conversacionales se podrá extender desde la oferta de servicios telefónicos hasta asistencias a personas discapacitadas.

Los sistemas interactivos humano-computadora facilitan la interacción entre ambos y crean un ambiente amigable. Las computadoras con el poder de hablar y escuchar hacen que no sea necesario utilizar las manos ni la vista, ya que el medio oral puede proveer un eficiente y económico modo de comunicación.

Las areas de conocimiento relacionadas con los sistemas conversacionales forman parte del campo de la lingüística computacional. Esta disciplina consiste en el estudio científico del lenguaje desde una perspectiva computacional y considera como uno de sus objetivos principales el modelar computacionalmente el lenguaje natural humano.

Los modelos otenidos benefician la elaboración de los sistemas conversacionales. Es posible crear cada vez mejores sistemas computacionales de síntesis, reconocimiento de voz e interpretación del lenguaje humano debido a que estos modelos permiten entender mejor el funcionamiento del lenguaje humano.

Para desarrollar sistemas conversacionales es necesaria la interpretación correcta de la entrada (lenguaje oral) y la generación exacta de la salida (voz sintetizada). El descifrar la semántica incluida en el mensaje de una señal acústica o codificar un mensaje en un discurso sintetizado, implica la utilización de diversas fuentes de conocimiento lingüístico [Mei-Ling, 1995].

Entre las áreas de conocimiento lingüístico [Mei-Ling, 1995] [Mungía et al., 1998] se encuentran las siguientes :

- **Fonética.** Estudia los sonidos de una lengua. Analiza la realización física de los sonidos, es decir, cómo se producen, cómo se perciben y cómo están formadas las ondas sonoras del habla.
- **Fonología.** Se ocupa del estudio de los sonidos, no como realizaciones físicas, sino como representaciones que permiten establecer diferencias de significado. Se interesa en analizar si el cambio de un sonido en una secuencia de sonidos provoca cambio de significado.
- **Morfología.** Se encarga del estudio de las palabras, su estructura interna y los procesos de su formación, así como de las modificaciones que sufren para indicar los distintos accidentes gramaticales de género, número, tiempo, modo, etc.
- **Lexicología.** Se ocupa del estudio de las palabras de una lengua y de las relaciones sistemáticas que se establecen entre ellas.
- **Sintaxis.** Estudia cómo ordenar, coordinar y subordinar las palabras, así como las relaciones que guardan éstas dentro de una oración.
- **Pragmática.** Es el estudio del lenguaje en su relación con las personas y el contexto de la conversación.
- **Discurso.** Estudio de las ideas que se expresan por medio de frases y oraciones que se suceden de manera continua.

Diferentes áreas de conocimiento lingüístico interactúan entre sí para modificar la señal de voz. La pronunciación de las palabras, que es la preocupación principal de los sistemas de síntesis y de reconocimiento de voz, puede ser influenciada por la morfología, la sintaxis, la semántica y el discurso de las palabras. Por ejemplo la semántica se necesita para conocer el significado de las palabras y de las oraciones, y para evitar ambigüedades de palabras que se pronuncian igual pero que se escriben diferente (homófonos), como "baya", "vaya" y "valla"; otro ejemplo, en el caso de la fonética y la fonología, es acerca del estudio que éstas hacen de la pronunciación de los fonemas ya que una misma palabra puede ser pronunciada de distinta forma; tal es el caso de la palabra "un", en la la frase "un bote", que puede ser pronunciada como *um* o *un*.

Estas fuentes de conocimiento lingüístico estrechamente relacionadas, son indispensables en el desarrollo de sistemas conversacionales, ya sea en la parte de síntesis, reconocimiento o en el entendimiento del habla. Generalmente los distintos tipos de información (conocimiento lingüístico) son incorporados de forma independiente dentro de los componentes de sistemas conversacionales ya existentes, en lugar de modelar sus correlaciones en una estructura integrada. La estructura resultante podrá facilitar la utilización concurrente de las fuentes de conocimiento lingüístico y podrá ser aplicable en síntesis, reconocimiento y en el entendimiento del lenguaje hablado.

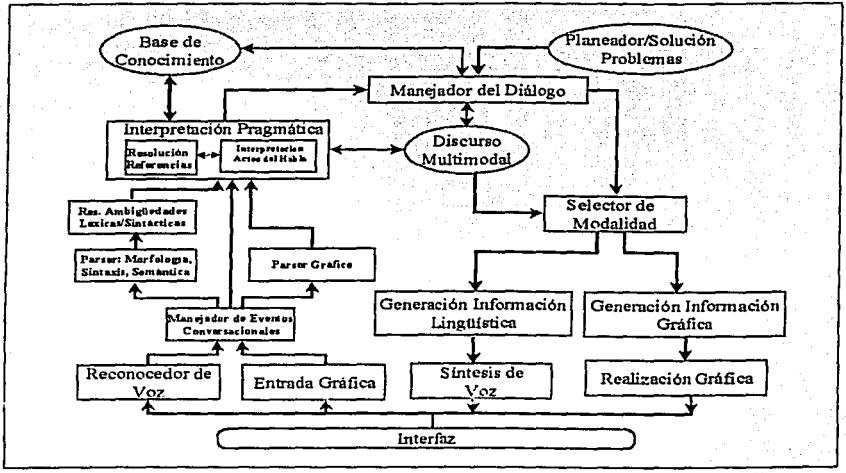


Figura 1.1: Arquitectura del sistema DIME.

1.2 Planteamiento del Problema

En el Departamento de Ciencias de la Computación del Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas (IIMAS) de la Universidad Nacional Autónoma de México (UNAM) se desarrolla actualmente un proyecto llamado "Diálogos Multimodales Inteligentes en Español" (DIME). El proyecto consiste en la creación de un sistema conversacional, es decir, un asistente computacional multimodal para el diseño de cocinas que interactúe por medio del lenguaje hablado y de una interfaz gráfica con el usuario [Pineda et al., 2001].

El proyecto está compuesto de distintos módulos (ver figura 1.1). En la actualidad se está trabajando principalmente en tres: el primero consiste en la recopilación, transcripción y etiquetado de una corpus multimodal (el corpus DIME), el segundo, en el desarrollo de un sistema de reconocimiento de voz para el español y el tercero, en la definición de una gramática y un parser para el español lo suficientemente comprensiva que permita la interpretación de la gramática observada en el corpus DIME.

Este trabajo de tesis se enfoca al estudio, análisis, entendimiento y organización del primer módulo del proyecto, el corpus DIME.

El corpus DIME [Villaseñor et al., 2001] es un corpus de voz que consiste en

una serie de grabaciones de audio y video, la cuáles fueron hechas a partir de diálogos entre dos personas tratando de diseñar cooperativamente una cocina. Todos los archivos de audio generados en DIME fueron transcritos textualmente; aunque el corpus DIME por el momento solo cuenta con la transcripción a nivel de expresiones, dentro del proyecto se contempla el desarrollo de otras transcripciones como fonética, referencial, prosódica y actos del habla.

En este trabajo de tesis se abordaron los siguientes problemas:

1. **¿Qué información lingüística se necesita para crear un sistema conversacional?** Para crear sistemas conversacionales es necesario tener información sobre cómo sucede una conversación, para investigar a fondo los fenómenos lingüísticos que ocurren.

   En la creación de sistemas conversacionales se pueden utilizar distintos tipos de información lingüística como, fonética,

   En este trabajo de tesis la información lingüística se limitará solo a ciertos aspectos que se necesitan en el proyecto DIME como información fonética, fonológica, morfológica, léxica, sintáctica, pragmática y discurso. Esta información debe de estar estrechamente relacionada en algún modelo jerárquico para que permita el estudio, organización y consulta del corpus y de su información.

   La información lingüística debe almacenarse en un modelo.

2. **¿Cómo representar y organizar los datos del corpus DIME?**

   El proponer un esquema de organización para un corpus es una tarea compleja, como lo menciona Llisterri en [Blecua et al.,1999] "el almacenar simplemente los textos de un corpus es una tarea que necesita poco equipamiento y escasos programas, pero tenerlo dispuesto para una fácil recuperación de la información y para la realización de procesos de análisis requiere ya ordenadores preparados y programas sofisticados, en algunos casos realizados *ad hoc*".

   Debido a que es complicado el almacenamiento y organización de un corpus, para la realización de éste propósito se analizaron dos sistemas desarrollados por la comunidad lingüística, que permiten la ejecución de las tareas de almacenamiento y organización. El análisis se hizo también con la intención de saber si se podía reutilizar algún sistema.

3. **¿Cómo extraer dicha información?**

   Se han creado muchos sistemas informáticos para el tratamiento de corpus de voz; sin embargo, existen algunos que aún se encuentran en proceso de desarrollo, tal es el caso de ATLAS y de NITE. Otros sistemas como MATE y EMU se encuentran disponibles en internet para ser utilizados. Todos estos sistemas pretenden hacer de un corpus, un recurso fácil de estudiar y

de manejar; así como se pretende explotar al máximo toda la información del corpus contenida en una base de datos.

En este trabajo de tesis se va a utilizar MATE para validar computacionalmente el modelo jerárquico que se propone en esta tesis. El modelo contendrá las fuentes de conocimiento lingüístico necesarias; estas fuentes van a estar relacionadas entre sí, para permitir la organización del corpus DIME en la base de datos. El modelo jerárquico propuesto se ilustra en la figura 1.2.

El esquema de organización utilizado por MATE permite hacer consultas de los tipos de conocimiento lingüístico existentes en el modelo. Para que se puedan realizar las consultas con MATE, es necesario representar el conocimiento lingüístico del corpus DIME en el formato XML, esta representación de la información constituye las transcripciones del corpus.

1.3 Objetivos

1.3.1 Objetivo general:

El presente trabajo tiene como objetivo general proponer e implementar un modelo jerárquico para integrar múltiples fuentes de conocimiento de un corpus.

1.3.2 Objetivos específicos:

Los objetivos específicos de este trabajo de tesis son:

1. Identificar las múltiples fuentes de conocimiento lingüístico que se manejarán en el corpus DIME y diseñar un modelo que incluya las relaciones y jerarquías entre ellas.
2. Analizar y seleccionar un esquema de representación de las unidades lingüísticas del corpus.
3. Incorporar la información lingüística del corpus DIME en un esquema de organización de base de datos semiestructurado.
4. Validar computacionalmente el modelo propuesto realizando su implementación con la herramienta MATE (*Multilevel Annotation, Tools Engineering*).
5. Demostrar la utilidad de la información de la base de datos realizando procesos de consultas y búsquedas.

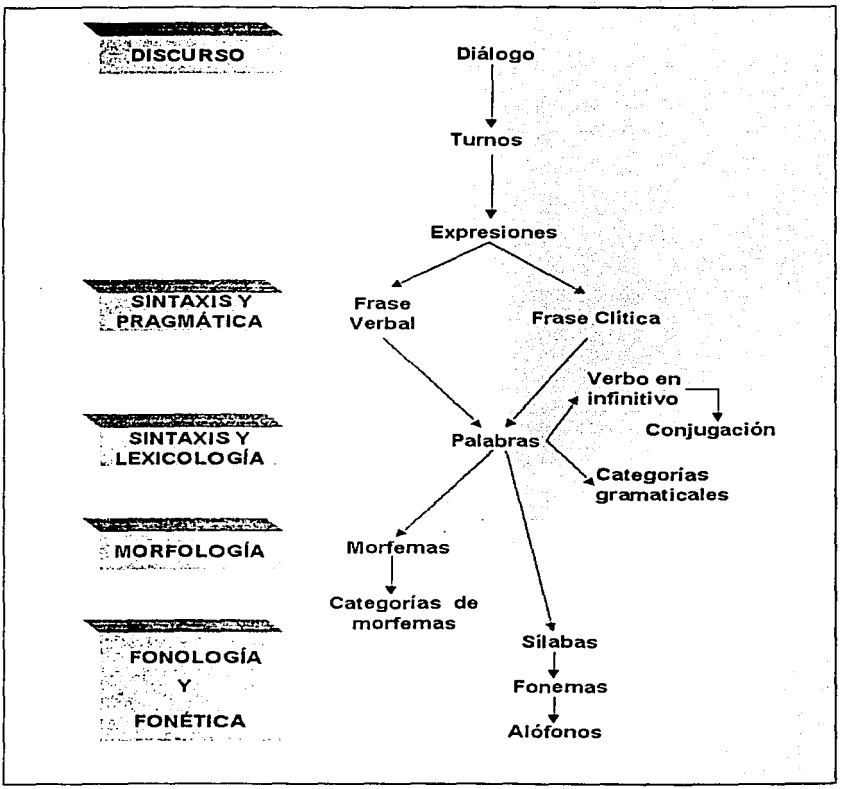


Figura 1.2: Modelo Jerárquico.

1.4 Alcances y Limitaciones

- Los archivos almacenados dentro de la base de datos consisten principalmente de las grabaciones de audio de cada uno de los diálogos del corpus.
- El corpus DIME contiene actualmente la descripción de la información léxica, fonológica y fonética de las grabaciones de audio, las cuales fueron almacenadas en la base de datos.
- El modelo propuesto en este trabajo de tesis incluye las siguientes fuentes de conocimiento lingüístico: fonética, fonología, morfología, lexicología, sintaxis, discurso y pragmática.
- Además de los archivos de audio generados en el corpus DIME también se crearon archivos de video; en este trabajo sólo se incluirá la información lingüística de los videos.
- Las consultas a la base de datos se formularán utilizando el lenguaje de consultas de MATE y los resultados de las consultas serán mostrados en el formato XML.

1.5 Organización de la Tesis

En el capítulo 2 se describe el concepto de corpus, se describen los distintos tipos que existen, así como los distintos corpus existentes para el español hablado en México. En el capítulo 3 se explica el modelo de base de datos semiestructurado así como su relación con XML; además, se describe MATE, la herramienta a utilizar en este trabajo de tesis. En el capítulo 4 se describe el modelo jerárquico y las relaciones de las transcripciones del corpus DIME. En el capítulo 5 se presenta la implementación del modelo jerárquico propuesto para representar la información del corpus DIME en la base de datos de MATE. En el capítulo 6 se describe el sistema de consultas de MATE; se presentan ejemplos de consultas a los datos del corpus DIME. Finalmente, en el capítulo 7 se muestran las conclusiones de este trabajo de tesis.

Capítulo 2

Corpus de Lenguaje Hablado y Escrito

En este capítulo se describe lo que es un corpus, así como la importancia que éstos tienen en el desarrollo de sistemas conversacionales. Además, se explica la conveniencia, necesidad y demanda de tener los corpus organizados y las posibles aplicaciones que éstos pueden tener en otras áreas. Se describe también, los recursos lingüísticos disponibles en la actualidad alrededor del mundo. Se hace una descripción de cada corpus de voz existente para el español hablado en México; se hace énfasis en el Corpus DIME porque éste es el objeto de estudio en esta tesis. Finalmente se mencionan y describen algunas de las herramientas existentes que se usan para analizar, modificar y transcribir un corpus.

2.1 Corpus

Un corpus es un acervo de información que contiene un conjunto de datos, orales o escritos, los cuales generalmente son seleccionados y ordenados de acuerdo a criterios lingüísticos, de tal forma que pueden ser usados como muestras del lenguaje. Un corpus tiende a mostrar a pequeña escala cómo funciona un lenguaje natural.

Existen principalmente dos tipos de corpus, los escritos y los de voz. Los primeros consisten en una serie de textos, muchos de esos textos ya existen y se encuentran disponibles en algún formato de publicación, como novelas, periódicos, manuales, etc; sin embargo, hay otras colecciones de textos que se recopilan porque se necesita tener información de algún tema en particular que no existe o no está disponible. Un corpus de voz consiste en un conjunto de grabaciones de voz, así como un conjunto de comentarios y documentos referentes a los datos de voz; es decir, las grabaciones deben de ser descritas en algún formato ortográfico o no-ortográfico (fonético, prosódico, gramatical, etc.) [Gibbon et al., 1997].

A continuación se mencionan algunas diferencias que existen entre un corpus escrito y un corpus oral o de voz [Gibbon et al., 1997]:

- La durabilidad del texto en comparación con la volatilidad de la voz
  Mientras que el texto generalmente permanece en papel cuando es anotado, el habla es transitoria. La naturaleza de los hechos fonéticos creados por los hablantes, hace que éstos desaparezcan al momento en que llegan a existir; es por esto, que si la lengua hablada quiere ser usada en el futuro debe de ser grabada ya sean en discos, cintas, computadoras, etc.

- Diferencias de tiempos entre la producción de texto y la de voz
  Los datos del habla son funciones de tiempo en un sentido en el cual los datos del texto no lo son; es decir, mientras que un escritor puede consumir el tiempo que desee en la producción de un texto; un hablante, debe codificar y transmitir la información fonética a través de transiciones de sonidos organizados silábica y rítmicamente. El tiempo que generalmente se consume en la creación de nuevo texto, es normalmente mayor al tiempo que se consume en leerlo en voz alta.

- Corrección de errores en la producción de texto y de voz
  En el lenguaje hablado de forma espontánea, todo el comportamiento que genera y corrige el hablante es detectable y forma parte de los datos grabados en un corpus de voz, es decir, se pueden detectar interrupciones, tartamudeos, repeticiones de palabras, así como correcciones producidas por el hablante; todos estos fenómenos percibidos son características del lenguaje natural y deben ser representados en las transcripciones del corpus. Por su parte, en la producción de texto, generalmente se corrigen todos los errores que se encuentran como, faltas de ortografía, palabras incompletas, etc; por lo que en la versión final del texto, todos los errores tienden a desaparecer.

- Cadenas ASCII imprimibles
  La puntuación explícita de frases y de oraciones es una característica importante de los datos escritos de la lengua; se utilizan elementos ASCII para representar los límites entre palabras como, espacios en blanco, puntos, comas, los dos puntos, las citas, los signos de exclamación y de interrogación, etc. En el lenguaje hablado no se encuentran este tipo de elementos ASCII para representar los límites entre palabras, lo que sí se encuentran son pausas al hablar las cuales se dan en función del tiempo.

- Diferencia de tamaño entre palabras escritas y palabras habladas
  Existe una gran diferencia de tamaño entre el espacio de almacenamiento que ocupan los datos hablados y los datos escritos. Lo datos hablados ocupan más espacio de almacenamiento que los datos escritos por la diferencia en la codificación entre el texto y la voz, generalmente esta última se transcribe a más niveles, de los cuales los principales son el ortográfico y el fonético.

- Derechos de autor tanto en textos escritos como en palabras habladas
  Cuando se recopilan corpus escritos que provienen de obras que ya fueron publicadas como novelas, periódicos; es más sencillo incluir una cita donde se haga referencia al artículo o texto que se está usando; en el caso de los corpus hablados, es conveniente que a la persona que fue grabada se le pida que firme unas hojas en las cuales se menciona que autoriza la utilización de las grabaciones de voz para los fines que sean requeridos.

Cuanto más grande sea el corpus existen mayores posibilidades de asegurar la presencia de todos los aspectos del lenguaje y por tanto de acercarse a la realidad; sin embargo, un corpus debe de ser selectivo ya que es complicado recopilar todo lo escrito y/o hablado de una lengua, porque implica un gran esfuerzo humano y bastantes recursos económicos.

Un corpus se utiliza como recurso lingüístico debido a que ha demostrado ser una herramienta excelente para muchas investigaciones, principalmente en el campo de la lingüística ya que proporciona bases mucho más reales (bases empíricas) para el estudio de los lenguajes.

Todo corpus se realiza con un propósito, a continuación se explican algunas de las aplicaciones que tienen los corpus de voz.

2.1.1 Aplicaciones

Un corpus se diseña generalmente para dominios específicos, dependiendo del dominio, se va a determinar qué tipos de datos se van a recolectar y la forma que se van a utilizar para diseñar el corpus. Por ejemplo, en una aplicación de reconocimiento de números a través del teléfono se van a recolectar datos muy distintos a los necesarios en una aplicación que tenga que ver con el diseño de cocinas. En el primer caso es necesario que las grabaciones se hagan a través del teléfono y que se graben solamente números; mientras que, en el segundo, los datos que se van a recopilar están relacionados con todos los elementos que se encuentran dentro de una cocina como, estufa, alacena, refrigerador, etc., y para las grabaciones se puede utilizar un micrófono con el que se puede obtener una mejor calidad en la grabación.

En está sección se presentan las aplicaciones que tiene un corpus de voz. Las aplicaciones se dividen en dos ramas principales [Gibbon et al., 1997], la utilización de un corpus para propósitos científicos y la utilización en aplicaciones tecnológicas.

Corpus de voz para propósitos científicos

Es muy variada la necesidad de los corpus de voz para propósitos científicos; dentro la investigación existen muchos campos en los cuales se puede utilizar un

corpus, por ejemplo:

- **Fonética.** En la investigación fonética todos los aspectos del habla son estudiados. La fonética se encarga del estudio de los sonidos de una lengua; analiza la realización física de los sonidos, es decir, cómo se producen, cómo se perciben, etc.
- **Sociolingüística.** En estas investigaciones se estudia la variación del uso del lenguaje en comunidades heterogéneas, especialmente de personas que viven en lugares urbanizados. Se toma en cuenta la edad, el género (masculino o femenino), el status social de las personas, etc.
- **Psicolingüística.** En esta área se estudia la psicología del lenguaje, es decir, se estudia la producción del habla y los desórdenes del lenguaje que se presentan desde que un niño está aprendiendo a hablar, hasta los procesos mentales que lleva a cabo un adulto para la comprensión del lenguaje.
- **Adquisición del primer idioma.** Para varias disciplinas la adquisición del primer idioma de un niño, es tema de gran interés. Por ejemplo, en la lingüística la forma de hablar de un niño, puede ser usada para investigar regularidades o irregularidades en el lenguaje; en el caso de la psicolingüística, los estudios pueden ser usados para aprender más acerca de cómo se organiza mentalmente el lenguaje.
- **Audiología.** La audiología es una disciplina que se ocupa del estudio científico de la audición, en algunos casos incluye el tratamiento de personas con defectos auditivos.
- **Patología del habla.** El objetivo de los estudios que se hacen en esta área es encontrar remedios a través de terapias que puedan curar algún desórden en el lenguaje hablado.

Corpus de voz para aplicaciones tecnológicas

En cuanto a las aplicaciones tecnológicas un corpus es usado para hacer reconocimiento de voz, síntesis de voz, sistemas de lenguaje y reconocimiento/verificación de un hablante. Dependiendo de la aplicación a utilizar, es distinto el corpus de voz que se necesita; por ejemplo, la síntesis de voz generalmente requiere una gran cantidad de datos de uno o dos hablantes, mientras que el reconocimiento de voz usualmente requiere una pequeña cantidad de datos por cada hablante, pero un gran número de hablantes.

A continuación se explican cuatro tipo de aplicaciones tecnológicas de un corpus de voz:

- **Síntesis de voz.** Ésta consiste en la reproducción de la voz a través de una computadora, es decir, la computadora tiene la capacidad de hablar. Es necesario contar con un corpus para determinar los parámetros del modelo que se va a usar, en la creación de diferentes tipos de sintetizadores de voz. Por ejemplo, si un usuario quiere distintos tipos de voz en su sintetizador, entonces el corpus de voz que se va a generar debe de contener las grabaciones de la voz de diversos hablantes.

- **Reconocimiento de voz.** Consiste en la extracción de una secuencia de palabras dada una señal acústica; es decir, una persona le va a decir algo a una computadora y ésta debe ser capaz de reconocer lo que se le dijo.

- **Sistemas conversacionales.** El propósito de estos sistemas es que una persona entable una plática con una computadora, es decir, la computadora debe de reconocer lo que la persona le está diciendo, una vez hecho esto, la computadora va a hablar (por medio del sintetizador de voz), de tal forma que va a responder la pregunta que la persona le hizo o simplemente va a continuar con la conversación que se está llevando a cabo.

- **Reconocimiento / Verificación de un hablante.** La tarea del reconocimiento automático de un hablante consiste en que la computadora determine la identidad del hablante; mientras que la verificación de un hablante consiste en decidir si un hablante es quien dice ser. El tipo de corpus necesario para reconocimiento de voz, es distinto al necesario para el reconocimiento de un hablante porque para este último caso, es necesario que el corpus contenga múltiples grabaciones en condiciones diferentes del mismo hablante.

En la actualidad se han recopilado o generado a nivel mundial una gran variedad de corpus; muchos de ellos, específicamente los que se generan para aplicaciones tecnológicas, se encuentran disponibles (generalmente desde internet) para que cualquier persona interesada pueda consultarlos. Sin embargo, todavía se siguen desarrollando muchos corpus, debido a que éstos cuentan con características particulares como formato, idioma, tamaño, etc; que los corpus actuales no las tienen; además de que, estos corpus se están generando para un dominio de aplicación en particular.

En la siguiente sección se explican y se muestran algunos porcentajes de los corpus que existen a nivel mundial.

2.1.2 Recursos Lingüísticos Disponibles

La lingüística computacional necesita recursos –materia prima empírica– a partir de los cuáles se pueda, por una parte, realizar análisis y descripciones de un lenguaje, y por la otra, desarrollar productos de tecnología lingüística (sistemas

informáticos que incorporan conocimientos lingüísticos). Los recursos lingüísticos pueden ser colecciones de datos ordenados como un corpus escrito o de voz, diccionarios, bases de datos, etc; estos recursos son adaptados para ser interpretados por las computadoras y por todos aquellos programas que los administran.

Algunos consorcios a nivel mundial, están interesados en producir, difundir y distribuir recursos lingüísticos; uno de los tópicos que se ha convertido en una prioridad, es la reusabilidad de los recursos. Algunos consorcios de este tipo son: *Linguistic Data Consortium* (LDC)[Cieri et al., 2000] y *European Language Resource Association* (ELRA) [Allen et al., 2000]. Asimismo existe un centro que pertenece a la compañía Motorola: *Human Language Data Resource Center* (HLDRC) [Talley, 2000], el cual tiene como misión facilitar el acceso y permitir la utilización de recursos lingüísticos a diferentes grupos dentro de la misma compañía.

ELRA, (Asociación Europea de los Recursos de la Lengua), es un consorcio que tiene por objetivo hacer disponible los recursos lingüísticos que existen a nivel mundial. Para alcanzar esta meta, ELRA es activa en la identificación, distribución, colección, validación, estandarización, mejora y promoción de la producción de los recursos lingüísticos, en el soporte de la infraestructura para realizar campañas de evaluación y en desarrollar un campo científico de los recursos lingüísticos y de su evaluación. Todas estas actividades se alcanzan a través del cuerpo operacional de ELRA. ELDA (*Evaluation & Language resources Distribution Agency*) [1].

En 1999 y a principios del año 2000, ELRA/ELDA realizó una investigación [Allen et al., 2000] a través de un cuestionario con el objetivo de determinar las necesidades de los usuarios en recursos lingüísticos, así como los recursos disponibles. Las cifras obtenidas en los cuestionarios permitieron definir un plan de acción eficaz para ELRA y calcular las inversiones necesarias para promover la producción de nuevos recursos. La investigación fue hecha a la comunidad lingüística a nivel mundial, entre los principales países participantes se encuentran: Estados Unidos, Reino Unido, Alemania, Francia, Japón, Holanda, España, Italia y Canadá. Ver figura 2.1.

La forma de realizar la investigación fue a través del envío del cuestionario a 1,234 direcciones de correo electrónico, de las cuales 987 direcciones eran válidas. De esas 987 direcciones se obtuvo respuesta de un 25.3% del total de los cuestionarios válidos que se enviaron. Todos los datos que se mencionan a continuación están basados en un total de 250 cuestionarios que fueron completamente contestados.

El primer análisis importante que se obtuvo a partir de los cuestionarios, fue acerca de los recursos lingüísticos que los usuarios indican que usan o que están interesados en usar (ver figura 2.2); entre esos recursos se encuentran, las bases

[1] [http://www.icp.inpg.fr/ELRA/home.html]

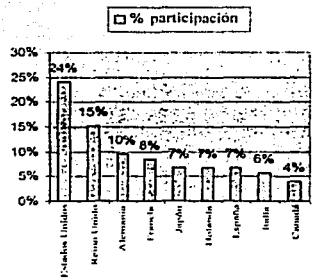


Figura 2.1: Países participantes.

de datos de voz, las bases de datos sintácticas, las bases de datos léxicas y las bases de datos escritas. Cada uno de los recursos lingüísticos fue divido en dos categorías, datos transcritos y datos no transcritos. Aproximadamente 1/3 de los encuestados están interesados en recursos lingüísticos del habla (ej. reconocimiento de voz) y aproximadamente 2/3 están interesados en recursos lingüísticos escritos.

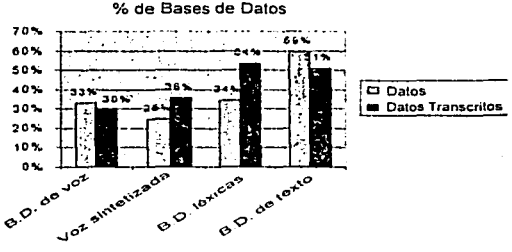


Figura 2.2: Recursos Lingüísticos que se usan o que se necesitan.

Una parte del cuestionario se refería a trabajos sobre tratamiento del lenguaje hablado. Los usuarios pertenecen a dos categorías: los que trabajan en la investigación y los que trabajan en el desarrollo de productos; dos ámbitos se pueden percibir a partir de los resultados obtenidos, reconocimiento de voz y síntesis de voz (ver figura 2.3). En el reconocimiento de voz, 33% de los encuestados trabajan en la investigación y un 13% desarrollan productos; en cuanto a síntesis de voz 24% se dedican a la investigación y 8% al desarrollo de productos. A partir

de estos resultados, se puede ver que 1/3 de los encuestados están relacionados al área de reconocimiento de voz.

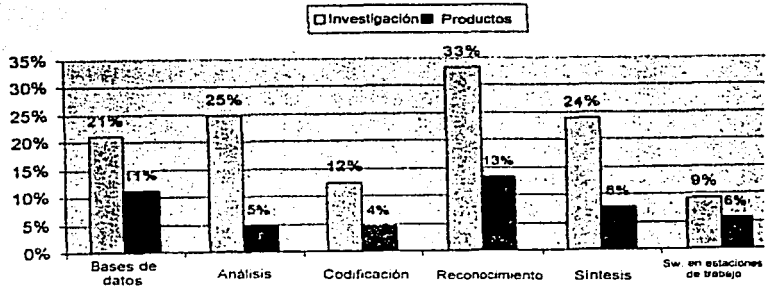


Figura 2.3: Procesamiento de Voz. Tipo de trabajo que se realiza con un corpus en investigación o en desarrollo de productostecnológicos.

Es importante hacer notar que recientemente existe una gran demanda de recursos lingüísticos multimodales y multimedia. Para procesamiento multimodal, el 52% de todos los encuestados están interesados en datos multimedia y 35% están interesados en datos multimodales. Globalmente las necesidades de este rubro han aumentado, ya que solamente 1/18 de los encuestados en otoño de 1997 estaban interesados en los recursos lingüísticos multimodales.

Así también, por esta investigación se supo que los idiomas de los cuales se está más interesado (dentro de la comunidad lingüística) por tener recursos lingüísticos son: Inglés, Francés, Alemán, Italiano y Español. Ver figura 2.4.

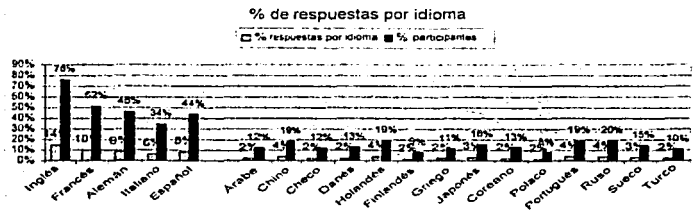


Figura 2.4: Porcentajes demográficos de los recursos lingüísticos más demandados según su idioma.

En esta sección se ha presentado un panorama general de los recursos lingüísticos disponibles alrededor del mundo. En la siguiente, se explican los distintos corpus existentes para el español hablado en México.

2.1.3 Corpus Existentes

La recopilación de un corpus para el español hablado en México es importante porque un mismo idioma (p.e. Español) se habla de diferente forma en diversos países del mundo. En muchos casos dentro de un mismo país se habla diferente en distintas regiones de éste.

La pronunciación de un idioma es importante cuando se habla de un corpus de voz porque, por ejemplo, es posible que un argentino utilice un sistema de reconocimiento de voz hecho en México, y que no le vaya a servir o puede suceder que obtenga un porcentaje de reconocimiento muy bajo; lo anterior puede pasar porque todo sistema de reconocimiento se tiene que entrenar con voces de personas y como el sistema fue hecho en México, las voces con las que se entrenó dicho sistema fue de mexicanos.

La creación de un corpus de voz para el español se hace con la intención de obtener mejores sistemas ya sea conversacionales, de reconocimiento de voz, de entendimiento del lenguaje natural, etc. A continuación se explican los corpus de voz OGI, TLATOA, SALA y DIME que se han creado para el español hablado en México.

OGI

Entre los años de 1996 y 1998 investigadores del "*Center for Spoken Language Resources*" (CSLU) perteneciente al "*Oregon Graduate Institute*" (OGI) en colaboración con investigadores del grupo TLATOA perteneciente a la Universidad de las Américas (UDLA) desarrollaron una versión en Español del software "*CSLU Toolkit*" (grupo de herramientas comprensivas que permiten la exploración, el aprendizaje y la investigación dentro del habla y en la interacción humano-computadora). [2]

Dentro del proyecto, los investigadores realizaron una recolección de datos y se crearon tres pequeños corpus: "Dígitos Continuos", "Letras Continuas" y "Nombres" [Barbosa et al., 1998]. De los tres corpus se realizaron grabaciones y transcripciones a nivel ortográfico y fonético, aunque las transcripciones no fueron hechas en su totalidad, es decir, solo un cierto porcentaje de las grabaciones fueron transcritas. A continuación se presenta una breve descripción de cada corpus:

[2][http://cslu.cse.ogi.edu/toolkit/]

En esta sección se ha presentado un panorama general de los recursos lingüísticos disponibles alredor del mundo. En la siguiente, se explican los distintos corpus existentes para el español hablado en México.

2.1.3 Corpus Existentes

La recopilación de un corpus para el español hablado en México es importante porque un mismo idioma (p.e. Español) se habla de diferente forma en diversos países del mundo. En muchos casos dentro de un mismo país se habla diferente en distintas regiones de éste.

La pronunciación de un idioma es importante cuando se habla de un corpus de voz porque, por ejemplo, es posible que un argentino utilice un sistema de reconocimiento de voz hecho en México, y que no le vaya a servir o puede suceder que obtenga un porcentaje de reconocimiento muy bajo; lo anterior puede pasar porque todo sistema de reconocimiento se tiene que entrenar con voces de personas y como el sistema fue hecho en México, las voces con las que se entrenó dicho sistema fue de mexicanos.

La creación de un corpus de voz para el español se hace con la intención de obtener mejores sistemas ya sea conversacionales, de reconocimiento de voz, de entendimiento del lenguaje natural, etc. A continuación se explican los corpus de voz OGI, TLATOA, SALA y DIME que se han creado para el español hablado en México.

OGI

Entre los años de 1996 y 1998 investigadores del "*Center for Spoken Language Resources*" (CSLU) perteneciente al "*Oregon Graduate Institute*" (OGI) en colaboración con investigadores del grupo TLATOA perteneciente a la Universidad de las Américas (UDLA) desarrollaron una versión en Español del software "*CSLU Toolkit*" (grupo de herramientas comprensivas que permiten la exploración, el aprendizaje y la investigación dentro del habla y en la interacción humano-computadora). [2]

Dentro del proyecto, los investigadores realizaron una recolección de datos y se crearon tres pequeños corpus: "Dígitos Continuos", "Letras Continuas" y "Nombres" [Barbosa et al., 1998]. De los tres corpus se realizaron grabaciones y transcripciones a nivel ortográfico y fonético, aunque las transcripciones no fueron hechas en su totalidad, es decir, solo un cierto porcentaje de las grabaciones fueron transcritas. A continuación se presenta una breve descripción de cada corpus:

[2][http://cslu.cse.ogi.edu/toolkit/]

- Dígitos Continuos: Se grabaron a 50 hablantes, 40 expresiones por hablante, se utilizó un micrófono a 8kHz en la grabación; 10 de los hablantes fueron transcritos manualmente. 20 hablantes adicionales fueron grabados por teléfono.
- Letras Continuas: Se grabaron a 50 hablantes con 20 expresiones por cada uno, se utilizó un micrófono a 16 kHz en la grabación, la grabación de 30 hablantes fue transcrita manualmente.
- Nombres: Se grabaron 50 hablantes, 20 expresiones por hablante, las grabaciones fueron hechas a través del teléfono y 30 grabaciones fueron transcritas manualmente.

TLATOA Common Questions Corpus

Este corpus fue desarrollado por el grupo de investigación TLATOA el cual pertenece al Centro de Investigación en Tecnologías de Información y Automatización (CENTIA) dentro del Departamento de Ingeniería en Sistemas Computacionales de la Universidad de las Américas (UDLA), en el estado de Puebla, México.

El objetivo de TLATOA es crear un corpus grande a través de recolectar voces grabadas por teléfono de personas de distintas regiones del país, las cuales su lengua nativa debe de ser el español mexicano.

El corpus que tienen, consiste de voz grabada por teléfono de más de 400 hablantes; los datos grabados fueron transcritos a dos niveles: ortográfico y fonético. De los datos que se grabaron encontramos que son datos comunes como: dígitos, nombres, apellidos, nombres de calles y repuestas a preguntas donde solo se encuentran un sí o un no como contestaciones.

"TLATOA Common Questions Corpus" es un corpus disponible.

SALA

"*Speechdat Across Latin America*" (SALA) es un proyecto que pertenece a la familia de los proyectos de SpeechDat, dentro de estos proyectos se producen bases de datos de voz para entrenar reconocedores utilizados por aplicaciones telefónicas. [Moreno et al., 2000]

SALA quiere generar varias bases de datos donde se incluyan los idiomas español y portugués con sus distintas variantes que se encuentran alrededor de Latinoamérica; a través de estas bases de datos se pretende obtener un buen reconocimiento de voz.

El español hablado en México es una de las variantes del español que el proyecto SALA ha tomado en cuenta. Se realizan grabaciones de 2000 hablantes de los

- Dígitos Continuos: Se grabaron a 50 hablantes, 40 expresiones por hablante, se utilizó un micrófono a 8kHz en la grabación; 10 de los hablantes fueron transcritos manualmente. 20 hablantes adicionales fueron grabados por teléfono.
- Letras Continuas: Se grabaron a 50 hablantes con 20 expresiones por cada uno, se utilizó un micrófono a 16 kHz en la grabación, la grabación de 30 hablantes fue transcrita manualmente.
- Nombres: Se grabaron 50 hablantes, 20 expresiones por hablante, las grabaciones fueron hechas a través del teléfono y 30 grabaciones fueron transcritas manualmente.

TLATOA Common Questions Corpus

Este corpus fue desarrollado por el grupo de investigación TLATOA el cual pertenece al Centro de Investigación en Tecnologías de Información y Automatización (CENTIA) dentro del Departamento de Ingeniería en Sistemas Computacionales de la Universidad de las Américas (UDLA), en el estado de Puebla, México.

El objetivo de TLATOA es crear un corpus grande a través de recolectar voces grabadas por teléfono de personas de distintas regiones del país, las cuales su lengua nativa debe de ser el español mexicano.

El corpus que tienen, consiste de voz grabada por teléfono de más de 400 hablantes; los datos grabados fueron transcritos a dos niveles: ortográfico y fonético. De los datos que se grabaron encontramos que son datos comunes como: dígitos, nombres, apellidos, nombres de calles y repuestas a preguntas donde solo se encuentran un sí o un no como contestaciones.

"TLATOA Common Questions Corpus" es un corpus disponible.

SALA

"*Speechdat Across Latin America*" (SALA) es un proyecto que pertenece a la familia de los proyectos de SpeechDat, dentro de estos proyectos se producen bases de datos de voz para entrenar reconocedores utilizados por aplicaciones telefónicas. [Moreno et al., 2000]

SALA quiere generar varias bases de datos donde se incluyan los idiomas español y portugués con sus distintas variantes que se encuentran alrededor de Latinoamérica; a través de estas bases de datos se pretende obtener un buen reconocimiento de voz.

El español hablado en México es una de las variantes del español que el proyecto SALA ha tomado en cuenta. Se realizan grabaciones de 2000 hablantes de los

cuales:

- 50% son de sexo masculino
- 50% de sexo femenino

Igual que TLATOA las grabaciones son por teléfono, sin embargo, los datos que corresponden al español hablado en México, están en proceso de grabación y transcripción; al momento sólo cuentan con un 50% de grabaciones realizadas. Entre los datos grabados se encuentran: dígitos, cadenas de dígitos, números y cantidades, fechas y tiempos, nombres de ciudades, compañías y nombres de personas, palabras y oraciones ricas fonéticamente.

DIME

El corpus DIME es desarrollado dentro del proyecto "Diálogos Multimodales Inteligentes en Español" (DIME); este proyecto tiene como objetivo el desarrollo de un agente conversacional multimodal, el cual va a obtener como entrada información hablada en Español [Pineda et al., 2001].

Para la construcción del sistema conversacional se tomó en cuenta el dominio de aplicación del sistema, el cuál tiene que ser lo suficientemente complejo donde valga la pena usar lenguaje natural; pero al mismo tiempo debe de ser lo más sencillo posible que permita ser modelado con la tecnología computacional existente.

El dominio de aplicación del proyecto DIME es el diseño de cocinas. Se escogió éste dominio porque la mayor parte de la gente sabe cómo y con qué está integrada una cocina (los muebles que se usan dentro de una, así como la distribución de esos muebles).

El corpus fue recolectado con la ayuda del escenario de los experimentos del Mago de Oz. Estos experimentos consisten en la simulación de una interacción humano-computadora, donde un humano (*mago*) juega el papel de una computadora y diferentes personas (*sujetos*), uno a la vez, se les pide resolver una o varias tareas dentro del dominio, con la ayuda del *mago*. El objetivo de realizar estos experimentos en el proyecto DIME fue el de obtener diálogos para el Español hablado en México, donde colaboraran juntos el sistema y el usuario para diseñar una cocina.

Se tomaron en cuenta seis características para diseñar y correr los experimentos del Mago de Oz [Villaseñor et al., 2001]:

- *Dominio de aplicación.* En este caso es el diseño de cocinas.

- *Complejidad de la tarea.* La cual debía de ser tan compleja que permitiera la utilización del lenguaje natural, pero a la vez debía de ser sencilla ya que era necesario poder resolver la tarea.

- *Adaptación del interlocutor.* Para los experimentos se seleccionaron participantes que no fueran cercanos a la persona que tomaba el rol del *mago* debido a que se podía esperar que la gente cercana al mago usara vocabulario amigable y llegará hasta bromear. Esto se hizo con la intención de obtener diálogos similares a los que hay entre un humano y una computadora.

- *Conocimiento del Sistema.* Ningún *sujeto* (participante) podía participar en más de un experimento del Mago de Oz, para prevenir que los *sujetos* adaptaran al sistema su conducta lingüística, sus acciones o soluciones.

- *Conducta del Mago.* El *mago* debía actuar como si fuera realmente un sistema computacional. Tenía que responder lo mas rápidamente posible, no debía cometer errores, tenía que hablar en forma impersonal, etc.

- *Conducta condicionada del sujeto.* La gente inconscientemente imita a otras personas con el fin de poder resolver alguna tarea; para los experimentos del Mago de Oz se hizo una sesión de demostración del funcionamiento del sistema a los *sujetos* involucrados en las grabaciones del corpus DIME, aunque esta demostración pudo tener el efecto de condicionar la conducta del *sujeto* dentro de los experimentos.

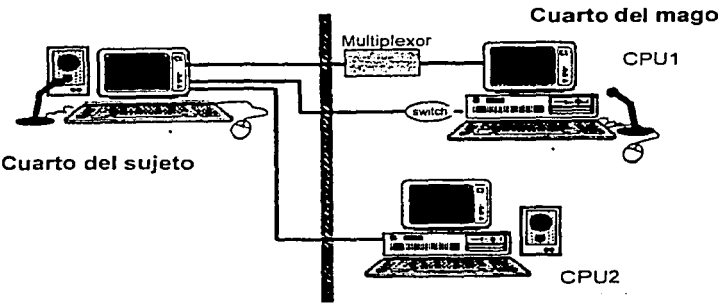


Figura 2.5: Disposición Física.

Para llevar a cabo los experimentos del Mago de Oz se diseñó y acondicionó un laboratorio (ver figura 2.5), el cual consistía de dos cuartos juntos, uno para el *sujeto* y el otro para el *mago*. En el cuarto del *sujeto* había un monitor que

compartía la imagen con el monitor del cuarto del *mago*, dos bocinas donde el *sujeto* podía escuchar al *mago*, un micrófono para poder hablar con el sistema y un ratón (*mouse*) para poder señalar objetos y regiones en la pantalla del monitor. En el cuarto del *mago* se tenían los mismos objetos para los mismos objetivos de comunicación, además se contaba con un *switch*, el cual impedía las señales de entrada del mouse del *sujeto* para que no interrumpiera al sistema cuando éste estaba trabajando. Se usaba también un multiplexor el cual permitía compartir la misma señal de video entre los dos participantes y se tenía otra computadora (CPU2) la cual grababa la voz del *mago*, mientras que la voz del *sujeto* y la interacción gráfica del *sujeto* y del *mago* fue grabada en el CPU1.

Participaron 16 personas en los experimentos, con edad promedio de 30 años y la mayor parte de esas personas están relacionadas al área de Ciencias de la Computación. Cada persona grabó dos diálogos. El corpus DIME provee información empírica entre el lenguaje hablado y los gestos deícticos, para el estudio del uso y la interacción humano-computadora dentro de un ambiente gráfico. DIME es un corpus de habla espontánea, es decir cada hablante es libre de escoger las palabras que va a usar y la forma en que lo va a hacer; aunque para mantener cierto control sobre el material hablado se escogió el dominio de aplicación (diseño de cocinas).

Las grabaciones de los participantes fueron hechas con micrófono, se obtuvo un total de 31 diálogos con 7:10 horas de grabación (ver tabla 2.1). El corpus contiene material grabado por audio y video. Este corpus está en proceso de transcripción a nivel ortográfico, fonético, actos del habla y categorías gramaticales, entre otros; se considera incluir datos acerca de las acciones relacionadas ente el audio y el video como el señalar un objeto en la pantalla mientras se dice "éste", como mostrar un objeto gráficamente mientras se dice "este es el catálogo de muebles", etc.

Total (31 diálogos)	En promedio (por diálogo)
27459 palabras	886 palabras
5779 expresiones	185 expresiones
3606 turnos	115 turnos
7:10 horas	14 minutos

Tabla 2.1: El corpus DIME.

En la actualidad el corpus DIME está transcrito en su totalidad a nivel textual. La tarea de transcripción consistió en segmentar los diálogos, esto es dividir cada diálogo grabado en pequeñas unidades completas (expresiones). De esta forma se tiene que un diálogo es un intercambio de expresiones entre dos interlocutores, donde el diálogo es estructurado por turnos; durante un turno cada participante puede contribuir con una o más expresiones.

En la siguiente sección se explican algunas herramientas que se han desarrollado para el tratamiento de corpus.

2.1.4 Herramientas

En virtud de que un corpus en general, por sí solo, no es suficiente para facilitar datos del comportamiento del lenguaje natural, en la actualidad se han desarrollado una gran número de sistemas informáticos que permiten el análisis, la manipulación y la organización de uno o varios corpus. Estas herramientas permiten, entre otras cosas, la transcripción y el almacenamiento de un corpus así como de los archivos asociados en bases de datos, la visualización de los textos, etc.

La mayor parte de las herramientas que se han desarrollado, se han hecho para propósitos específicos y para dominios de aplicación en particular; es por eso que en la actualidad muchos investigadores están interesados en reutilizar herramientas ya existentes, para así ahorrar todo el tiempo que se lleva el desarrollar una herramienta.

Existen diversas herramientas especializadas en el tratamiento de un corpus de voz, sin embargo, no existen muchas herramientas que manipulen más que señales de voz o textos, es decir, que manejen información multimodal como imagenes o videos. El corpus DIME, que cuenta con información multimodal, necesita ser organizado dentro de una herramienta que maneje distintos tipos de información, entre ellos uno que permita la visualización, consulta y transcripción de los videos, no obstante, por el momento las herramientas que en sus funciones se encuentra el manejo de videos, se encuentrán todavía en proceso de desarrollo, tal es el caso de ATLAS y de NITE.

Cabe mencionar que existen algunas herramientas que manejan la transcripción o la visualización de videos, pero estas herramientas lo hacen independientemente; es decir, esas herramientas pueden hacer solamente esas funciones. Lo que se necesita para el corpus DIME es una sola herramienta que maneje textos, audios y videos.

A continuación se describen brevemente algunas de las herramientas más destacadas [Dybkjaer et al., 2001] para el tratamiento de un corpus de voz, algunas manejan información multimodal (herramientas para la transcripción de gestos, expresiones faciales, etc.)

- Anvil (*Annotation of Video and Language Data*) es una herramienta hecha en Java para la transcripción de lenguaje natural y de videos. Maneja voz y gestos.
- ATLAS (*Architecture and Tools for Linguistic Analysis Systems*). La herramienta está en proceso de desarrollo. La estructura de ATLAS propor-

ciona una arquitectura encaminada a facilitar el desarrollo de aplicaciones lingüísticas. La meta principal de ATLAS es proporcionar una abstracción de las diversidas transcripciones lingüísticas existentes.

- **CLAN** (*Computerized Language Analysis*). Es un programa diseñado específicamente para analizar datos transcritos al formtado de CHILDES (*Child Language Data Exchange System*). Las transcripciones pueden ser referenciadas a archivos de audio o de video.

- **CSLU Toolkit** (*Center for Spoken Language Understanding Toolkit*) Consiste en una serie de herramientas entre las que se encuentran, *Rapid Application Developer*, *BaldiSync* (para la animación facial), *SpeechView* (para la visualización de señales de audio), *OGIsable* (una herramienta de transcripción), *speech recognition tools* (herramientas para el reconocimiento de voz) y un ambiente de programación (CSLUsh). *OGIsable* es la única herramienta incluida en el *CSLU Toolkit*, permite agregar propiedades al texto antes de que éste es reproducido, por ejemplo sincronizar expresiones faciales con la salida de audio.

- **EMU**. Es un sistema de base de datos del habla que proporciona un conjunto de interfaces flexibles, que permiten el desarrollo y la extracción de datos de las bases de datos del habla. Una base de datos del habla en EMU consiste en una colección de archivos de datos de voz, cada uno de los cuales tiene una o más transcripciones asociadas.

- **MATE** (*Multilevel Annotation Tools Engineering*). MATE es una herramienta basada en Java que permite la transcripción a diferentes niveles de corpus de voz basados en diálogos, así como consultas a la información transcrita.

- **MultiTool** (desarrollado como parte de un proyecto sueco acerca de una plataforma para un corpus multimodal de lenguaje hablado). Es una herramienta basada en Java que permite el uso de audio y video del lenguaje hablado. Esta herramienta está parcialmente implementada.

- **NITE** (Natural Interactivity Tools Engineering) Se encuentra en proceso de desarrollo. Su objetivo es desarrollar una herramienta que permita la transcripción a varios niveles, así como la recuperación y la explotación de los datos en diálogos interactivos naturales múltiples (humano-humano, humano-computadora).

- **SmartKom** Es un proyecto alemán que se encuentra en la etapa de especificación y tiene como objetivo combinar las ventajas de la comunicación basada en diálogos con las ventajas de una mezcla de interfaces gráficas e interacciones mímicas y gesturales. SmartKom utiliza las herramientas desarrolladas en el proyecto de Verbmobil para la transcripción de audio.

- **syncWRITER** es una herramienta de transcripción diseñada para proporcionar ayuda en la transcripción de "acontecimientos síncronos". Por ejemplo, datos de voz y de video. Esta herramienta se ejecuta solamente en máquinas Macintosh.

En este trabajo de tesis se va a utilizar la arquitectura planteada por MATE, debido a que ésta herramienta es la que mejor se adecúa a las necesidades de organización y consulta del corpus DIME. MATE fue creada con la finalidad de permitir la reutilización de recursos lingüísticos, tratando de afrontar los problemas de creación, mantenimiento y adquisición de los mismos. MATE es una herramienta que ayuda a resolver los problemas anteriores, utiliza un lenguaje (XML) como formato estándar, el cual permite codificar las transcripciones de cada corpus.

2.2 Conclusiones

En este capítulo se presentó la importancia que tiene la creación de los corpus de voz, y más aún, se hizo énfasis en la importancia que tiene el desarrollo de un corpus para el español hablado en México. Como se mostró anteriormente, la recopilación del corpus DIME fue necesaria para diseñar el sistema conversacional que se pretende desarrollar dentro del proyecto DIME.

Es necesario que cualquier corpus que se genere esté bien organizado. De esta forma se podrá explotar más y mejor la infomación que contiene mediante la ayuda de una herramienta computacional.

En el siguiente capítulo se describe MATE a más detalle. También se explica brevemente lo que son las bases de datos semiestructuradas y la relación que tienen con el lenguaje XML.

Capítulo 3

Descripción de MATE

Antes de iniciar la descripción de la arquitectura de MATE, se va a explicar el modelo de base de datos semiestructurado. También se describe el lenguaje XML (eXtensible Markup Language), el cual está muy relacionado con las bases de datos semiestructuradas, así como con la herramienta de MATE.

3.1 Bases de Datos Semiestructuradas

Recientemente se han empezado a utilizar más las bases de datos semiestructuradas, las cuales pueden ser accesadas comúnmente desde la Web. Dentro de internet se manejan extensas cantidades de información, muchas veces la gente quiere hacer consultas a los datos ahí contenidos; sin embargo, estas consultas no se pueden realizar fácilmente porque la información que se encuentra ahí almacenada tiene distintos formatos.

Cuando hablamos de bases de datos tradicionales (relacionales), toda la información que se va a almacenar dentro de ellas sigue un formato de almacenamiento, el cual ya fue establecido. Es decir, se tiene el modelo donde se van a guardar todos los datos y con éste se empiezan a almacer dichos datos. Si los datos no concuerdan con el modelo de almacenamiento previamente establecido, estos datos no podrán ser almacenados en este tipo de bases de datos.

Se pueden entender a los datos semiestructurados como los datos que no cuentan con un esquema (*schemaless*) o como los datos que se describen por sí solos (*self-describing*). Estos datos no cuentan con una descripción por separado de sus tipos o de su estructura de datos.

A diferencia de los modelos estructurados (p.e. bases de datos relacionales) donde la estructura de los datos se describe generalmente primero en un esquema por separado, y después se crean los casos (instancias) de este esquema; para los datos semiestructurados, éstos se describen directamente usando una sintaxis

simple.

Una de las ventajas principales de los datos semistructurados es la capacidad de acomodar las variaciones de los datos en una estructura. Las variaciones consisten generalmente de campos duplicados, datos perdidos, o en cambios de menor importancia en su representación.

Existen algunas características importantes que se deben de considerar al diferenciar los datos semiestructurados de otros datos [Abiteboul, 1997]:

1. **La estructura es irregular.** En muchas aplicaciones las grandes colecciones de datos que se tienen, generalmente consisten de elementos heterogéneos. Algunos elementos pueden estar incompletos. Por otro lado, otros elementos pueden registrar información extra (p.e. transcripciones). Distintos datos pueden ser usados para los mismos tipos de información (p.e. en algunas partes de la base de datos los precios pueden estar en pesos y en otras en dólares).

2. **La estructura está implícita.** Por ejemplo, los documentos electrónicos generalmente están compuestos de un texto y una gramática que describe la estructura del documento (p.e. un DTD en XML). Un parseo[1] a uno de los documentos nos permite aislar piezas de información y detectar relaciones entre esas piezas. Sin embargo, la interpretación de estas relaciones puede estar más allá de la capacidad de los modelos de base de datos estándar; es por esto que la interpretación la llevan a cabo otras herramientas. Se puede ver como está implícita esta estructura porque:

   - se requiere realizar algún proceso computacional para obtenerla (p.e. parseo) y
   - no siempre es inmediata la correspondencia entre el árbol parseado y la representación lógica de los datos.

3. **La estructura es parcial.** Un objetivo, comúnmente difícil de lograr, es el estructurar completamente los datos. Se pueden encontrar diversos tipos de información dentro de la misma base de datos que hacen que la estructura sea parcial. Algunos datos carecen totalmente de estructura (p.e. los mapas de bits); otros pueden revelar solamente una cierta estructura (p.e. texto no estructurado). Otros datos pueden proveer una limitada forma de estructura (p.e. algunas herramientas de recuperación de información). Y también encontramos aquellos datos que están estructurados completamente.

[1] Parsear es hacer un análisis gramatical. El análisis se hace dividiendo y subdividiendo las partes de una oración en sus componentes. Además el parser describe las relaciones que existen entre esas partes.

4. **Estructura indicativa vs. estructura restringida.** En las aplicaciones de las bases de datos estándar se tiene una política estricta de escritura que se hace cumplir para proteger a los datos. En estas aplicaciones, se usa un esquema que describe los tipos de escritura para los datos que solamente pueden manejar al sistema. Por el otro lado, para las bases de datos semiestructuradas, no existe una estructura restringida donde solamente se pueden usar los tipos de datos que se encuentran en el esquema; al contrario, la estructura puede aceptar datos parecidos a los que se estan usando o puede aceptar nuevos datos de otros tipos.

5. **Creación de los esquemas.** Los sistemas de bases de datos tradicionales se basan en la hipótesis de que un esquema debe de ser definido antes (a priori) de introducir cualquier dato. Sin embargo; este no es el caso para los datos semiestructurados, donde la definición de un esquema se hace después de la existencia de los datos.

6. **El esquema evoluciona rápidamente.** En los sistemas de base de datos estándar, el esquema es visto como algo inmutable; las actualizaciones se hacen muy de vez en cuando y generalmente siempre son muy costosas. En el contexto de las bases de datos semiestructurados, el esquema es muy flexible y puede ser actualizado tan fácilmente como cualquier dato.

7. **La distinción entre esquema y datos es confusa.** En las aplicaciones de bases de datos estándar, un principio básico es la distinción entre el esquema (aquel que describe la estructura de la base de datos) y los datos (las instancias a la base de datos). En el caso de las bases de datos semiestructuradas muchas de las diferencias entre esquema y datos desaparecen; por ejemplo, las actualizaciones de los esquemas son frecuentes, las reglas de los esquemas pueden ser violadas, el esquema puede ser muy largo, las mismas consultas/actualizaciones se pueden hacer tanto al esquema como a los datos.

La explicación de qué son los datos semiestructurados ayuda a entender de dónde proviene la idea de la creación de XML; así como de hacer notar que en la actualidad es muy difícil tratar de estructurar la información desde un principio, y en virtud de que algunos sistemas no aceptan ciertos datos, éstos se dejen fuera de un posible almacenamiento en la base de datos. Hoy en día es más conveniente que los sistemas sean más flexibles y permitan comunicarse con otros sistemas, de esta forma se podrá intercambiar información más facilmente.

3.2 XML

El lenguaje XML "eXtensible Markup Language" es un lenguaje que fue diseñado para describir datos. XML comparte muchas características de los datos semiestructurados. Por ejemplo, la estructura puede ser irregular, no se conoce desde

un principio, puede cambiar frecuentemente y sin notarlo. Además, los datos en XML son autodescriptivos.

Para consultar la información en los documentos XML es necesario realizar algunas operaciones de bases de datos como extracción, integración, traducción y almacenamiento de datos. Las investigaciones que se han hecho hasta el momento en el área de los datos semiestructurados ofrecen algunas soluciones a los problemas de las operaciones en bases de datos que se tienen en XML.

La abundancia de recursos lingüísticos ha creado la necesidad de estandarizarlos. La estandarización consiste en describir a través de un lenguaje toda la información que contiene un corpus, para esto se han utilizado lenguajes como SGML (*Standard Generalized Markup Language*) y XML (*eXtensible Markup Language*) los cuales son lenguajes de marcas textuales que se usan para representar documentos en formato digital. Estos lenguajes fueron diseñados para describir datos y enfocarse a ellos; a diferencia de otro lenguaje llamado HTML (*HyperText Markup Language*), tanto SGML como XML no se preocupan por cómo va a aparecer la información desplegada en una pantalla. A continuación se explica el lenguaje XML.

3.2.1 El lenguaje XML

XML, "*eXtensible Markup Language*" es un lenguaje que fue diseñado para describir datos. XML es un lenguaje de marcas; es decir, consiste en una serie de etiquetas que engloban a todo un documento. Estas etiquetas no se encuentran predefinidas en XML, cada persona define sus propias etiquetas y pueden ser tantas como se necesiten.

Un documento XML no hace nada por sí solo. Fue creado como una forma de estructurar, almacenar y mandar información. La figura 3.1 es un ejemplo de un documento en XML. El ejemplo muesta un recado que hace *Pablo* para *Pedro* en el formato de XML. El recado tiene la fecha, un mensaje, y se sabe quien envía y quien recibe el recado. Sin embargo, este documento sigue sin hacer algo; solamente es información que está entre varias etiquetas. Alguien debe de escribir un programa de computadora ya sea, para mandar, recibir o visualizar el documento.

Los datos en XML son agrupados en elementos delimitados por etiquetas, estos elementos pueden ser anidados. En la figura del recado en XML, podemos distinguir las etiquetas que se uasaron; éstas son aquellas cadenas que se encuentran entre un < y un > como, *recado, para, de, mensaje y fecha.* Cada etiqueta que inicia (P.e. <*recado*>) debe de tener una etiqueta de finalización que concuerde (P.e. </*recado*>). A la información que se encuentra entre estas dos etiquetas se les llama elementos.

Las etiquetas del ejemplo anterior tienen el siguiente significado:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE note SYSTEM "recado.dtd">
<recado>
    <para> Pedro </para>
    <de> Pablo </de>
    <mensaje> No olvides que hoy tenemos
            entrenamiento a las 5
    </mensaje>
    <fecha> 24 noviembre </fecha>
</recado>
```

Figura 3.1: Documento en XML: recado.xml

- **<recado>** Esta etiqueta contiene toda la información del recado que se hizo. *<recado>* es el elemento raíz del XML y tiene cuatro hijos (*para, de, mensaje y fecha*).
- **<para>** Aquí se indica hacia quién va dirigido el recado.
- **<de>** Por esta etiqueta se sabe quien fue la persona que hizo el recado.
- **<mensaje>** Esta etiqueta contiene el mensaje del recado. Aquello que le se le quiere decir a la persona que va dirigido el mensaje.
- **<fecha>** Se indica la fecha en la cual se hizo el mensaje.

En un documento XML, se describe información, XML no fue diseñado para mostrar datos, por lo que no se preocupa por cómo se va a visualizar posteriormente toda la información que contiene.

Todo documento en XML puede ser usado para:

- **Intercambiar datos.** En el mundo real, los sistemas computacionales y las bases de datos contienen datos en formatos incompatibles. Uno de los retos para los desarrolladores que se ha llevado más tiempo, ha sido el intercambio de datos a través de internet entre tales sistemas. La conversión de los datos al formato de XML puede reducir significativamente esta complejidad y los documentos podrán ser entendidos por distintas aplicaciones.
- **Compartir datos.** Con XML se puede intercambiar información entre sistemas incompatibles. Como los documentos en XML son almacenados

en el formato de texto plano, esto permite una manera de compartir los datos independientemente del software y del hardware que se tenga. La independencia facilita la creación de datos con los cuales diferentes aplicaciones pueden trabajar y la expansión de un sistema a nuevos sistemas operativos, servidores, aplicaciones, etc.

- **Almacenar datos.** XML puede ser usado para almacenar datos en archivos o en bases de datos. Distintas aplicaciones pueden almacenar y recuperar información de la base de datos, y se pueden usar otras herramientas para mostrar los datos.

Existen dos tipos de documentos XML: bien formados y válidos. Los primeros son aquellos documentos que cumplen con todas las especificaciones del lenguaje respecto a las reglas sintácticas. Los documentos XML tienen una estructura jerárquica muy estricta y los documentos bien formados deben de cumplirla. Los segundos son documentos que además de estar bien formados, deben de cumplir con unas especificaciones que se encuentran en un DTD (*Document Type Definition*). Estos documentos siguen una estructura y una semántica determinada por un DTD.

Los documentos en XML pueden ir acompañados de otros documentos, un DTD y un XSL (*eXtensible Stylesheet Language*). A continuación se explican cada uno.

3.2.2 Definición del tipo de documento (DTD)

El propósito de un DTD es definir los bloques de construcción legales para crear un documento en XML. El DTD define la estructura del documento con una lista de elementos que son permitidos.

Las ventanjas de utilizar DTD's consisten principalmente en las cuatro siguientes[2]:

- Con un DTD, cada archivo en XML puede llevar consigo una descripción de su propio formato.
- Diferentes grupos de personas pueden acordar en el uso de un mismo DTD para el intercambio de información.
- Las aplicaciones que se tengan dentro de un sistema pueden usar un DTD estandar para verificar que los datos que se reciben, de personas externas al sistema, son válidos.

[2]http://www.w3schools.com/dtd/default.asp

- Se pueden usar los DTD's para verificar la propia información que se encuentra en el sistema.

Un DTD puede residir dentro del mismo documento XML como parte de su declaración de tipo de documento o puede ser externo al XML, lo cual, permite que el mismo DTD pueda ser usado por varios XML's. En la figura 3.2 se presenta un ejemplo de un archivo DTD externo. Este DTD valida el ejemplo de XML (recado.xml) mostrado anteriormente.

```
<!DOCTYPE recado [
  <!ELEMENT recado (para, de, mensaje, fecha)>
  <!ELEMENT para (#PCDATA)>
  <!ELEMENT de (#PCDATA)>
  <!ELEMENT mensaje (#PCDATA)>
  <!ELEMENT fecha (#PCDATA)>
]>
```

Figura 3.2: recado.dtd

A continuación se explica la figura 3.2 "recado.dtd":

- **!DOCTYPE recado** (linea 1) define que es un documento del tipo *recado*
- **!ELEMENT recado** (linea 2) define al elemento *recado*, el cual contiene cuatro elementos: *para, de, mensaje y fecha.*
- **!ELEMENT para** (linea 3) define al elemento *para* que es del tipo #PCDATA. *#PCDATA* es texto que va a ser analizado por un parser.
- **!ELEMENT de** (linea 4) define al elemento *de* que es del tipo #PCDATA.
- **!ELEMENT mensaje** (linea 5) define al elemento *mensaje* que es del tipo #PCDATA.
- **!ELEMENT fecha** (linea 6) define al elemento *fecha* que es del tipo #PCDATA.

3.2.3 Lenguaje para definir hojas de estilo (XSL)

Debido a que en XML no existen etiquetas predefinidas, el significado de éstas no se entiende. Es decir, un navegador no sabe como desplegar un documento en XML.

XSL es un lenguaje que sirve para expresar hojas de estilo, lo que permite definir una presentación o un formato para un documento en XML. Por ejemplo, que aparezcan unas letras grandes y de color rojo, otras letras que sean de color morado, otras que estén en negritas y de color azul, etc. Un mismo documento en XML puede tener varias hojas de estilo XSL que lo muestren en diferentes formatos (PDF, HTML, PostScript, etc).

Un XSL consta de 3 partes:

1. **XSLT.** Es un lenguaje que transforma documentos en XML a otros documetos en XML.
2. **XPath.** Es un lenguaje que se usa para definir las partes de un documento en XML.
3. **XSL Formatting Objects.** Es un vocabulario que se usa para darle formato a un documento en XML.

No es necesario que un documento en XML vaya acompañado de un documento en XSL; sin embargo, si se quiere mostrar el XML en algún navegador y se quiere resaltar la información por medio de colores, o se necesita que la información se encuentre en algún formato en particular; entonces es recomendable definir documentos en XSL para los XML.

XML además de que ha sido usado extensamente, ha comprobado ser un formato efectivo para la transcripción de corpus de texto y de voz. Cuando aún no existía XML, numerosos proyectos de investigación, incluyendo al TEI (*Text Encoding Initiative*)[3] y al CES (*Corpus Enconding Standard*)[4] desarrollaron un estándar en SGML de propósito general para la transcripción lingüística. En la actualidad muchos investigadores que desarrollan corpus, han optado por utilizar a XML como el formato estándar para la codificación de las transcripciones de cada corpus.

La estructura jerárquica flexible de XML concuerda muy bien con muchos tipos de representación lingüística existentes. MATE (que se explica en la siguiente sección) utiliza XML como su formato de entrada/salida; así como también, usa una estructura de modelo de datos interna muy similar.

[3]http://www.tei-c.org/

[4]http://www.cs.vassar.edu/CES/

3.3 Arquitectura de MATE

MATE (*Multilevel Annotation, Tools Engineering*) [Mckelvie et al., 2000] fue diseñado para ayudar en la transcripción de un corpus lingüístico y para ayudar a los usuarios a explorar las relaciones que existen entre diferentes transcripciones de un mismo corpus. La tarea que realiza MATE está encaminada a la transcripción de diálogos de lenguaje hablado, por lo que almacena transcripciones estructuradas (en texto) y relaciona estas transcripciones con una señal de voz.

MATE provee una estructura general para definir editores especializados en la transcripción de un corpus. Hace relativamente fácil la tarea de transcribir un corpus y de escribir editores para un esquema de transcripción en particular. Esta facilidad se ha dado porque se ha permitido que cualquier esquema de transcripción se codifique con el lenguaje XML, y que se escriban y se utilicen documentos en XSL para la transformación de esos XML's.

Cualquier persona que utilice MATE podrá, visualizar y editar algún corpus codificado en XML, realizar consultas a una parte o a todo el corpus, agregar nuevos niveles de transcripción y podrá mostrar o tener como salida los resultados de las consultas.

La arquitectura de MATE consiste de los siguientes componentes (ver figura 3.3):

- **Base de datos interna.** Es una representación de una serie de documentos XML que se encuentran relacionados. Hay funciones para cargar y mostrar archivos XML, dentro y fuera de la base de datos.
- **Lenguaje y procesador de consultas.** Se encargan de seleccionar partes de la base de datos sobre las que se harán consultas o se visualizarán datos.
- **Lenguaje y procesador de hojas de estilo.** Se usan para definir e implementar (respectivamente) un lenguaje que describa las transformaciones estructurales a la base de datos.
- **Procesador para la visualización.** Es responsable de las acciones de edición y visualización. Lo que hace es tomar un objeto que se obtuvo después de transformarlo con una hoja de estilo y lo muestra al usuario.
- **Interfaz de usuario.** Maneja la manipulación de archivos y las herramientas que se invocan (aquellas que se mandan llamar).

MATE es independiente de la plataforma sobre la que se use, debido a que está escrito completamente en Java. MATE permite el reuso de programas computacionales, usando a XML como el formato de intercambio de datos; o llamando a otros modulos de Java, quienes pueden accesar a la representación interna de MATE.

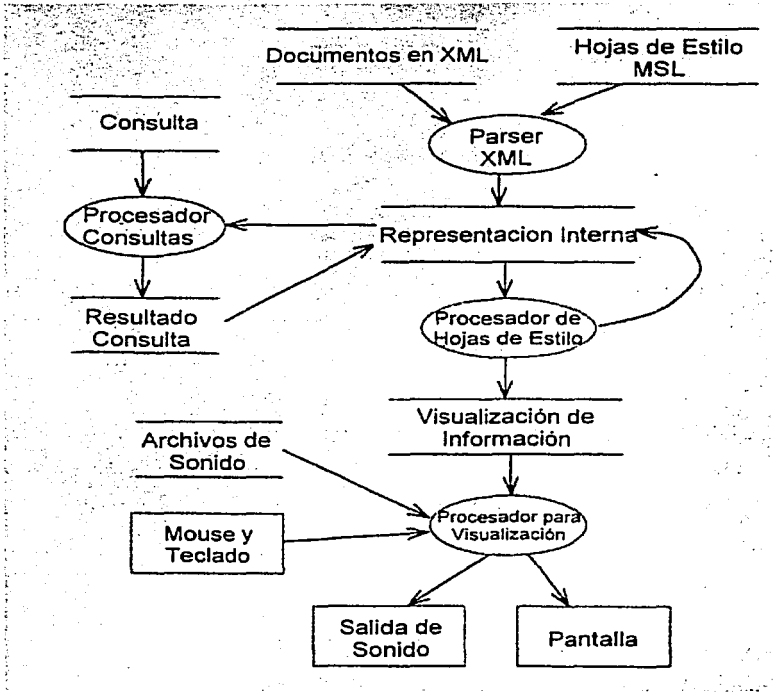


Figura 3.3: Arquitectura de MATE.

3.3.1 Base de datos interna

En la arquitectura de MATE se encuentra una base de datos, la cual contiene los documentos en XML. El modelo abstracto de los datos que usa es un grafo de anotación (Figura 3.4), y cada grafo puede ser representado en la base de datos.

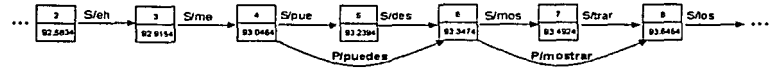


Figura 3.4: Ejemplo de un Grafo de Anotación.

El grafo está formado por nodos (representados gráficamente con un rectángulo) y arcos (representados por flechas) que relacionan a los nodos. Los nodos tienen la información temporal de las unidades lingüísticas, y los arcos, el tipo de información lingüística y la etiqueta o símbolo correspondiente a la unidad. Por ejemplo, en el nodo 4 se indica el inicio de la pronunciación de una palabra en el milisegundo 93.0464, y el nodo 6 indica el final en el milisegundo 93.3474. La etiqueta *P/puedes* que tiene el arco entre los nodos 4 y 6, indica que la información corresponde a una palabra (P) cuyo símbolo es "puedes". De forma similar *S/pue* indica que la información corresponde a una sílaba (S) cuyo símbolo es "pue".

Cada nodo del grafo tiene un nombre (p.e. palabra, fonema, etc), un conjunto de atributos y una lista de nodos hijo. Cualquier nodo puede tener varios nodos padre; pero ese nodo debe de tener por lo menos un nodo padre. El nodo padre, puede ser usado para tratar al grafo de anotación como un conjunto de árboles. Esta forma de ver el grafo es necesaria para aquellos algoritmos que necesitan una estructura arborescente; p.e. el escribir la salida de la representación interna en un conjunto de archivos, aplicar el procesador de hojas de estilo, hacer preguntas acerca del orden lineal de dos elementos, etc.

La interpretación del modelo de datos es flexible y puede variar dependiendo de los requerimientos del esquema de transcripción que se usen. La interpretación de un nodo y de la relación padre-hijo puede ser; por ejemplo, que un nodo representa el inicio de un segmento de voz y su hijo representa una subdivisión más fina del mismo segmento de voz.

Es fácil entender cómo son interpretados, por el modelo de datos, los archivos codificados en XML. Una vez que se lee un archivo XML en la base de datos, cada elemento y cada pieza de texto del XML, es representado como un nuevo nodo; este nuevo nodo va a ser del tipo de nombre que tiene el elemento XML. Todos los atributos del XML se copian (si es necesario, tomando los valores del DTD del XML). El nodo padre, va a ser el nodo del elemento XML textualmente adjuntado y la relación padre-hijo de los nodos se va a derivar de la inclusión textual de los archivos XML. Todo lo anterior da como resultado un conjunto de

árboles (uno por cada archivo cargado).

Los elementos de un XML que tienen atributos *href*, presentan otro tipo de relación padre-hijo, a través del uso de ligas. Los atributos *href* son evaluados para que den una representación interna de las relaciones, den una lista de otros nodos, y para añadan los nuevos nodos (como hijos), al nodo de origen.

Algo importante que es necesario hacer notar acerca del diseño de la arquitectura de MATE, es que ésta es reflexiva; es decir, toda la información del sistema se almacena en la representación interna en un formato similar. Por ejemplo, los resultados de una consulta y las hojas de estilo se almacenan en el mismo formato.

3.3.2 Lenguaje y procesador de consultas

Todo lo que pueda ser transcrito debe ser recuperado y cualquier combinación de información codificada (texto y voz) se debe poder buscar. Sin embargo, toda la información que se puede buscar, debe ir acompañada de una salida estructurada de los datos. Para satisfacer los requerimientos presentados anteriormente, MATE desarrolló un procesador de consultas (*Query for MATE*, Q4M), el cual se puede usar para todas las trascripciones de un corpus, codificadas en XML. Las razones [Mckelvie et al., 2000] por las que se desarrolló el Q4M son las siguientes:

1. Cuando se empezó a desarrollar MATE no existía ningún módulo de consultas de XML escrito en java.
2. Tampoco existía un lenguaje de consultas estándar para XML.
3. Se necesitaba que el procesador de consultas interactuara muy bien con la base de datos interna.
4. Se requería de un modelo de datos extendido (un grafo de anotación) y no solamente una estructura arborescente.

A través del Q4M se pueden realizar principalmente dos tipos de consultas: simples y complejas. Las primeras consisten en buscar un elemento directamente en un documento XML. Las consultas complejas consisten en buscar elementos en dos o más documentos relacionados de forma indirecta; estas consultas comprenden la búsqueda de la información en dos o más niveles. En el capítulo 6 se describen ampliamente los dos tipos de consultas.

A continuación se presenta un ejemplo de una consulta compleja. Se quiere buscar en la base de datos la siguiente información:

Encontrar todos los adverbios que incluyan un acento "H" dichos por Jorge, que se encuentren después de una respuesta hecha por Laura.*

	MATE	Explicación
(1)	($p pros)	$p hace referencia a los elementos <pros>
(2)	($s sent)	$s hace referencia a los elementos <sent>
(3)	($w word);	$w hace referencia a los elementos <word>
(4)	($s type ~ "ans") &&	el atributo *type* de <sent> tiene el valor "ans" Y
(5)	($s who ~ "Laura") &&	el atributo *who* de <sent> tiene el valor "Laura" Y
(6)	($s][$w) &&	<sent> precede inmediatamente a <word> Y
(7)	($w pos ~ "adv") &&	el valor de *pos* en <word> es "adv" Y
(8)	($w who ~ "Jorge") &&	el valor de *who* en <word> es "Jorge" Y
(9)	($w @ &p) &&	los elementos <pros> ocurren durante <word> Y
(10)	($p type ~ "H*")	el valor de *type* en <pros> es "H*"

Tabla 3.1: Ejemplo de una consulta en el lenguaje de MATE.

La tabla 3.1 muestra el equivalente de la consulta en el lenguaje de MATE. La consulta tiene una parte para la definición de variables (1-3) y una para limitar la consulta (4-10). Lo que se hace primero es declarar las variables que se van a usar, en este caso se declararon tres variables ($p, $s y $w). Después de la declaración de variables se crean las expresiones que van a restringir la consulta. En la tercera columna de la tabla 3.1 se presenta la explicación de cada una de las expresiones.

En una consulta se pueden utilizar operadores matemáticos, operadores para cadenas, comodines, operadores para agrupar, y todas las expresiones dentro de una consulta se pueden combinar a través del uso de operadores lógicos (en este ejemplo se usó "&&"). Se pueden utilizar expresiones regulares en vez de nombres para los elementos, con el fin de denotar un rango de nombres.

Los operadores permiten realizar consultas más específicas de la información que se pretende encontrar. Con el uso de varibales y operadores se puede completar una expresión de consulta. La tabla 3.2 muestra los operadores que se pueden hacer con el lenguaje de consultas de MATE.

El resultado de una consulta con Q4M es almacenada como una nueva estructura dentro de la representación interna; desde donde, puede ser accesada para que el usuario la visualice o pueda ser posteriormente manipulada dentro de la arquitectura de MATE.

Descripción	Ejemplo	Operadores	Explicación
Comparación de los elementos por el valor de sus atributos			
a una cadena	($a tipo ~ "V")	- !-	igual, no es igual
valor numérico	($a inicio < 0.2)	< <=	menor o igual que
		> >=	mayor o igual que
		==, !=	igual, no es igual
relativo a otros valores			
para cadenas	($a tipo ~ $b tipo)	- !-	igual, no es igual
para números	($a inicio > $b inicio)	< <=	menor o igual que
		> >=	mayor o igual que
		==, !=	igual, no es igual
algún cambio	($a f0 > $b f0 *2)	+ - * /	operaciones matemáticas
posición relativa a otros elementos			
en una jerarquía	($a ^ $b)	^	es el padre de
en una secuencia	($a << $b)	, <<	es vecino directo/se encuentra a la izquierda
respecto al tiempo	($a [[$b)	% [[ ]] []][// @	relaciones de tiempo

Tabla 3.2: Operaciones en el Lenguaje de MATE.

MATE provee una herramienta para la formulación interactiva de consultas (vea sección 3.3.5).

3.3.3 Lenguaje y procesador de hojas de estilo

Cuando se empezó a desarrollar MATE, el lenguaje XSL (que permite transformar XML's) no se encontraba completamente definido, por lo que se implementó un propio lenguaje de transformación al que se le denominó MSL (*MATE Stylesheet Language*).

MSL usa el lenguaje de consultas de MATE (Q4M) que permite que las hojas de estilo accesen información de un DTD.

A la especificación de las transformaciones escritas en XSL o MSL se le conoce como "hoja de estilo"; cada hoja de estilo consiste de una o más plantillas (*templates*) y cada plantilla contiene una descripción contra la cual las etiquetas del documento de entrada pueden coincidir. Y tiene un conjunto de instrucciones a seguir, siempre y cuando exista una coincidencia. (Ver figura 3.5)

Al ejecutar el procesador, se debe especificar una hoja de estilo en MSL y un

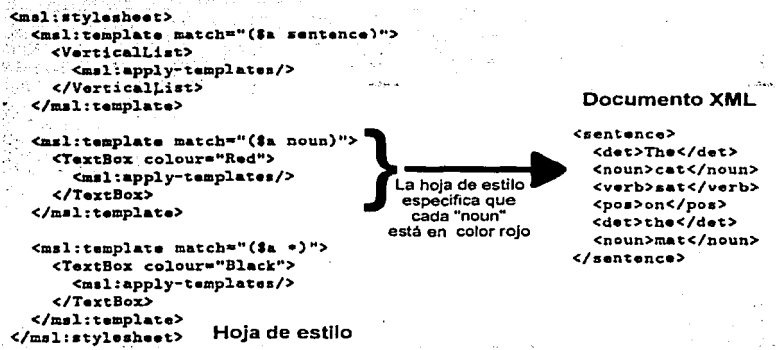


Figura 3.5: Ejemplo de una hoja de estilo asociada a un documento XML.

documento o una serie de documentos que se vayan a transformar. Normalmente el procesador de hojas de estilo crea una estructura de visualización, que después es utilizada por el pocesador de visualización (sección 3.3.4) para mostrarla al usuario.

3.3.4 Procesador para la visualización

MATE cuenta con un número limitado de objetos implementados que se usan para la visualización, entre ellos encontramos: listas, tablas, menus,texto y un reproductor de sonido. Sin embargo, la arquitectura fue construida con la intención de permitir que se escriban nuevas clases en java que ayuden a visualizar nuevos objetos.

3.3.5 Interfaz de usuario

Existen varias ventanas de interfaz de usuario, entre las cuales se encuentran las siguientes:

- **Menú de proyectos.** Cuando se ejecuta MATE aparece una ventana con la lista de proyectos disponibles. (Ver figura 3.6). El usuario puede escoger un proyecto que puede mandar a ejecutar. Cuando se ejecuta el proyecto aparece una nueva interfaz, la cual puede contener un reproductor de sonido o varias partes de texto. En el capítulo 5 se hace una descripción más amplia.

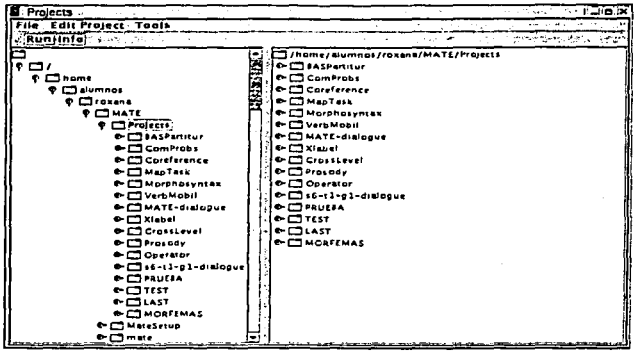


Figura 3.6: Proyectos en MATE.

- **Procesador de consultas.** Éste se puede ejecutar de dos formas: 1) Se selecciona el proyecto del cual se va a hacer la consulta y desde el menú *Tools* (en la ventana de inicio), se selecciona la opción *Query Window*; y 2) Cuando se ha ejecutado un proyecto, desde la hoja de estilo de éste, se puede abrir la ventana de consultas haciendo *clic* a la opción *Query Window* que aparece dentro del menú *Tools*. Estos dos métodos muestran una ventana de consultas, la cual guía interactivamente al usuario en el proceso de hacer una consulta.(Ver figura 3.7). Este proceso se describe ampliamente en el capítulo 6.

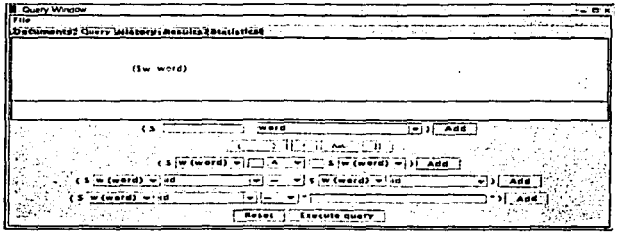


Figura 3.7: Ventana para hacer consultas.

- **Reproductor de audio.** La arquitectura de MATE incluye un programa para la reproducción de audio, a partir de archivos en formato *u-law*.

3.4 Conclusiones

En este capítulo se describió el concepto de bases de datos semiestructuradas. Se describió que el lenguaje XML (*eXtensible Markup Language*) comparte muchos conceptos con las bases de datos semiestructuradas, y que se ha estado usando como un estándar en la transcripción de un corpus.

También se describió la herramienta de MATE, la cual utiliza al lenguaje XML como el formato de transcripción de cada corpus que se encuentra en su base de datos. MATE está integrada por una arquitectura la cual comprende cinco componentes principales: 1) una base de datos interna, 2) el lenguaje y el procesador de consultas, 3) el lenguaje y el procesador de hojas de estilo, 4) el procesador para la visualización y 5) la interfaz de usuario. Estos componentes fueron descritos en este capítulo.

En el siguiente capítulo se presenta el modelo jerárquico del corpus DIME el cual organiza a través de niveles y relaciones jerárquicas todas las transcripciones del corpus.

Capítulo 4

Modelo Jerárquico del Corpus DIME

Para crear sistemas conversacionales es necesario tener información sobre cómo sucede una conversación, es por eso que la grabación de conversaciones reales resulta indispensable para analizarla y para extraer información lingüística útil.

4.1 Información Lingüística

Un corpus de voz es un conjunto de grabaciones, así como un conjunto de comentarios y documentos referentes a los datos de voz [Gibbon et al., 1997]. Uno de los documentos referentes son las transcripciones, las cuales consisten en un conjunto de etiquetas analíticas de algunos aspectos lingüísticos; por ejemplo, en la indicación de frase subordinada, en aspectos pragmáticos, aspectos gramaticales, etc. [Barbosa et al., 1998].

A partir de la creación de un corpus de voz se puede obtener bastante información lingüística de él. La información lingüística se genera cada vez que el audio se transcribe con distintos tipos de información. Se pueden generar transcripciones ortográficas, fonéticas, prosódicas, gramaticales, etc. El número de transcripciones que se generen va a depender del propósito del corpus.

Las transcripciones se utilizan, entre otras cosas, para investigar a fondo los fenómenos lingüísticos que ocurren dentro de un corpus. Estos fenómenos son la materia de estudio de la lingüística computacional, la intención de estudiarlos es modelar con la ayuda de una computadora los fenómenos que ocurren, de esta forma se pretende trabajar con datos que permitan reproducir con la máxima fidelidad las características del objeto de estudio.

El contar con diversas transcripciones permite tener un corpus más completo, a partir del cual se puede explotar la información. Además de que permite

la creación de mejores sistemas de síntesis, reconocimiento y entendimiento del habla porque con toda la información que contiene, los sistemas se pueden basar en muchos más aspectos para poder realizar mejor sus funciones. Por ejemplo, en los sistemas de reconociento de voz se podría obtener un mejor porcentaje de reconocimiento si el sistema pudiera diferenciar a través de un análisis sintáctico que las palabras "ola" y "hola" tienen distinto significado, aunque se pronuncien igual.

En la siguiente sección se va a explicar el tipo de información lingüística que compone al Corpus DIME.

4.2 El corpus DIME

Como se mencionó en el capítulo 2 el corpus DIME es un corpus de voz que cuenta con algunas transcripciones. En la actualidad todos los diálogos que componen al corpus DIME están transcritos a nivel textual. La tarea de transcripción consistió en segmentar los diálogos, esto es, dividir cada diálogo grabado en *pequeñas unidades completas (expresiones).*

El corpus está compuesto por 31 diálogos, cada diálogo contiene información lingüística implícita. Un diálogo es la unidad lingüística de información más general. Cada diálogo fue generado por dos interlocutores (el *mago* y el *sujeto*).

Los diálogos están estructurados por *turnos*. Los turnos son aquellas intervenciones que tuvo un interlocutor en el diálogo con otro interlocutor. Durante un turno un participante puede contribuir con una o varias *expresiones*.

Como se mencionó anteriormente, una *expresión* es una *unidad que expresa una idea o intención*. El objetivo de una *expresión* es el comunicar una propuesta (una orden, una oferta, una promesa, una afirmación, etc.) Sin embargo, durante un diálogo de habla espontánea las expresiones tienen una función por sí solas; por ejemplo, admitir que un mensaje fue recibido (con un gesto), el mantener un turno, el expresar un estado emocional (por medio del uso de interjecciones), etc. Es importante mencionar que existe una gran complejidad en distinguir las fronteras de las *expresiones*, por lo que la tarea de segmentación de los diálogos fue difícil de realizar. Por esta complejidad se tomaron en cuenta algunos criterios para identificar las fronteras de las expresiones [Villaseñor et al., 2001]:

- Un cambio en el tono de la voz puede marcar el inicio de una nueva expresión.
- Cuando inicia una categoría sintáctica importante (una oración o una frase nominal).
- Cuando ocurre una pausa o una respiración considerable. Aunque, pueden existir expresiones largas con varios silencios.

- Un turno está compuesto por lo menos de una expresión (p.e. un simple *ok*).
- Una expresión debe preservar los fenómenos locales del lenguaje oral. También, debe considerar ruidos como interjecciones de duda, sonidos con los labios, etc.

En la figura 4.1 se presenta un extracto del diálogo 6, donde se puede ver la estructura de éste a traves de turnos, representados por "*s*" y "*u*", el primero hace alusión al *mago* y el segundo al *sujeto*. Cada turno está compuesto por una o varias expresiones, las cuáles se pueden distinguir por medio de las letras *utt* que representan a la palabra en inglés *utterance*; éstas van seguidas de un número consecutivo.

```
utt10 : s:  okey
utt11 :     quieres que mueva este objeto <sil> hacia acá ?
utt12 : u:  no
utt13 :     hacia el otro lado
utt14 : s:  okey
utt15 :     este objeto hacia acá ?
utt16 : u:  sí
utt17 : s:  okey
utt18 :     <ruido>
utt19 :     ahí está bien ? <no-vocal>
utt20 : u:  un poco menos por favor
utt21 : s:  okey
utt22 :     <ruido>
utt23 :     ahí está bien ?
utt24 : u:  ahí está bien
utt25 :     después <sil> me puedes poner <sil> [e]l el extractor de aire encima de la <sil> de la estufa
utt26 : s:  okey
utt27 :     <ruido> <no-vocal> así está bien ?
utt28 : u:  sí
utt29 :     así está bien
utt30 :     <no-vocal> eh me puedes mostrar los tipos de muebles que tengo ?
utt31 : s:  okey
```

Figura 4.1: Extracto del diálogo 6 del Corpus DIME.

En este trabajo de tesis se procesó una parte del diálogo 6 y se identificaron varios tipos de información lingüística. A continuación se enumeran estos tipos:

1. Información pragmática
2. Información léxica
3. Información sintáctica
4. Información morfológica.
5. Información fonológica.
6. Información fonética.

En las siguientes secciones se describen estos tipos de información y su anotación correspondiente en un archivo de transcripción. Antes de mencionar en que consiste cada una de estas transcripciones, se explica el formato que tienen los archivos de transcripción.

4.2.1 Formato general de los archivos de transcripción

Los distintos archivos de transcripción cuentan con la misma estructura. Por ejemplo, se presenta a continuación la transcripción ortográfica de la expresión 30 del diálogo 6 "*eh me puedes mostrar los tipos de muebles que tengo*":

```
#
92.425406 115 .bn
92.583406 115 .ls
92.915406 115 eh
93.046406 115 me
93.347406 115 puedes
93.646406 115 mostrar
93.814406 115 los
94.159406 115 tipos
94.216406 115 de
94.576406 115 muebles
94.677406 115 que
95.066406 115 tengo
95.509406 115 .bn
```

La primera línea de la transcripción anterior (#) indica el fin del encabezado, que en este caso no se describe, ya que es información opcional que no fue incluida en las transcripciones. Las siguientes líneas del archivo están divididas en tres columnas. La primera columna consiste en las *unidades de tiempo*, estas unidades corresponden al archivo de audio de la expresión 30. La siguiente columna representa el color que se va a usar cuando se vea su transcripción. La última columna son etiquetas que contienen la información lingüística, en este caso las etiquetas son palabras que describen la información ortográfica.

Es importante mencionar que todas las demás transcripciones que se presentan en este capítulo, contienen la misma estructura de la información.

En las siguientes secciones se describen los demás tipos de información lingüística.

4.2.2 Información pragmática

La información pragmática consiste en distinguir lo que quiere decir una persona. Cotidianamente las personas usan oraciones como "*¿me puedes decir la hora?*",

cuando lo que realmente quieren decir es: " *Dame la hora por favor*". Se tiene otro ejemplo en la siguiente oración: "*¿podrías quitar el peine de la mesa?*", donde lo que se quiere decir es "*quita el peine de la mesa*".

Para una computadora es complicado distinguir las intenciones reales que tienen las personas al decir alguna oración, es por esto que es necesario identificar y transcribir la información pragmática que se puede presentar dentro de un corpus.

La información pragmática se estudia en el corpus DIME, a partir de la identificaión de frases verbales que aparecen en una expresión. Una frase comprende una secuencia de palabras que no constituyen una oración completa.

La transcripción de las frases verbales consiste en distinguir la intencón de estas frases. La intención se describe a través del uso de una etiqueta, en este caso se usan verbos en infinitivo.
P.e. la expresión 30[1] del diálogo G tiene la frase siguiente:

me puedes mostrar

La intención de la frase se etiquetó con el verbo *mostrar*. (Ver tabla 4.1)

Información léxica →	me puedes mostrar	← frase
Información pragmática →	mostrar	← intención de la frase

Tabla 4.1: Frase Verbal.

La figura 4.2 ilustra las transcripción anterior. Esta figura contiene 2 tipos de transcripciones. La primera línea de la figura corresponde a la forma de onda[2] del archivo de audio de la expresión 30. La segunda línea presenta la transcripción a nivel de expresiones. La tercera corresponde a la transcripción de frase verbal.

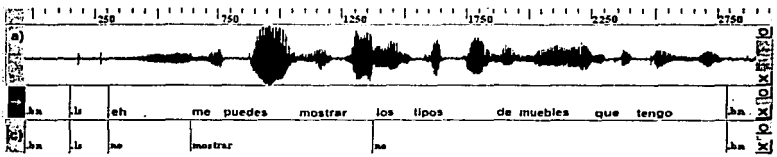


Figura 4.2: Identificación de frases. a)Forma de onda de la señal de voz, b)Transcripción a nivel de expresiones, c)Identificación de frases verbales.

[1]En los siguientes ejemplos cada vez que se mencione a la expresión 30, se hace referencia a aquella expresión que pertenece al diálogo G del corpus DIME.

[2]La forma de onda es la imagen gráfica de un sonido.

4.2.3 Información léxica

A partir de la información léxica que se encuentra en un corpus, se pueden crear diccionarios léxicos. En estos diccionarios se estudian el conjunto de palabras que conforman un idioma, o el conjunto de palabras que se utilizan en un lugar en específico como en el área de "Polanco" o en el área de "Tepito" de la Ciudad de México. Así como el conjunto de palabras que se usan para "el diseño de una cocina", etc.

Para este tipo de información es necesario realizar transcripciones ortograficas de un corpus. La transcripción ortográfica consiste en dividir una expresión en palabras. Es decir, los segmentos de voz de un diálogo se transcriben palabra por palabra.

P.e. se tiene la expresión 30 a continuación:

```
utt30 : eh me puedes mostrar los tipos de muebles que tengo ?
```

La figura 4.3 ejemplifica la transcripción ortográfica del archivo de audio de la expresión 30. En la parte superior de la figura se encuentra la forma de onda y debajo de éste su transcripción a nivel de palabras.

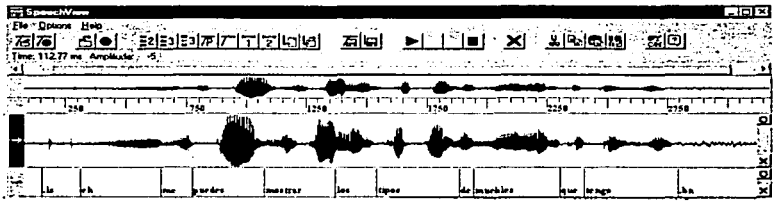


Figura 4.3: Transcripción ortográfica.

El archivo de transcripción a nivel de palabras de la expresión queda de la siguiente forma:

```
#
92.425406 115 .bn
92.583406 115 .ls
92.915406 115 eh
93.046406 115 me
93.347406 115 puedes
93.646406 115 mostrar
93.814406 115 los
94.159406 115 tipos
```

```
94.216406 115 de
94.576406 115 muebles
94.677406 115 que
95.066406 115 tengo
95.509406 115 .bn
```

A partir de la transcripción ortográfica se pueden obtener varias transcripciones. Por ejemplo, cuando se tiene una palabra, ésta se puede dividir en sílabas; se puede determinar la estructura de ésta, diviéndola en morfemas; se pueden identificar las clases de palabras (categorías gramaticales); etc. Por esta razón las transcripciones que se explican en las siguientes subsecciones se van a ejemplificar partiendo de la transcripción ortográfica.

4.2.4 Información sintáctica

La información sintáctica comprende tres tipos de transcripciones:

- Frase clítica.
- Transcripción de categorías gramaticales.
- Identificación de verbos y su conjugación.

La estructura sintáctica de una oración indica la manera en que las palabras se relacionan unas con otras dentro de ésta. Además esta estructura indica cómo se agrupan las palabras para formar frases, qué palabras modifican otras palabras y qué palabras son de central importancia en la oración.

La estructura sintáctica puede identificar los tipos de relaciones que existen entre frases y puede almacenar información adicional acerca de la estructura de la oración.

En las siguientes subsecciones se explican cada una de las transcripciones.

Frase clítica

La transcripción de frase clítica consiste en detectar aquella frase que contenga algún clítico. Un clítico es un pronombre átono como *me, se, te, lo, etc.* Cuando se encuentre una frase de éste tipo, ésta se va a etiquetar como *frase_clitica*.
P.e. se tiene la frase clítica:

me puedes mostrar

Información léxica →	me puedes mostrar	← frase
Información sintáctica →	frase_clítica	← categoría sintáctica

Tabla 4.2: Frase Clítica.

El clítico de esta frase es me. En la tabla 4.2 se muestra la relación entre la transcripción ortográfica y la transcripción de frase clítica.

La figura 4.4 ilustra las transcripción anterior. Esta figura contiene 2 tipos de transcripciones. La primera línea de la figura corresponde a la forma de onda. La segunda línea presenta la transcripción a nivel de expresiones. La tercera corresponde a la identificación de frase clítica (la cual está marcada por la etiqueta *frase_clitica*).

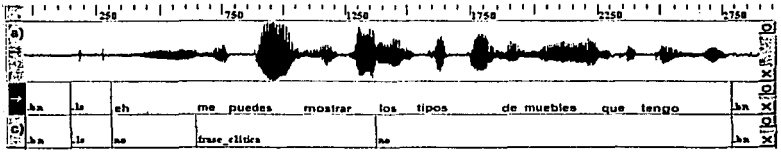


Figura 4.4: Identificación de frases. a)Forma de onda de la señal de voz, b)Transcripción a nivel de expresiones, c)Transcripción de frases clíticas.

Categorías gramaticales

Existe otro tipo de transcripción, el de las categorías gramaticales, esta transcripción se genera a partir de la transcripción ortográfica. Todas las palabras pueden clasificarse en categorías gramaticales (clases de palabras), esto se hace dependiendo de su estructura, de la función que desempeñen y de su significado. Las clases que se usaron para describir a cada una de las palabras fueron: *sustantivo, adjetivo, artículo, pronombre, verbo, adverbio, preposición, conjunción e interjección.*

En la figura 4.5 se muestran la transcripción ortográfica y las categorías gramaticales de las palabras de la expresión 30. En esta figura se puede ver que la palabra *los* es un artículo (*art*), que *tipos* es un sustantivo (*sust*), que *de* es una preposición (*pre*), etc.

A continuación se presenta la transcripción gramatical de la expresión 30 (la misma expresión que se usó para ejemplificar la transcripción ortográfica)[3]:

[3]Para los siguientes ejemplos que se presentan en las demás transcripciones, siempre van a ser haciendo uso de la expresión 30 del diálogo 6.

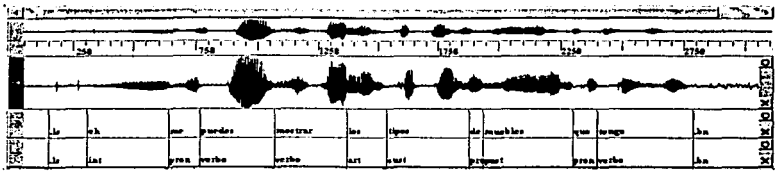


Figura 4.5: Categorías gramaticales.

```
#
92.425406 115 .bn
92.583406 115 .ls
92.915406 115 int
93.046406 115 pron
93.347406 115 verbo
93.646406 115 verbo
93.814406 115 art
94.159406 115 sust
94.216406 115 prep
94.576406 115 sust
94.677406 115 pron
95.066406 115 verbo
95.509406 115 .bn
```

En esta transcripción se preservan las mismas unidades de tiempo que en la transcripción ortográfica. Sin embargo, se puede distinguir en la tercera columna distintas etiquetas que representan la información de categoría gramatical. En la tabla 4.3 se presenta la correspondencia de las etiquetas con las categorías gramaticales.

Categorías	Etiqueta
interjección	*int*
pronombre	*pron*
verbo	*verbo*
artículo	*art*
sustantivo	*sus*
preposición	*pre*
adjetivo	*adj*
adverbio	*adv*
conjunción	*conj*

Tabla 4.3: Clases de palabras y sus etiquetas correspondientes.

La transcripción de categorías gramaticales se usa para saber qué pronombres se utilizan en el corpus, con qué frecuencia se usan; cuáles verbos son los más usados, con que frecuencia, etc.

Identificación de verbos y su conjugación

Esta transcripción consiste en identificar los verbos que se encuentran en una expresión. Cuando se encuentra un verbo, se trascriben dos tipos de información:

1. **Verbos**. Primero se identifica el "verbo en infinitivo" a partir de la forma ortográfica. P.e. si la forma ortográfica es *puedes*, el verbo correspondiente es *poder*. La figura 4.6 muestra la transcripción ortográfica y los verbos en infinitivo.

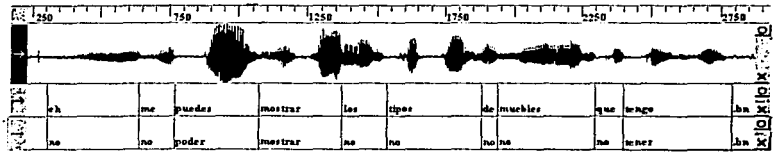


Figura 4.6: Verbos en infinitivo.

   Este tipo de transcripción permite identificar perífrasis verbales. Una perífrasis verbal consiste en utilizar una frase para decir lo que podría expresarse con una palabra [Beristain, 2000]. Las perífrasis verbales están compuestas por un "*verbo conjugado*", seguido de un "*verbo en infinitivo*"; por ejemplo "*puedes mostrar*".

2. **Conjugación**. Esta consiste en poner la conjugación del verbo identificado. P.e. si el verbo identificado es *tengo*, y este verbo está conjugado en *la primera persona del singular (1s) en el tiempo presente (pres), y si el verbo es intransitivo (i)*, la etiqueta que le corresponde es **1s-pres-i**.

   En la figura 4.7 se muestra la conjugación de los verbos.

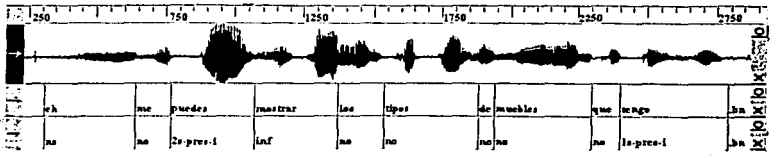


Figura 4.7: Identificación de verbos.

4.2.5 Información morfológica

La transcripción morfológica consiste en distinguir la estructura interna de las palabras y sus procesos de formación.

Las palabras están formadas por pequeñas unidades que tienen significado; estas unidades se llaman *morfemas*, y no necesariamente coinciden con las sílabas [Mungía et al., 1998].
Por ejemplo:

cocin-a
alacen-as
blanc-o

Las palabras anteriores tienen dos morfemas:

1. El morfema raíz: **cocin-**, **alacen-**, **blanc-**. Generalmente la raíz se mantiene invariable y porta el significado de la palabra.
2. El morfema flexivo o derivativo [Mungía et al., 1998], **-a, -as, -o.** Éste siempre varía y agrega el significado de género, número, tiempo, etc.

Existen algunas palabras en el corpus DIME que están compuestas de un solo morfema:

me
eh

La figura 4.8 presenta la transcripción morfológica de una parte de la expresión 30. En la parte superior se encuentra la forma de onda, debajo de ésta la transcripción a nivel de palabras. La siguiente línea es la transcripción morfológica, dode se puede ver que una palabra, p.e., *tipos* está compuesta por los morfemas "*tip*" y "*os*". Por último se encuentran las categorías de los morfemas, éstas consisten en distinguir los "morfemas raíz" (etiquetados como *raiz*) y los "morfemas derivativos" (etiquetados como "*prefijo, sufijo o infijo*"). En el ejemplo anterior el morfema "*os*" fue etiquetado con el valor de *sufijo*, ya que éste se encuentra al final del morfema *raíz* (*tip*).

A continuación se presenta un fragmento del archivo de transcripción morfológica y la transcripción que indica la categoría de los morfemas:

```
93.276406 115 pued          93.276406 115 raiz
93.347406 115 es            93.347406 115 sufijo
93.606406 115 mostr         93.606406 115 raiz
```

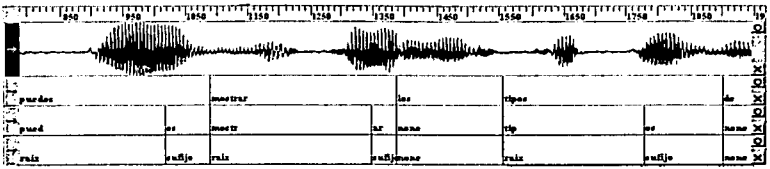


Figura 4.8: Transcripción morfológica.

```
93.646406 115 ar
93.814406 115 none
94.037406 115 tip
94.159406 115 os
```

```
93.646406 115 sufijo
93.814406 115 none
94.037406 115 raiz
94.159406 115 sufijo
```

4.2.6 Información fonológica

La transcripción fonológica consiste en la representación de una expresión a través de fonemas. Los hablantes de cualquier lengua utilizan sonidos lingüísticos articulados para formar palabras. En la escritura, estos sonidos se representan por letras. Los fonemas son esos sonidos que se pueden diferenciar de otros en una lengua.

En la figura 4.9 se ilustran las transcripciones ortográfica, fonológica y fonética.

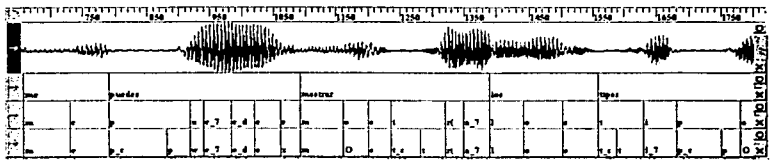


Figura 4.9: Transcripciones fonética y fonológica.

Se presenta a continuación la transcripción fonológica de la frase "*me puedes mostrar*" de la expresión 30.

```
92.986406 115 m
93.046406 115 e
93.174406 115 p
93.197406 115 u
93.240406 115 e_7
93.276406 115 e_d
93.317406 115 e
```

```
93.347406 115 s
93.417406 115 m
93.457406 115 o
93.492406 115 s
93.576406 115 t
93.606406 115 r(
93.646406 115 a_7
```

donde *e_7* significa que la vocal "e" está acentuada. *a_7* indica que la vocal "a" está acentuada.

4.2.7 Información fonética

Dentro de este tipo de información se encuentran dos tipos de transcripciones:

- Transcripción fonética.
- Transcripción silábica.

Transcripción fonética

A través de la transcripción fonética se puede analizar la realización física de los sonidos; es decir, cómo se producen, cómo se perciben y como están formadas las ondas sonoras. Está transcripción está compuesta por alófonos. Un *alófono* es una variante que se da en la pronunciación de un fonema, según su posición en una palabra, en una sílaba, según el carácter de los fonemas vecinos, etc. (Ver figura 4.9).

Por ejemplo: el fonema *u*, puede estar represenatado por el alófono *w*. El fonema *s* por el alófono *z* o *s*.

La transcripción fonética de "*me puedes mostrar*" es la siguiente:

```
92.986406 115 m
93.046406 115 e
93.137406 115 p_c
93.174406 115 p
93.197406 115 w
93.240406 115 e_7
93.276406 115 e_d
93.317406 115 e
93.347406 115 z
93.417406 115 m
93.457406 115 O
```

```
93.492406 115 s
93.538406 115 t_c
93.576406 115 t
93.606406 115 r(
93.646406 115 a_7
```

Transcripción silábica

La transcripción silábica es otro tipo de información que se encuentra en el modelo propuesto. La transcripción consiste en dividir en sílabas las palabras de las expresiones de un diálogo. Las palabras pueden estar compuestas por una o más sílabas. En toda sílaba debe de existir por lo menos una vocal.

La transcripción silábica de la expresión 30 se puede ver en la figura 4.10.

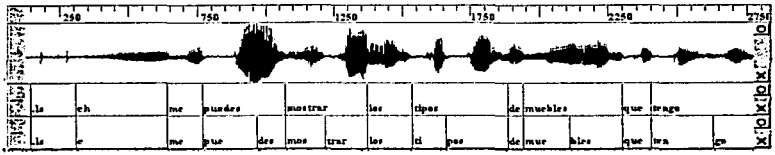


Figura 4.10: Transcripción silábica.

A continuación se presenta una parte de la transcripción silábica de esta expresión:

```
#
92.425406 115 .bn
92.583406 115 .ls
92.915406 115 e
93.046406 115 me
93.239406 115 pue
93.347406 115 des
93.492406 115 mos
93.646406 115 trar
93.814406 115 los
```

En esta seción se explicaron los tipos de información lingüística del corpus DIME sin presentar las relaciones que hay entre ellos. Los distintos tipos de transcripción se encuentran almacenados cada uno en un archivo. En las explicaciones de cada una de las transcripciones que se presentó anteriormente, se mencionaron algunas relaciones existentes entre las transcripciones. Principalmente, la

transcripción ortográfica, es aquella transcripción que se puede relacionar con la mayoría de las transcripciones que se presentaron aquí.

Como se mencionó en el capítulo 1, el propósito de esta tesis es organizar al corpus DIME a través de la creación de un modelo jerárquico que relacione adecuadamente a todos los tipos de información lingüística que se necesiten.

En la siguiente sección se presenta el modelo que se propone para organizar la información y se explican las relaciones jerárquicas que se establecieron en él.

4.3 Relaciones Jerárquicas

La relación directa que existe entre dos niveles de transcripción se representa a través de flechas. Existen otras relaciones (indirectas), las cuales no se representan de ninguna forma; pero por el simple hecho de que un nivel se encuentre arriba de otro, puede existir una relación de este tipo. Todos los niveles que se encuentren en la parte superior del modelo están integrados por los niveles inferiores.

La primera parte de la información lingüística que se tiene comprende tres niveles: los *diálogos*, los *turnos* y las *expresiones*. Estos niveles se pueden relacionar de la siguiente forma: *un diálogo está compuesto por varios turnos; los turnos a su vez, están compuestos por una o varias expresiones.* La relación entre estos tipos de información se muestra en la figura 4.11.

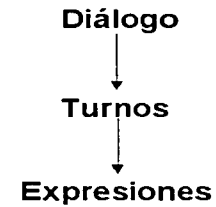


Figura 4.11: Primeras relaciones jerárquicas.

P.e. El diálogo 6 está formado por 107 turnos y el turno 11 está formado por las expresiones 17, 18 y 19 (figura 4.12).

```
utt17 : s:  okey
utt18 :     <ruido>
utt19 :     ahí está bien ? <no-vocal>
```

Figura 4.12: Expresiones del turno 11 (diálogo 6).

La siguiente relación que se estableció fue entre las expresiones y las frases. Cada expresión puede contener una o más frases dentro de ella. En la información lingüística del corpus se tienen dos tipos de frases, frases verbales y frases clíticas. La relación de las expresiones con los tipos de frases se muestra en la figura 4.13.

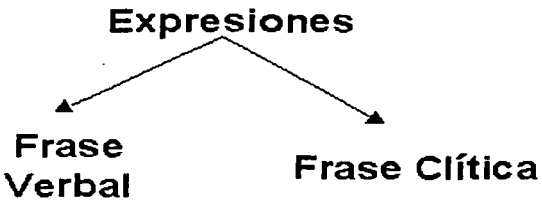


Figura 4.13: Expresiones y frases.

Cuando se unen varias palabras, éstas pueden formar frases y expresiones. La siguiente relación que se presenta aquí es entre las frases y las palabras (ver figura 4.14). Esta relación hace posible que los niveles que se encuentren en la parte superior del modelo estén relacionados con los niveles inferiores.

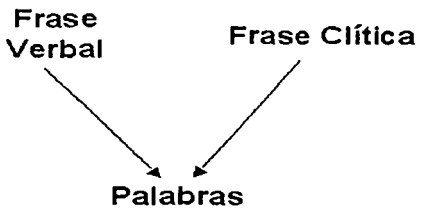


Figura 4.14: Frases y palabras.

A partir del nivel de palabras se crearon cuatro relaciones.

- Palabras con Categorías Gramaticales.
- Palabras con Verbos en infinitivo.
- Palabras con Morfemas.
- Palabras con Sílabas.

A continuación se explica cada uno.

4.3.1 Palabras con Categorías Gramaticales

Cada una de las palabras pertenece a cierta categoría gramatical, la cual se etiqueta según la clase de palabra que le corresponda.

Por ejemplo:

cocinas -> *sustantivo*

La relación se presenta en la figura 4.15.

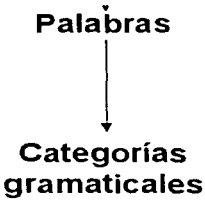


Figura 4.15: Palabras y categorías gramaticales.

4.3.2 Palabras con Verbos en Infinitivo

Aquellas palabras que funcionen como verbo, se etiquetan con su verbo en infinitivo correspondiente. Además del infinitivo del verbo, es necesaria la información de la conjugación del verbo que se relacionó como se muestra en la figura 4.16. (Ver sección 4.2.5).

Figura 4.16: Palabras y verbos.

La palabra y la conjugación del verbo no están directamente relacionadas como se puede ver en la figura 4.16; los verbos en infinitivo se encuentran de por medio. Sin embargo el modelo permite relacionar imdirectamente la información sobre la conjugación con las palabras.

4.3.3 Palabras con Morfemas

Las palabras están compuestas por uno o más morfemas, y estos morfemas pertenecen a una categoría de morfemas; que asocia a tipos de morfemas con características comunes. En la figura 4.17 se muestra el modelo para relacionar estas unidades.

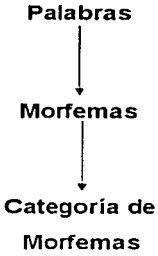


Figura 4.17: Palabras, morfemas y categoría de morfemas.

4.3.4 Palabras con Sílabas

Se establece una relación entre las palabras y las sílabas para representar que todas las palabras están compuestas por una o más sílabas, las sílabas por fonemas y éstos por alófonos.

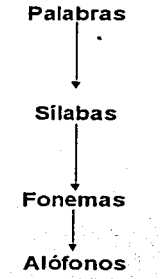


Figura 4.18: Palabras,sílabas, fonemas y alófonos.

Se podía establecer una relación directa entre las palabras y los fonemas o entre las palabras y los alófonos, pero este tipo de modelo sería redundante. Así que la

figura 4.18 expresa el modelo más simple y general con las relaciones deseadas.

Con el modelo que se establece en esta sección es posible relacionar indirectamente palabras con alófonos, o palabras con fonemas.

En esta sección se fueron describiendo por partes las relaciones jerárquicas existentes entre cada uno de los tipos de transcripción. El modelo jerárquico completo se ilustra en la figura 4.19.

Es importante mencionar que a partir de este modelo se pueden relacionar las expresiones con los fonemas, sin la necesidad de establecer su relación explícitamente en el modelo. De esta misma forma se pueden relacionar otras transcripciones, una que pertenezca a un nivel superior del modelo y la otra a uno intermedio o inferior; sin la necesidad de estar directamente relacionadas.

4.4 Conclusiones

En este capítulo se describieron los tipos de información lingüística que puede tener un corpus de voz. Para el corpus DIME se pretende tener distintos tipos de información; esta información lingüística se genera cada vez que la reproducción de un audio se transcribe a distintos tipos de información. Se pueden generar transcripciones ortográficas, fonéticas, prosódicas, gramaticales, etc.

En este capítulo se presentó el modelo jerárquico que se propone para relacionar los diversos tipos de información lingüística del corpus DIME; y se explicaron todas las relaciones que se establecieron en él. Este modelo jerárquico permite tener una adecuada organización de la información del corpus.

En este trabajo de tesis se va a utilizar la arquitectura planteada por MATE para validar computacionalmente el modelo jerárquico que se propone en esta tesis. (Ver Apéndice A). El modelo contiene las fuentes de conocimiento lingüístico necesarias, las cuales están relacionadas entre sí; lo que permitirá la organización del corpus DIME en una base de datos.

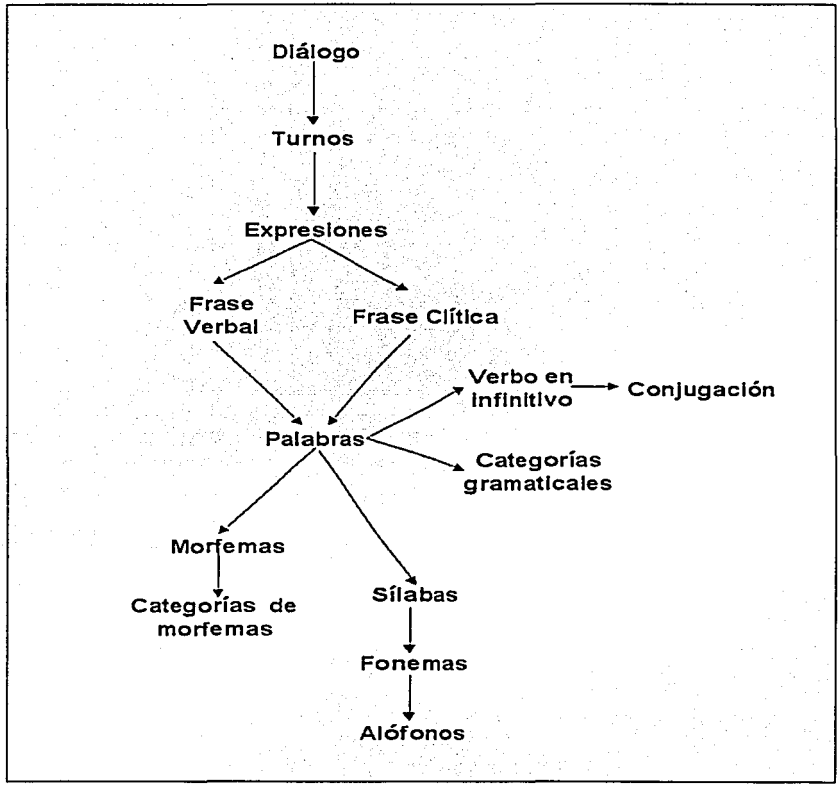


Figura 4.19: Modelo Jerárquico.

Capítulo 5

Implementación del Modelo

En este capítulo se describe la implementación del modelo jerárquico propuesto. La implementación se tuvo que hacer a través de modelos lineales. A partir del modelo jerárquico de DIME se obtuvieron cinco modelos lineales.

5.1 Modelos Lineales

La herramienta de MATE solo permite implementar modelos lineales; las relaciones jerárquicas que se establecen entre dos niveles del modelo solamente se pueden relacionar de padre a hijo y es imposible tener relaciones de padre a hijos. Esta relación de padre a hijo se hace a través de un atributo (*href*) que se emplea dentro de un archivo XML. (Vea sección 5.3).

Para implementar el modelo jerárquico en MATE es necesario descomponerlo en un conjunto de modelos lineales; de esta forma se obtuvieron cinco modelos lineales. Estos modelos incluyen diferentes niveles de información lingüística. Todos los modelos lineales cuentan con los tres niveles más generales del modelo jerárquico principal; estos niveles corresponden a los *diálogos*, los *turnos* y las *expresiones*. (Ver figura 5.1).

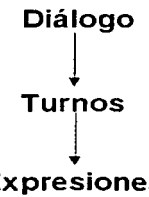


Figura 5.1: Niveles más generales del corpus DIME.

A continuación se presentan los cinco modelos lineales que se obtuvieron:

1. **El Modelo de "Transcripciones".** Este modelo lineal (figura 5.2) es el más general. Incluye las transcripciones más comunes que, casi siempre, tiene un corpus de voz. A partir de las expresiones, se incluyen a las palabras, las sílabas, los fonemas y los alófonos.

Figura 5.2: El modelo de "Transcripciones".

2. **El Modelo de "Morfemas".** Este modelo contiene la información de los morfemas y de las categorías de los morfemas. (Ver figura 5.3).

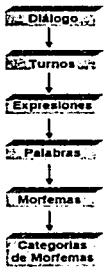


Figura 5.3: El modelo de los Morfemas.

3. **El Modelo "Cat-gra".** El modelo "Cat-gra" contiene las categorías gramaticales de las palabras. En este modelo se pueden obtener las categorías gramaticales del diálogo completo, de una expresión, de algún turno o de cada una de las palabras (figura 5.4).

Figura 5.4: Modelo de las Categorías Gramaticales.

4. **El Modelo "Clíticos".** En este modelo (figura 5.5) se pueden encontrar todas aquellas frases que contengan algún clítico.

Figura 5.5: El modelo de las frases clíticas.

5. **El Modelo "Intención-Verbal".** El modelo presenta todas las frases con intención verbal, así como todos los verbos que se hayan identificado. A partir de la identificación de algún verbo se puede obtener su conjugación o el verbo en infinitivo que le corresponde.(Ver figura 5.6).

El proceso de implementación de los modelos lineales consiste de cinco pasos:

1. Conversión de las transcripciones a XML
2. Implementación de las relaciones jerárquicas.
3. Creación de los DTD's.
4. Creación de las Hojas de Estilo.
5. Creación y organización de los proyectos.

En las siguientes secciones se explica cada uno de los pasos de este proceso.

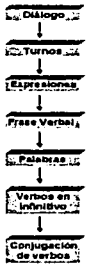


Figura 5.6: Frases con Intención Verbal.

5.2 Conversión de las transcripciones a XML

El primer paso consistió en codificar las transcripciones de la expresión 30 del diálogo 6 del corpus DIME al formato de XML. Un archivo en XML describe la información que contiene cualquier documento. Por medio del uso de XML's se pretende describir la información de los archivos de transcripción. (Ver Apéndice B y C).

El corpus DIME está transcrito en su totalidad solamente a nivel de expresiones. Para poder implementar el modelo jerárquico propuesto en la herramienta de MATE, se transcribió la expresión 30 del diálogo 6 a los distintos tipos de transcripción que se encuentran en el modelo como, ortográfico, silábico, morfológico, fonético, etc.

Todos los archivos de transcripción que se obtuvieron presentan la misma estructura en su información. Por ejemplo se tiene la transcripción ortográfica de la expresión 30, del diálogo 6:

```
#
92.425406 115 .bn
92.583406 115 .ls
92.915406 115 eh
93.046406 115 me
93.347406 115 puedes
93.646406 115 mostrar
93.814406 115 los
94.159406 115 tipos
94.216406 115 de
94.576406 115 muebles
94.677406 115 que
```

```
95.066406 115 tengo
95.509406 115 .bn
```

Esta transcripción está divida en 3 columnas. La primera columna consiste en las *unidades de tiempo*, estas unidades corresponden al archivo de audio de la expresión 30. La siguiente columna representa el color que se va a usar cuando se vea su transcripción. La tercera contiene la información lingüística de la transcripción que se trate, en este caso contiene palabras.

Puede verse la representación de la transcripción ortográfica en el siguiente grafo de anotación (Ver figura 5.7). Este grafo muestra cómo está relacionada la información dentro de un archivo de transcripción.

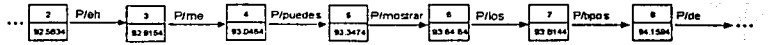


Figura 5.7: Grafo de anotación de la transcripción ortográfica

En la figura 5.8 se presenta la codificación en XML de la transcripción ortográfica de la expresión 30 del diálogo 6.

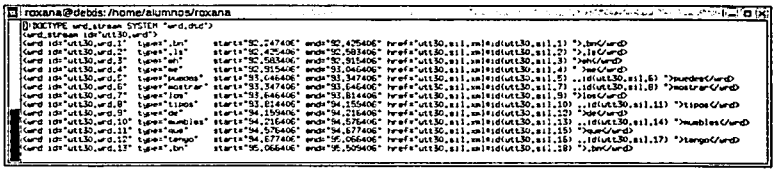


Figura 5.8: XML de la transcripción ortográfica.

La primera línea del archivo XML de la transcripción ortográfica corresponde a la declaración de un DTD, es decir se dice que este XML va acompañado del DTD llamado "wrd.dtd". En la siguiente línea se declara el elemento *wrd_stream*. Para la declaración de elementos dentro de un documento XML se puede usar cualquier nombre, generalmente se usan aquellos nombres relacionados al tipo de información que se quiere describir. En el caso de este XML, que corresponde a la transcripción ortográfica, se declara el elemento *wrd_stream* que significa "palabra raíz". Para los demás tipos de transcripción, se utilizaron nombres relacionados al tipo de transcripción correspondiente; p.e. para los fonemas se usó *fon_stream*, para los alófonos *phn_stream*, para las sílabas *sil_stream*, etc.

El elemento *wrd_stream* contiene al atributo *id* el cual, es el identificador único de este elemento. En la figura 5.8 se puede ver que el atributo *id* tiene el valor de "*utt30.wrd*".

La tercera línea y las consecutivas presentan un formato de la información

muy similar. En todas estas líneas se declara el elemento *wrd*, el cual contiene cinco atributos. Lo que hace diferentes a cada una de estas líneas es el valor que contienen sus atributos.

Cada tipo de transcripción tiene un elemento único que se pone al inicio de cada segmento; para las palabras se usa *wrd*, para las sílabas se usa *sil*, para los fonemas se usa *fon*, para los alófonos se usa *phn*, etc. Todos estos elementos contienen cinco atributos: *id, type, start, end* y *href*.

A continuación se explican cada uno de los atributos. Se presentan ejemplos de cada uno de los atributos a partir del archivo de la transcripción ortográfica:

- **id.** Este atributo permite definir un identificador único para la información que se encuentra dentro de un elemento *wrd*. Es decir, se define un identificador por cada palabra. Los identificadores están enumerados de forma consecutiva.
- **type.** El valor asignado a este atributo corresponde a la información lingüística del tipo de transcripción. La información que se guarda en este atributo es la información contenida en la tercera columna del archivo de transcripción correspondiente. Si se trata del archivo que contiene la transcripción ortográfica, el valor que va a tener almacenado el atributo *type* va a ser una palabra.

  Por ejemplo, se tienen las siguientes cuatro líneas del archivo de transcripción ortográfica:

```
92.915406 115 eh
93.046406 115 me
93.347406 115 puedes
93.646406 115 mostrar
```

  La codificación en XML de la información lingüística de la transcripción ortográfica anterior se muestra en la figura 5.9.

```
roxana@deixis: /home/alumnos/roxana
<wrd id="utt30.wrd.3"  type="eh"      > </wrd>
<wrd id="utt30.wrd.4"  type="me"      > </wrd>
<wrd id="utt30.wrd.5"  type="puedes"  > </wrd>
<wrd id="utt30.wrd.6"  type="mostrar" > </wrd>
```

Figura 5.9: Segmento de la codificación de una transcripción ortográfica.

En el extracto de la figura 5.9 se puede ver cómo el atributo *type* contiene la misma información lingüística de la transcripción ortográfica. También, se

puede ver que el atributo *id* contiene valores que se encuentran enumerados de forma consecutiva.

- **start.** El atributo *start* contiene unidades de tiempo. Este atributo indica el tiempo en milisegundos en el que se inició la pronunciación de la unidad lingüística correspondiente.

- **end.** Este atributo, como el atributo *start*, contiene unidades de tiempo. El atributo *end* contiene la unidad de tiempo en la que se finalizó la pronunciación de la unidad lingüística.

  A continuación se ejemplifican el atributo *start* y el atributo *end*.

  Continuando con el ejemplo del archivo de la transcripción ortográfica, se tiene el siguiente extracto de esta transcripción:

```
92.915406 115 eh
93.046406 115 me
93.347406 115 puedes
93.646406 115 mostrar
```

  En la primera columna de la transcripción ortográfica anterior se puede notar que el atributo *start* contiene el valor de las unidades de tiempo. Sin embargo, el valor que contiene el atributo *start* corresponde a las unidades de tiempo de una línea anterior. El atributo *end* contiene el valor de las unidades de tiempo del archivo de transcripción, correspondientes a la misma línea en la que se encuentra la palabra.

  Por ejemplo; 92.915406 corresponde al tiempo final de la palabra "*eh*" y al tiempo inicial de la palabra "*me*". Mientras que 93.046406 corresponde al tiempo final de "*me*"y al tiempo inicial de "*puedes*".

  En la figura 5.10 se observa que la palabra "*me*" se comenzó a pronunciar en el milisegundo 92.915406 y se terminó en el 93.046406.

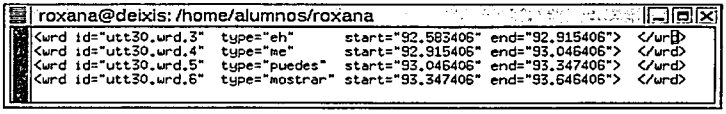


Figura 5.10: XML de la transcripción ortográfica.

- **href** En el atributo *href* se indica la relación jerárquica entre las transcripciones. Casi todas las transcripciones que se encuentran en un modelo lineal, contienen el atributo *href* (en su XML correspondiente). Las transcripciones que no contienen este atributo son aquellas que se encuentran en el nivel inferior del modelo. P. e., en el modelo lineal de los "Morfemas", el nivel que corresponde a la transcripción "*Categorías de morfemas*", no

tiene el atributo *href* en su archivo XML porque este nivel no se relaciona con ningún otro. En la siguiente sección se describe con más detalle en que consiste este atributo.

5.3 Implementación de las relaciones jerárquicas

El atributo *href* permite mantener relaciones padre-hijo entre los niveles de transcripción de un mismo modelo lineal. Por ejemplo, en el modelo lineal del proyecto "Transcripciones" (figura 5.11), las flechas representan la relación que existe entre cada nivel de transcripción. Es decir, las palabras hacen referencia a las sílabas a través del atributo *href*; las sílabas (del mismo modo que las palabras) hacen referencia a los fonemas; los fonemas a los alófonos y los alófonos que se encuentran en el nivel inferior, no tienen definida una relación a otro nivel. En este caso el archivo XML de la transcripción fonética no contiene el atributo *href*.

Figura 5.11: El modelo de "Transcripciones".

La declaración del atributo *href* dentro de un XML se hace de la siguiente forma:

$$href = "A\#I_1..I_n"$$

donde A es el archivo XML que se va a relacionar; I_1 ..I_n son los atributos *id* que se van a referenciar. Ejemplo:

href="utt30.sil.xml#id(utt30.sil.16) ..id(utt30.sil.17)"

Por ejemplo se quiere relacionar la información que se presenta en la figura 5.12, la cual contiene la transcripción ortográfica y la transcripción silábica.

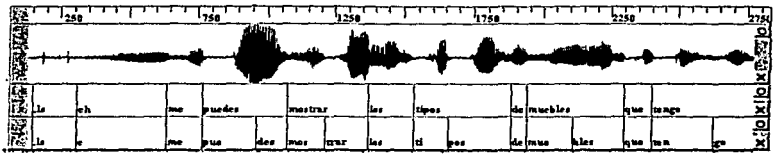


Figura 5.12: Transcripción silábica.

En la figura 5.13 se puede ver claramente la relación existente entre estos dos niveles de transcripción (ortográfico y silábico).

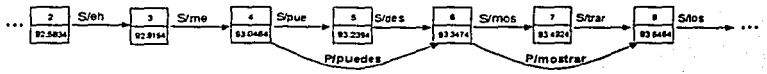


Figura 5.13: Grafo de anotación que muestra las relaciones entre 2 niveles de transcripción: ortográfica y silábico.

La figura 5.14 muestra un ejemplo de cómo se utilizan las referencias entre distintos niveles de transcripción. Esta figura muestra la relación que existe entre cuatro niveles de transcripción del modelo "Transcripciones", comprende los niveles que contienen a los alófonos, a los fonemas, a las sílabas y a las palabras.

En esta figura se puede ver que el nivel inferior corresponde a los alófonos (utt30.phn.xml), y que el documento no contiene el atributo *href*. El siguiente nivel de transcripción es el de los fonemas (utt30.fon.xml), su XML hace referecia al XML de los alófonos a través del atributo *href*. La relación se hace específicamente a los valores que contienen el atributo *id* del archivo XML de los alófonos (utt30.phn.xml). Todas las relaciones que se presentan en esta figura se establecen referenciando a los atributos *id* del nivel inmediato inferior. En el nivel inmediatamente superior al nivel de los fonemas, se encuentran las sílabas quienes hacen referencia al XML de los fonemas. De la misma, forma el XML de las palabras (utt30.wrd.xml) hace referencia al de las sílabas (utt30.sil.xml)

La manera de relacionar los niveles de transcripción de cada uno de los modelos lineales, se hizo utilizando al atributo *href* de la misma forma que se acaba de presentar.

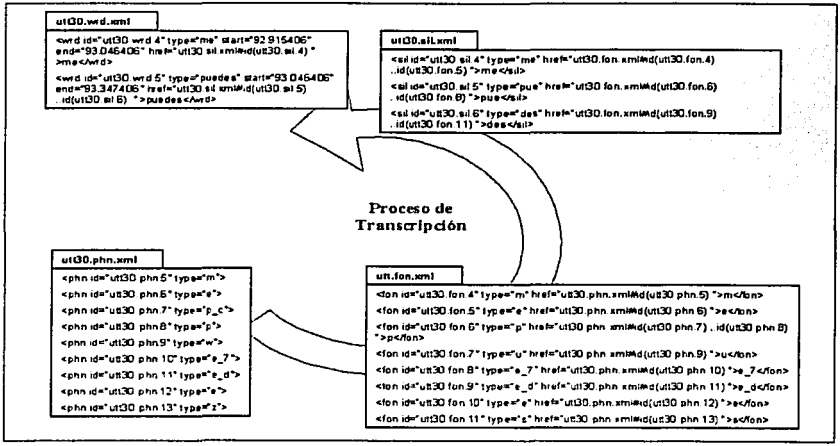


Figura 5.14: Ejemplo del uso del atributo *href* en los documentos XML.

5.4 Creación de los DTD's

Después de haber creado los archivos XML para cada uno de los niveles de transcripción, se empezaron a crear los archivos DTD. Se creó un archivo DTD para cada uno de los archivos en XML existentes.

En la figura 5.15 se presenta el archivo DTD que valida al documento XML (figura 5.8) de la transcripción ortográfica.

Las tres primeras líneas del archivo DTD son comentarios, los cuales siempre se deben de escribir entre <!-- y -->. Después de los comentarios el DTD está compuesto por la siguiente información:

- ELEMENT. Se usa cuando se declaran los elementos que contiene un archivo XML. Por ejemplo, en el DTD ("wrd.dtd") el primer *ELEMENT* que se declara es el *wrd_stream*; este elemento está formado por secuencias de cero o más palabras (elementos *wrd*). El elemento *wrd* se declara posteriormente, y se especifica que debe de contener texto.
- ATTLIST. *ATTLIST* se usa para declarar los atributos que contiene un elemento. Por ejemplo, el elemento *wrd_stream* contiene al atributo *id*. Por su parte el elemento *wrd* contiene varios atributos, *id, type, href, start* y *end*.

roxana@deixis: /h...ts/Transcripciones

```
<!-- DTD  de las palabras  wrd.dtd  UTT30 -->
<!-- Creado por Roxana P.M. -->


<!-- ABBREVIATIONS -->

<!ENTITY % id
          'id ID #REQUIRED'
>

<!ENTITY % href
          'href       CDATA #IMPLIED
           xml:link   CDATA #FIXED   "simple"
           show       CDATA #FIXED   "embed"
           actuate    CDATA #FIXED   "auto"'
>

<!ENTITY % start
          'start      CDATA #IMPLIED'
>

<!ENTITY % end
          'end        CDATA #IMPLIED'
>

<!ELEMENT wrd_stream (wrd)*>
<!ATTLIST wrd_stream
          id ID  #REQUIRED>

<!ELEMENT wrd  (#PCDATA)>
<!ATTLIST wrd
          %id;
          type       CDATA      #IMPLIED
          %href;
          %start;
          %end;>
```

Figura 5.15: wrd.dtd

- **ENTITY.** Se usa para declarar entidades. Las entidades son variables que se usan para definir las características que va a tener un atributo. Para este ejemplo se declararon cuatro entidades, *id, href, start* y *end.* Las entidades se deben declarar al inicio del DTD (antes de la declaración de los elementos).

5.5 Creación de las Hojas de Estilo

Las hojas de estilo se aplican a los archivos XML para mostrar la información que contienen éstos a través de colores, tipos de letra, tamaños de letra, etc. Una hoja de estilo le da formato a un XML.

La figura 5.16 muestra un fragmento del diálogo 6, al archivo XML de este diálogo se le aplicó una hoja de estilo muy sencilla, sin color ni formato, la cual tiene la única función de desplegar la información del archivo XML.

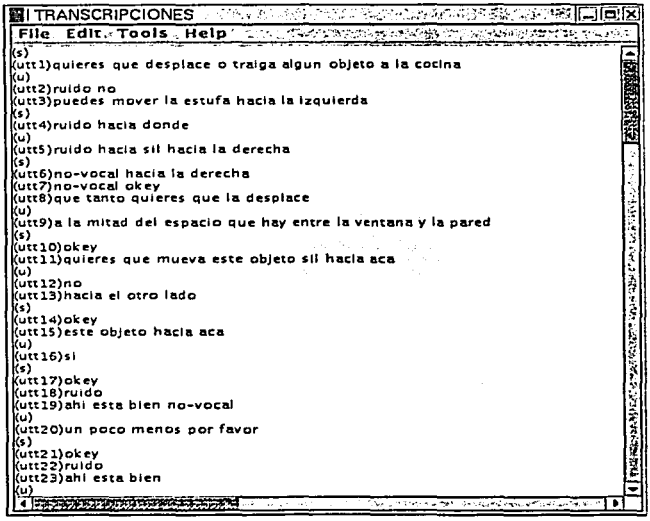


Figura 5.16: Hoja de estilo sencilla aplicada al archivo XML del diálogo 6.

La figura 5.17 presenta el mismo extracto del diálogo 6. En esta ocasión al XML se le aplicó otra hoja de estilo. El nuevo estilo consiste en distinguir las expresiones de cada turno. Los turnos se encuentran de color rojo y el número

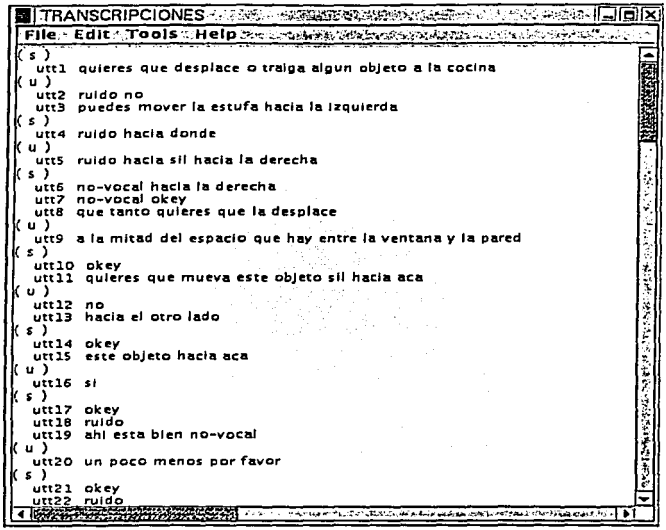


Figura 5.17: Otra hoja de estilo aplicado al mismo XML del diálogo 6.

de cada expresión en color azul.

Al aplicar las hojas de estilo a los documentos XML del corpus DIME, la información muestra con diferentes colores y formato de texto para distinguir entre lo que decía el "mago" (s) y lo que decía el "sujeto" (u).

En la siguiente sección se explica la implementación de los modelos lineales en 5 proyectos de MATE. Estos proyectos organizan la información del corpus DIME en la base de datos de MATE.

5.6 Creación y organización de los proyectos

La información del corpus DIME, fue almacenada en la base de datos de MATE. La herramienta de MATE organiza la información que contiene a través de proyectos. Para el corpus DIME se crearon cinco proyectos, uno para cada modelo lineal que se tiene. Los proyectos que se crearon fueron "Transcripciones", "Morfemas", "Cat-Gram", "Intencion-Verbal" y "Cliticos".

Un proyecto en MATE consiste de:

- varios archivos XML,
- archivos DTD que validan cada archivo XML,
- un archivo *MSL*, que es la hoja de estilo que se le va a aplicar al archivo XML.
- un archivo *MP* y
- un archivo *RUN*.

Los archivos XML, los DTD y las hojas de estilo se explicaron en las secciones anteriores. A continuación se explican los archivos *MP* y *RUN*.

5.6.1 El archivo *MP*

Este archivo contiene el nombre de todos los archivos XML que se encuentren dentro del mismo proyecto, es decir, son todos aquellos archivos xml que pertenecen al mismo modelo lineal. También contiene el nombre de la hoja de estilo (*MSL*) asociada al proyecto, y el nombre del archivo *RUN*.

La figura 5.18 corresponde al contenido del archivo MP (*trans.mp*). Este archivo pertenece al proyecto "Transcripciones".

Todos los archivos que contiene el archivo *trans.mp* se van a cargar cuando se ejecute el proyecto "Transcripciones". Este proyecto se ejecuta desde la interfaz *menú de proyectos*. (Ver figura 3.6)

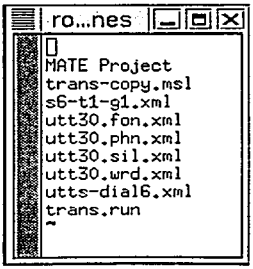


Figura 5.18: Archivo *trans.mp* del proyecto "Transcripciones".

5.6.2 El archivo *RUN*

El archivo *RUN* contiene una descripción de las características de la ventana que mostrará la información lingüística. En este archivo se define el tamaño y el título de la interfaz de usuario que va a aparecer cuando se ejecute un archivo *MP*. Se define también el archivo XML que va a aparecer en la interfaz de usuario (s6-t1-g1.xml), y la hoja de estilo que se le va a aplicar a este archivo en XML (trans-copy.msl).

En la figura 5.19 se presenta el archivo *trans.run*; el cual pertenece al proyecto "Transcripciones".

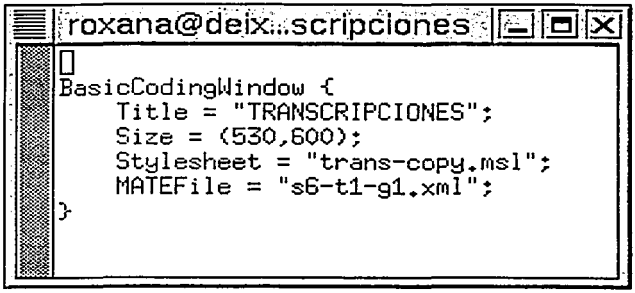


Figura 5.19: Archivo *trans.run*

Todos los proyectos en MATE deben de contar con todos estos archivos (por lo menos un archivo XML y un archivo DTD). Si faltara alguno de estos archivos en un proyecto, éste no podrá ejecutarse, ninguna ventana de interfaz de usuario aparecerá, y ninguna consulta se podrá hacer.

En cada uno de lo proyectos correspondientes se almacenaron los archivos en XML de cada una de las transcripciones, los archivos DTD para cada uno de los XML, la hojas de estilo *MSL*, el archivo *MP* y el archivo *RUN*.

A continuación se presentan, a través de figuras, cada modelo lineal con su proyecto correspondiente. En cada una de las figuras se puede ver cuáles archivos están asociados a cada proyecto.

En la figura 5.20 se presenta el modelo "Transcripciones". Del lado izquierdo se muestra el modelo lineal, del lado derecho la estructura del proyecto.Se puede ver que cada archivo XML tiene asociado un archivo DTD. Cada archivo XML está asociado a un nivel del modelo, excepto el archivo *utts.dial6.xml*. Este XML se ecuentra asociado a tres niveles de transcripción; sin embargo, dentro del XML se consevan las jerarquías entre los niveles.

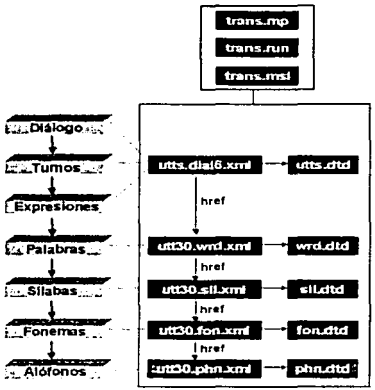


Figura 5.20: Proyecto "Transcripciones".

Las jerarquías se conservan de la siguiente forma: (Ver figura 5.21)

```
<turn id="turn.1" who="s">
     <utt id="utt1">
          quieres que desplace o traiga algun objeto a la cocina
     </utt>
</turn>

<turn id="turn.2" who="u">
     <utt id="utt2">
          ruido no
     </utt>
     <utt id="utt3">
          puedes mover la estufa hacia la izquierda
     </utt>
</turn>

<turn id="turn.3" who="s">
     <utt id="utt4">
          ruido hacia donde
     </utt>
</turn>
```

Figura 5.21: Jerarquías dentro de un archivo XML.

La figura 5.21 es un extracto del archivo *utts.dial6.xml* que contiene al diálogo 6. Como se puede ver el *diálogo* contiene varios *turnos* (turn) y éstos a su vez contienen una o varias *expresiones* (utt).

El proyecto además de contener a los archivos XML y a los DTD contiene los archivos *trans.mp*, *trans.run* y a *trans.msl*.

La figura 5.22 corresponde al proyecto de los "Morfemas". La figura contiene,

igual que la figura del proyecto "Transcripciones", de un lado al modelo lineal y del otro la estructura de archivos que contiene su proyecto.

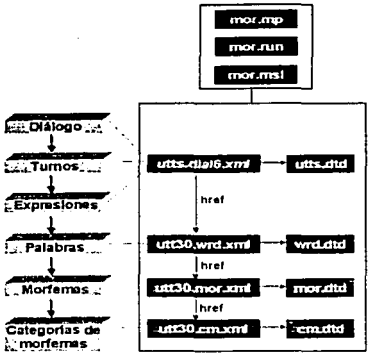


Figura 5.22: Proyecto "Morfemas".

Las siguientes figuras presentan el mismo formato.(Ver figuras 5.23, 5.24 y 5.25).

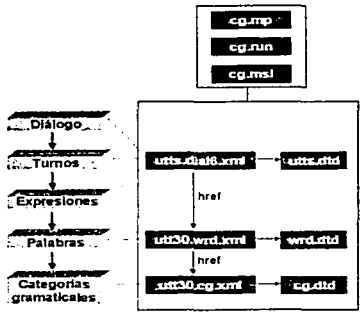


Figura 5.23: Proyecto "Cat-gra".

La implementación del modelo jerárquico del corpus DIME consistió en generar todos los archivos *XML, DTD, MSL, MP y RUN* de cada uno de modelos lineales organizados como proyectos en la base de datos.

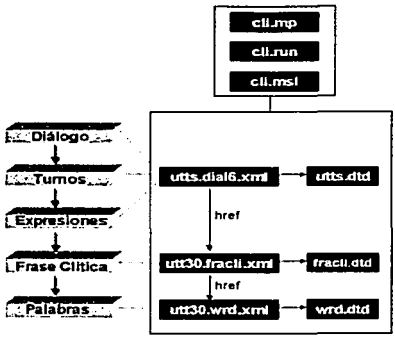


Figura 5.24: Proyecto "Clíticos".

5.7 Conclusiones

En este capítulo se explicó la creación de cinco modelos lineales que se obtuvieron a partir del modelo jerárquico del corpus DIME.

También se explicó la creación de cinco proyectos, "Transcripciones", "Morfemas", "Cat-Gram", "Intencion-Verbal" y "Clíticos", uno por cada modelo lineal. Estos proyectos se crearon para poder organizar el corpus DIME dentro de la base de datos de MATE. Cada proyecto contiene los archivos XML de cada una de las transcripciones correspondientes a la expresión 30 del diálogo 6. Además se crearon los DTD's (uno por cada archivo XML); las hojas de estilo, y los archivos "*MP*" y "*RUN*"; estos últimos permiten cargar y correr un proyecto en la herramienta de MATE.

Después de la creación e integración de cada uno de los proyectos, la herramienta de consulta de MATE está lista para realizar búsquedas y consultas de los archivos transcritos.

En el siguiente capítulo se explica el proceso que se tiene que seguir para realizar una consulta en la herramienta de MATE. Además, se presentan ejemplos de los tipos de consultas que se pueden hacer a la información del corpus DIME que está transcrita.

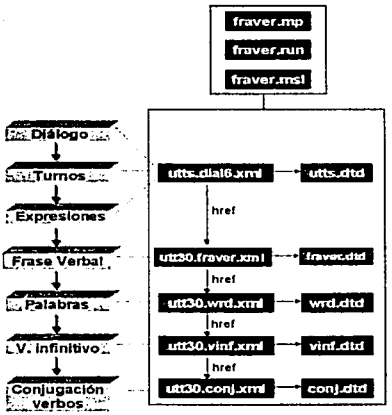


Figura 5.25: Proyecto "Intencion-Verbal".

Capítulo 6

Consultas a la Base de Datos

En este capítulo se explica el proceso necesario para realizar consultas y búsquedas de la información del corpus DIME en la base de datos.

Existen principalmente tres pasos que se tienen que seguir para poder realizar una consulta con la herramienta Q4M de MATE. Los pasos se enumeran a continuación:

1. Cargar un proyecto.
2. Crear la expresión de consulta.
3. Ver los resultados de la consulta.

En las siguientes secciones se explica cada uno de estos pasos.

6.1 Cargar un proyecto

Para cargar un proyecto del corpus DIME en la herramienta de MATE, se tiene que hacer lo siguiente:

1. **Ejecutar MATE.** Este paso consiste en mandar a ejecutar la herramienta de MATE a través del siguiente comando:

   java mate.Workbench

   Cuando se hace esto, aparecen dos ventanas:

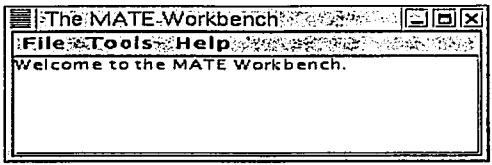


Figura 6.1: Información general.

- La primera ventana que aparece contiene información general de la herramienta de MATE. Esta interfaz ayuda a entender mejor el funcionamiento de MATE. (Ver figura 6.1).
- La segunda, es la ventana del *Menú de Proyectos.* (Ver figura 6.2).

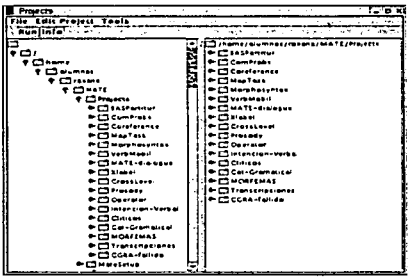


Figura 6.2: Menú de proyectos.

2. **Seleccionar un proyecto.** El segundo paso consiste en seleccionar un proyecto de la lista de proyectos que se encuentra en la ventana de Menú de Proyectos.
   La selección se logra haciendo un *clic* con el *mouse* sobre el nombre del proyecto con el cual se quiere trabajar. Al seleccionar un proyecto aparece una ventana con la lista de documentos asociados al proyecto. A continuación se debe seleccionar el archivo *MP* dándole un *clic* con el *mouse.* En la figura 6.3 se seleccionó el proyecto "Transcripciones" y el archivo *"trans.mp"*.

3. **Presionar el botón Run.** El siguiente paso consiste en cargar el proyecto seleccionado, a través de presionar el botón *Run* que se encuentra en la parte superior izquierda de la ventana.
   Al presionar el botón *Run*, se cargan todo los archivos del proyecto descritos en el archivo *MP* y que aparece una nueva ventana con la hoja de estilo

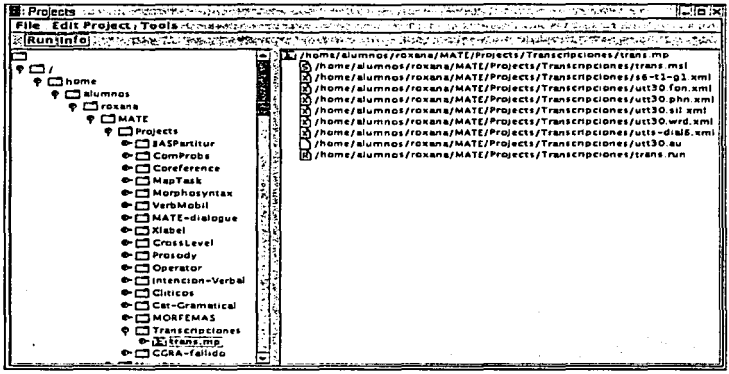


Figura 6.3: Selección del proyecto Transcripciones.

asociada al proyecto seleccionado.

4. **Escoger la ventana de consultas.** Desde la ventana de la hoja de estilo se puede abrir la ventana de consultas, la cual va a permitir hacer búsquedas en el proyecto de la información que se necesite.

   La ventana de la hoja de estilo tiene varios menús, uno de ellos se llama *Tools*. En este menú aparecen dos herramientas que se pueden ejecutar, *Audio Player* y *Query Window*.

   El siguiente paso es seleccionar la opción *Query Window*, para abrir la herramienta de consultas. (Ver figura 6.4).

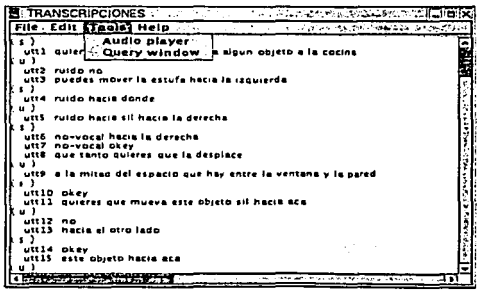


Figura 6.4: Herramientas de MATE.

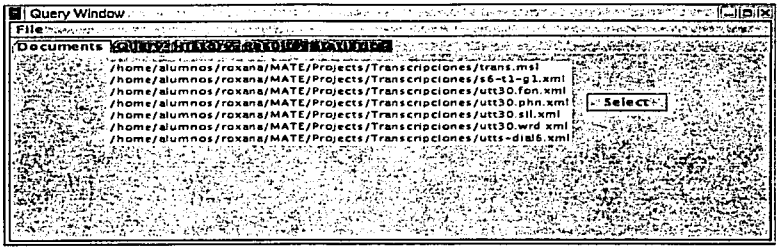


Figura 6.5: La ventana de consultas muestra los documentos XML disponibles para hacer las búsquedas.

6.2 Crear la expresión de consulta

La ventana de la figura 6.5 tiene varias pestañas en su parte superior: *Documents, Query, History, Results y Statistics.* Cada vez que se presiona una pestaña dentro de esta ventana, se ve distinta información. P.e., si se selecciona la pestaña de *Documents* aparece la lista de documentos XML, después de seleccionar los archivos en que se van a realizar consultas, se presiona el botón *Select*, lo cual hace que se active la pestaña *Query* y que se muestre la interfaz para hacer consultas.

En la ventana de consultas se crea la expresión de consulta. Este proceso consiste de tres pasos:

1. **Selección de archivos.** Consiste en seleccionar los archivos que se van a utilizar en la consulta que se va a formular. Si se escoge un sólo archivo, la selección se hace dándole un *clic* al nombre de este archivo, posteriormente se debe de presionar el botón *Select.* Si son dos o más archivos los que se van a usar, la selección se hace dándole un *clic* a uno de los archivos, y con el uso del teclado, específicamente las teclas "*control*" y las "*flechas*", se seleccionan los demás archivos que se van utilizar. Los archivos seleccionados quedan sombreados como se ve en la figura 6.6. Después de la selecciom de archivos se presiona el botón *Select.* Después se activa la pestaña *Query* (fig. 6.7).

   En el ejemplo de la figura 6.6 se presentan los siguientes documentos seleccionados:

   - utt30.fon.xml
   - utt30.phn.xml
   - utt30.sil.xml

- utt30.wrd.xml
- utts-dial6.xml

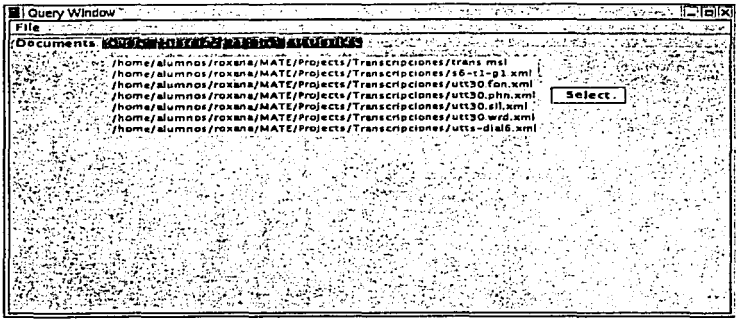


Figura 6.6: Selección de documentos XML.

2. **Crear la expresión de consulta.** Desde la interfaz *Query* (ver figura 6.7) se pueden formular las consultas, ya sea de forma interactiva o escribiendo directamente en el área del texto de la ventana. Las consultas se hacen declarando variables y creando expresiones con las especificaciones del lenguaje de consultas de MATE (ver capitulo 3, sección 3.2.2).

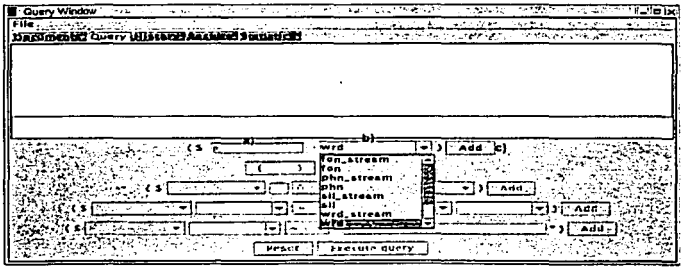


Figura 6.7: Interfaz de consultas.

Para este ejemplo, se quieren buscar "todas aquellas palabras de la expresión 30 que contengan el alófono *e*". La figura 6.8 muestra la expresión de consulta que se va a formular. En la parte superior del área de texto de la ventana aparece la declaración de las variables *p* y *a* que se van a utilizar,

y en la parte inferior aparecen las operaciones que se van a hacer con las variables declaradas.

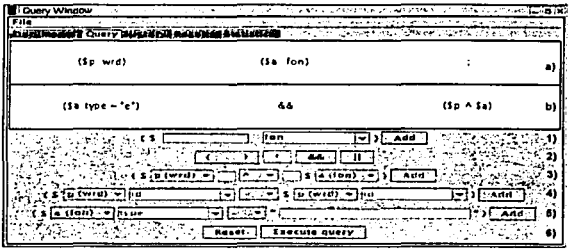


Figura 6.8: Expresión de consulta.

Para declarar una variable de forma interactiva, se debe escribir su nombre en el área de texto señalada en la figura 6.7 a). En el lado derecho del área de texto (figura 6.7 b)), se encuentra una barra de selección con los tipos de elementos XML que se han cargado.Uno de estos tipos debe ser seleccionado para indicar el tipo de variable que se declara. Después se presiona el botón *Add* para que esta parte se añada a la expresión de consulta (figura 6.7 c)).

Después de declarar una variable se tienen dos opciones. Se puede declarar otra variable (como se hizo anteriormente) o se puede seleccionar el tipo de operaciones que se quiere realizar con las variables ya declaradas. Cada vez que se especifique una operación de las filas 3, 4 y 5 del panel (figura 6.8), se presiona el botón *Add* correspondiente para que éstas sean añadidas a la expresión de consulta. Las distintas operaciones que se presentan en la consulta, pueden ser compuestas o anidadas a través del uso de algún operador lógico (&&, !, ||), o de paréntesis ( ), respectivamente.

Para la expresión de consulta de este ejemplo, se declararon dos variables, una del tipo *wrd* ($p) y la otra del tipo *fon* ($a). La primera consiste en los elementos *palabra* de las transcripciones (nivel ortográfico) y la segunda en los elementos *alófono* (nivel fonético).

La declaración de las variables se hizo de la siguiente forma:

($p wrd) ($a fon)

Es conveniente hacer notar que en esta consulta se están involucrando dos niveles distintos de transcripción (ortográfico y fonético), lo cual hace a esta consulta del tipo compleja, ya que la información se busca en dos niveles del modelo jerárquico que no están relacionados directamente.

Luego de la declaración de las variables, se completó la consulta especificando las restricciones y las operaciones de la consulta, utilizando la siguiente expresión:

($a type ~ "e") && ($p ^ $a)

La expresion *($a type ~ "e")*, asigna a la variable "$a" el valor de *e*, es decir, buscará a los alófonos " *e*". Con la expresión *($p ^ $a)* se dice que la información de la variable "p" (*el elemento palabra*) está en un nivel superior que la información de la variable "*a*", y que $p contiene a $a.

3. **Ejecutar consulta** Cuando se ha terminado de escribir la expresión de consulta se presiona el botón *Execute query*, el cual se encuentra en la parte inferior de la ventana (figura 6.8 6)). En caso de que se quiera cambiar la consulta, o exista algún error en la expresión, se puede presionar el botón *Reset*, el cual borrará toda la información de la consulta que se estaba formulando.

   Cuando se presiona el botón *Execute query*, el procesador de consultas de MATE busca la información solicitada.

6.3 Resultado de la consulta

Los resultados de una consulta en MATE aparecen en el formato de XML, dentro de la ventana de consultas. El resultado de la búsqueda del ejemplo que se ha estado presentando en las secciones anteriores, se puede ver en la figura 6.9.

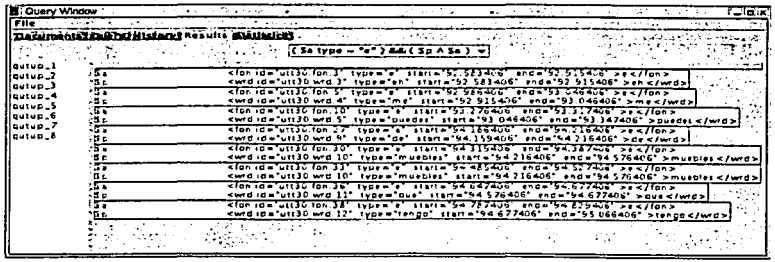


Figura 6.9: Resultados de la consulta.

De la búsqueda que se hizo, se encontraron 7 palabras que contienen al alófono *e*. Las palabras encontradas según el orden de aparición son las siguientes:

eh
me
puedes
de
muebles
que
tengo

Es importante mencionar que en la ventana de resultados aparece la palabra *muebles* dos veces, ya que esta palabra contiene dos alófonos *e*.

Siguiendo los pasos mencionados anteriormente se pueden realizar las consultas en MATE, las cuales pueden ser más complicadas usando varias variables para diferentes niveles de transcripción, y/o con el uso de bastantes operadores para construir expresiones compuestas. Para realizar más consultas dentro del mismo proyecto, se da un *clic* a la pestaña *Query*.

La pestaña *history* muestra todas las expresiones de consulta que se han hecho dentro del proyecto (Ver figura 6.10).

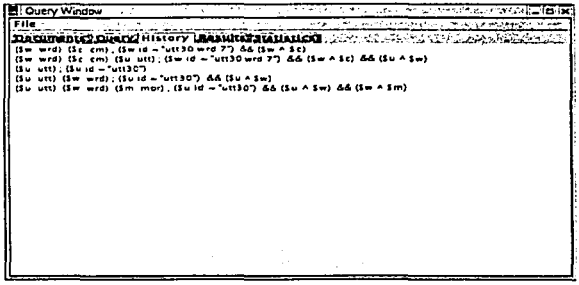


Figura 6.10: Consultas realizadas.

6.4 Consultas del Corpus DIME

En la base de datos se pueden realizar principalmente dos tipos de consultas, simples y complejas. Las primeras consisten en buscar la información dentro de un mismo nivel de transcripción. Las segundas consisten en hacer una búsqueda en dos o más niveles de transcripción. A continuación se presentan algunos ejemplos de estos dos tipos de consultas que se pueden hacer a la base de datos.

6.4.1 Consultas simples

Ejemplo 1

Como primer ejemplo, se buscará la palabra *muebles* dentro del corpus. Para hacer esto, se genera la expresión de consulta declarando una variable $w del "elemento palabra" (*wrd*), de la siguiente forma:

($w wrd)

La expresión de consulta es:

($w type ~ "muebles")

Esta expresión indica que se busque la palabra "muebles". La expresión de consulta completa queda de la siguiente forma:

($w wrd);($w type ~ "muebles")

Los resultados de esta consulta se muestran en la figura 6.11. Se observa que la palabra muebles aparece entre las unidades de tiempo 94.216406 y 94.576406.

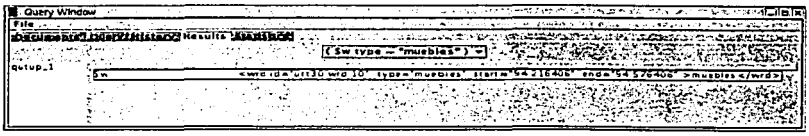


Figura 6.11: Búsqueda de la palabra *muebles*, dentro del corpus.

Ejemplo 2

En el siguiente ejemplo se busca la octava sílaba de la oración. Para esta búsqueda se utilizará el atributo *id* cuyo valor sea la información a la que corresponde *utt30.sil.8*. Esta búsqueda se realiza en el nivel de la transcripción silábica.

Se declara la siguiente variable, del tipo sílaba ("elemento sil"):

($s sil)

y se completa la expresión de consulta de la siguiente forma:

($s id ~ "utt30.sil.8")

La expresión de consulta completa es la siguiente:

($s sil) ; ($s id ~ "utt30.sil.8")

Como resultado de esta búsqueda se tiene que (*"id=utt30.sil.8"*) la octava sílaba, corresponde a la sílaba *trar*. En la figura 6.12 se puede ver el resultado.

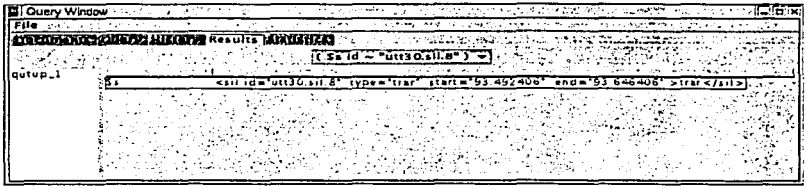


Figura 6.12: Búsqueda en el nivel de la transcripción silábica.

Ejemplo 3

Esta vez la busqueda se hace a nivel de expresiones. Se buscará la oración 30 del diálogo (*id=utt30*).

La variable es del tipo expresión (utterance) y se declara de la siguiente forma:

($u utt)

la expresión de consulta es la siguiente:

($u id ~ "utt30")

La expresión de consulta completa (($u utt) && ($u id ~ "utt30")), obtuvo como resultado la siguiente información de la expresión 30 (fig. 6.13):

no-vocal eh me puedes mostrar los tipos de muebles que tengo

Ejemplo 4

A continuación se va a buscar cada alófono *e* que exista en la transcripción fonética. Primero, se declara la variable de la siguiente forma:

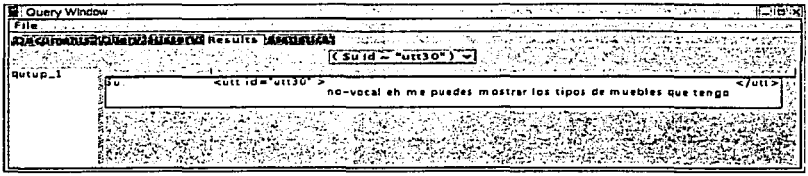


Figura 6.13: Búsqueda de la expresión 30 del corpus DIME.

($p phn)

y se completa con la siguiente expresión de consulta:

($p type ~ "e")

La expresión de consulta completa es:

($p phn) ; ($p type ~ "e")

En esta búsqueda se encontró el alófono *e* en 6 ocasiones. La figura 6.14 muestra los resultados de esta búsqueda.

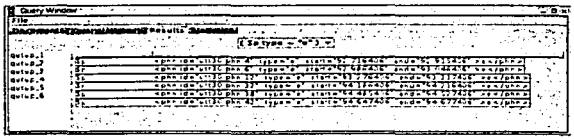


Figura 6.14: Resultados de la búsqueda del alófono *e*.

En la siguiente sección se presentan algunos ejemplos de las consultas que se pueden hacer entre dos o más niveles de transcripción.

6.4.2 Consultas complejas

Ejemplo 1

Este ejemplo consiste en buscar qué sílabas conforman la octava palabra de una oración (aquella palabra que tiene como identificador a *utt30.wrd.8*).

La expresión de consulta es la siguiente:

($w wrd)($s sil) ; ($w id ~ "utt30.wrd.8") && ($w ^ $s)

En la primera parte de la expresión se encuentra la declaración de variables. En este caso se declararon dos variables una del tipo *wrd* y la otra del tipo *sil*. La primera corresponde a la transcripción ortográfica (nivel de palabras) y la segunda a la transcripción silábica (nivel de sílabas).

En la expresión ($w id ~ "utt30.wrd.8") se indica que el valor de la variable corresponde a la octava palabra (que su *id* tiene el valor de "utt30.wrd.8").

($w ^ $s) $w está en un nivel superior que $s , es decir que $w contiene a $s.

Los resultados de esta consulta se pueden observar en la figura 6.15.

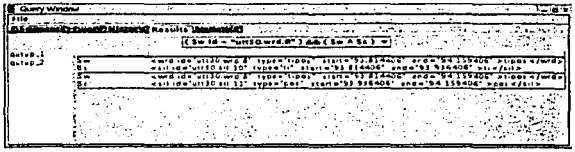


Figura 6.15: Consulta en dos niveles de transcripción: ortográfico y silábico.

De la consulta anterior se obtuvo que la octava palabra es *tipos*. Esta palabra esta compuesta por dos sílabas "*ti*" y "*pos*" (figura 6.15).

Ejemplo 2

Retomando el ejemplo anterior, ahora se quiere buscar además, los fonemas de la octava palabra.

La expresión de consulta se puede ver en la figura 6.16.

En la parte superior de la ventana se puede ver la declaración de tres variables:

($w wrd)
($s sil)
($f fon)

Las dos primeras variables corresponden al ejemplo anterior. La tercera variable (*$f*) que se declara es del tipo *fon*, es decir, pertenece a la transcripción fonológica (fonemas).

En la parte inferior de la ventana, se encuentran las operaciones que se van a hacer con esas variables. A este ejemplo se le agregó una nueva operación (*$s ^ $f*). Con esta operación se indica que $s está en un nivel superior que $f y que $s contiene a $f.

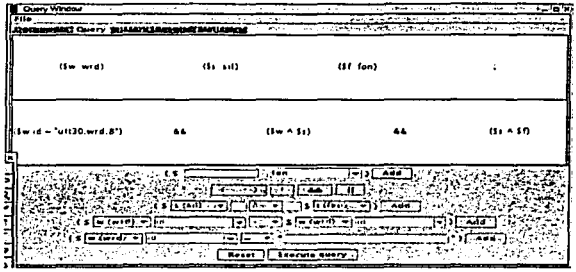


Figura 6.16: Consulta en tres niveles de transcripción.

Como resultado de esta consulta se obtuvo que la palabra *tipos*, está compuesta por las sílabas "*ti*" y "*pos*", y por los fonemas /*t*, *i* , *p*, *o*, *s*/. (Ver figura 6.17)

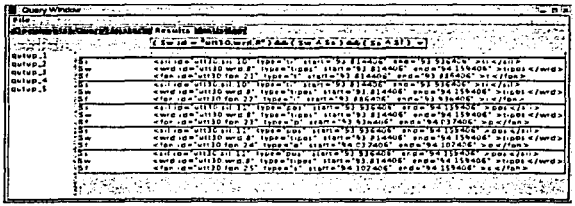


Figura 6.17: Resultados de la consulta.

Ejemplo 3

Para este ejemplo solamente se van a declarar dos variables:

($w wrd) y ($f fon)

el resto de la expresión de consulta se presenta a continuación:

($w id ~ "utt30.wrd.8") && ($w ^ $f)

Este ejemplo es parecido a los dos anteriores. Aquí, también se quiere buscar la octava palabra Sin embargo, en esta ocasión no se quiere obtener las sílabas de esta palabra, lo único que se quiere son los fonemas.

A través de la operación ($w ^ $f), se especifica que el elemento *wrd* está a un nivel superior que el elemento *fon*. En este caso $w está a una distancia de dos niveles superiores que $f, según el modelo presentado en la figura 5.1 del capítulo 5.

En la figura 6.18 se muestran los resultados de esta búsqueda, donde se obtuvo la palabra "*tipos*" con sus respectivos fonemas.

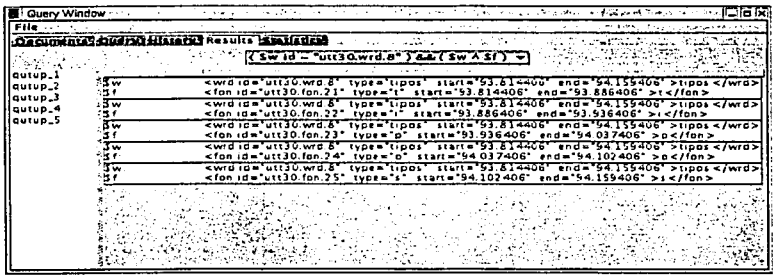


Figura 6.18: La palabra "*tipos*" y sus fonemas.

Ejemplo 4

El siguiente ejemplo consiste en obtener las palabras de la expresión 30. A continuación se presenta la expresión de consulta:

($u utt) ($w wrd) ; ($u id ~ "utt30") && ($u ^ $w)

En esta expresión se declaran dos variables, una que corresponde a la expresión *($u utt)* y la otra a la palabra *($w wrd)*. En la segunda parte de la expresión se indica que la expresión es la 30 (su atributo *id* tiene el valor de "utt30") y que la expresión está formada por las palabras $w ($u ^ $w). El resultado de esta busqueda se muestra en la figura 6.19

6.5 Conclusiones

En este capítulo se describió el proceso para realizar una consulta en la herramienta de MATE. También se presentaron algunos ejemplos de los tipos de consultas que se pueden hacer. Principalmente, las consultas son de dos tipos, simples y complejas. Las primeras consisten en buscar la información dentro de un mismo nivel y las segundas, en dos o más niveles del modelo jerárquico.

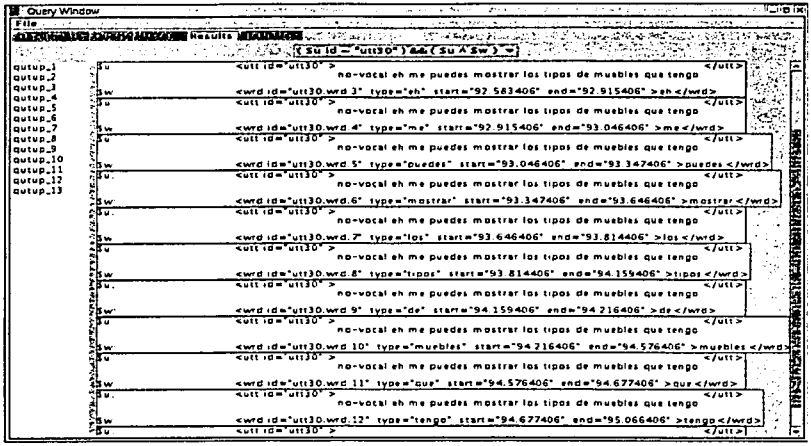


Figura 6.19: Búsqueda de las palabras de la expresión 30.

En el siguiente capítulo se presentan las conclusiones a las que se llegaron, a partir de este trabajo de investigación.

Capítulo 7

Conclusiones

En este trabajo de tesis se propuso una metodología para incorporar diferentes fuentes de conocimiento lingüístico dentro de un modelo jerárquico. La finalidad de este modelo es presentar un esquema de organización para el corpus DIME. La recopilación de este corpus fue necesaria para modelar una parte del sistema conversacional que se pretende desarrollar dentro del proyecto DIME. La organización del corpus permitió explotar más y mejor la infomación que contiene mediante la ayuda de una herramienta computacional.

En esta tesis se presentó la importancia que tiene la creación de los corpus de voz, haciendo énfasis en aquellos desarrollados para el español hablado en México.

El modelo jerárquico propuesto organiza a través de niveles y de relaciones jerárquicas los diferentes tipos de información lingüística del corpus DIME (información ortográfica, fonética, prosódica, gramatical, etc.), y las relaciones establecidas entre estos tipos. Toda la información lingüística estuvo representada en un conjunto de transcripciones.

Para validar computacionalmente el modelo jerárquico, éste se implementó utilizando la herramienta de MATE. MATE está integrada por una arquitectura, la cual comprende cinco componentes principales: 1) una base de datos interna, 2) un lenguaje y un procesador de consultas, 3) un lenguaje y un procesador de hojas de estilo, 4) un procesador para la visualización y 5) una interfaz de usuario.

Para integrar el corpus en la herramienta de MATE, se utilizó el lenguaje XML como el formato de transcripción del corpus. Se describió también, como el lenguaje XML (eXtensible Markup Language) comparte muchos conceptos con las bases de datos semiestructuradas, además de que éste lenguaje se ha estado usando como un estándar en la transcripción de corpus.

También, se presentó el proceso de implementación de cinco modelos lineales creados a partir del modelo jerárquico propuesto, ya que en la herramienta de

MATE solamente se establecen relaciones uno a uno entro dos niveles del modelo.

Dentro del modelo jerárquico, el nivel de palabras tiene una relación de uno a muchos. Las palabras se relacionan con las sílabas, con los morfemas, las categorías gramaticales y con los verbos; estas relaciones se separaron en diferentes modelos para asociar a las palabras con distintas subunidades (palabras con sílabas, palabras con morfemas, etc.).

Para la implementación se crearon cinco proyectos, uno por cada modelo lineal que se obtuvo. Los proyectos "Transcripciones", "Morfemas", "Cat-Gram", "Intencion-Verbal" y "Clíticos", se crearon para poder organizar las transcripciones del corpus DIME dentro de la base de datos de MATE.

Para poder validar computacionalmente el modelo jerárquico, se escogió el diálogo 6 del corpus DIME. Todos los niveles de la información lingüística (fonética, morfológica, silábica, fonológica, categorías gramaticales, ortográfica, a nivel de expresiones, frases clíticas, frases verbales y verbos) fueron implementados con las transcripciones de la expresión 30 del diálogo 6; de esta forma se comprobó que el modelo funciona adecuadamente dentro de la herramienta de MATE.

Es posible extender este modelo jerárquico agregando otros tipos de información (prosódica, referencial, actos del habla). Para hacer esto, es necesario incorporar cada relación que va a existir entre la información existente y la nueva información.

Por último, se puede decir que la herramienta de consultas de MATE trabaja satisfactoriamente al hacer búsquedas de la información lingüística. El único inconveniente de MATE es que está implementado en Java y esto hace que el poceso de consultas sea lento.

La información lingüística obtenida en los procesos de consultas de MATE permitirá crear modelos de fenómenos lingüísticos útiles para desarrollar sistemas conversacionales y sistemas de reconocimiento de voz.

Apéndice A

Instalación de MATE

MATE esta disponible en internet [1] para que cualquier persona pueda usarla. Una de la ventajas de MATE es que, está escrita en Java, puede usarse sobre cualquier plataforma (siempre y cuando se tenga la versión de Java 1.2.X). La herramienta de MATE se descargó en una máquina que tiene de sistema operativo a linux.

Los archivos obtenidos desde internet fueron: MateWorkbench.jar, matedata.tar.gz y matesrc.tar.gz, todos fueron guardados dentro un mismo directorio al que se le denominó MATE. Tanto el archivo matedata.tar.gz como matesrc.tar.gz, tuvieron que ser descomprimidos para poder usarlos.

El archivo matedata.tar.gz creó dos directorios: Projects y MateSetup; el primero contiene varios ejemplos de proyectos que cuentan con diversas transcripciones codificadas en XML; el segundo contiene todos los archivos necesarios para el buen funcionamiento de MATE.

Para ejecutar MATE se creó un archivo ejecutable "runMATE" con las siguientes líneas:

- **export CLASSPATH=/home/alumnos/roxana/MATE/ MateWorkbench.jar**
- **java mate.Workbech**

La ejecución de MATE en linux se realizó con el comando ./runMATE.

[1] http://www.cogsci.ed.ac.uk/ dmck/MateCode/

Apéndice B

Codificación de las Transcripciones

Antes de incluir los datos del corpus en la base de datos, se codificaron las transcripciones del corpus DIME en el formato de XML. Para codificar las transcripciones se usaron dos programas, uno hecho en perl y otro en Java (este último viene con la herramienta de MATE).

Se desarrolló el programa en perl llamado cslu2esps para transformar las transcripciones que estaban en el formato de la herramienta *Speech Viewer* y se convirtieron al formato de *Entropic Xwaves Xlabel*, para que el programa de MATE las pudiera entender.

El programa CallXlabel2Xml convierte archivos del formato de transcripción del programa "Entropic Xwaves Xlabel" al formato de XML (versión 1.0). El programa tiene como entrada un archivo de transcripción (xlabel) y da como salida un archivo en XML y un arvhivo DTD que valida al XML.

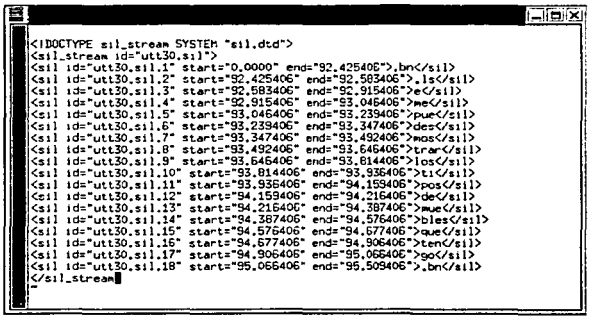
Este programa cuenta con dos opciones que se pueden utilizar[1]:

- **-element:** Se usa para especificar el nombre de las etiquetas del archivo que se va a transformar. Por default este programa pone etiquetas del tipo "word".
- **-nodtd:** Con está opción no se genera ningún archivo DTD de salida.

Para ilustrar mejor el funcionamiento del programa CallXlabel2Xml a continuación se presenta un ejemplo:

Se tiene el archivo "*utt30.sil*":

[1] http://www.cogsci.ed.ac.uk/ dmck/MateCode/conversion/xlabel_user_doc.html



```
<!DOCTYPE sil_stream SYSTEM "sil.dtd">
<sil_stream id="utt30.sil">
<sil id="utt30.sil.1" start="0.0000" end="92.425406">.bn</sil>
<sil id="utt30.sil.2" start="92.425406" end="92.583406">.ls</sil>
<sil id="utt30.sil.3" start="92.583406" end="92.915406">e</sil>
<sil id="utt30.sil.4" start="92.915406" end="93.046406">me</sil>
<sil id="utt30.sil.5" start="93.046406" end="93.239406">pue</sil>
<sil id="utt30.sil.6" start="93.239406" end="93.347406">des</sil>
<sil id="utt30.sil.7" start="93.347406" end="93.492406">mos</sil>
<sil id="utt30.sil.8" start="93.492406" end="93.646406">trar</sil>
<sil id="utt30.sil.9" start="93.646406" end="93.814406">los</sil>
<sil id="utt30.sil.10" start="93.814406" end="93.936406">ti</sil>
<sil id="utt30.sil.11" start="93.936406" end="94.159406">pos</sil>
<sil id="utt30.sil.12" start="94.159406" end="94.216406">de</sil>
<sil id="utt30.sil.13" start="94.216406" end="94.387406">mue</sil>
<sil id="utt30.sil.14" start="94.387406" end="94.576406">bles</sil>
<sil id="utt30.sil.15" start="94.576406" end="94.677406">que</sil>
<sil id="utt30.sil.16" start="94.677406" end="94.906406">ten</sil>
<sil id="utt30.sil.17" start="94.906406" end="95.066406">go</sil>
<sil id="utt30.sil.18" start="95.066406" end="95.509406">.bn</sil>
</sil_stream>
```

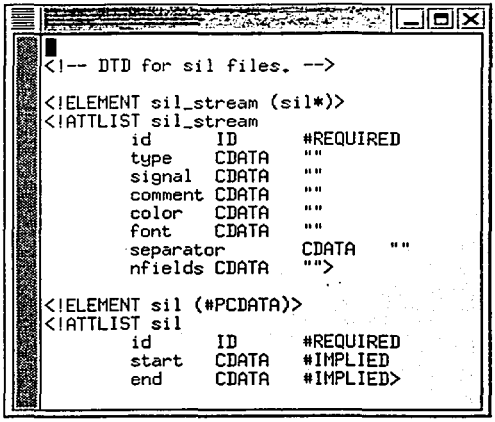
Figura B.1: Archivo "utt30.xml".

```
#
92.425406 115 .bn
92.583406 115 .ls
92.915406 115 e
93.046406 115 me
93.239406 115 pue
93.347406 115 des
93.492406 115 mos
93.646406 115 trar
93.814406 115 los
93.936406 115 ti
94.159406 115 pos
94.216406 115 de
94.387406 115 mue
94.576406 115 bles
94.677406 115 que
94.906406 115 ten
95.066406 115 go
95.509406 115 .bn
```

y se ejecuta el programa de conversión de MATE con la siguiente línea:

java CallXlabel2Xml -element "sil" utt30.sil

como resultado se van a obtener dos archivos, uno llamado *utt30.xml* y el otro llamado *sil.dtd*, éste último corresponde a un archivo DTD.



```
<!-- DTD for sil files. -->

<!ELEMENT sil_stream (sil*)>
<!ATTLIST sil_stream
        id        ID        #REQUIRED
        type      CDATA     ""
        signal    CDATA     ""
        comment   CDATA     ""
        color     CDATA     ""
        font      CDATA     ""
        separator           CDATA     ""
        nfields   CDATA     "">

<!ELEMENT sil (#PCDATA)>
<!ATTLIST sil
        id        ID        #REQUIRED
        start     CDATA     #IMPLIED
        end       CDATA     #IMPLIED>
```

Figura B.2: Archivo "sil.dtd".

Cuando se ejecutó el programa de MATE se obtuvieron solamente los archivos XML de las transcripciones del corpus DIME. Se creó un archivo XML por cada nivel de cada uno de los modelos lineales. Con el programa CallXlabel2Xml no se generaron los archivos DTD, éstos archivos se generaron manualmente ya que se debían definir otros elementos, como los atributos *href*, que el DTD generado por default no contemplaba.

Apéndice C

Archivos XML de las transcripciones

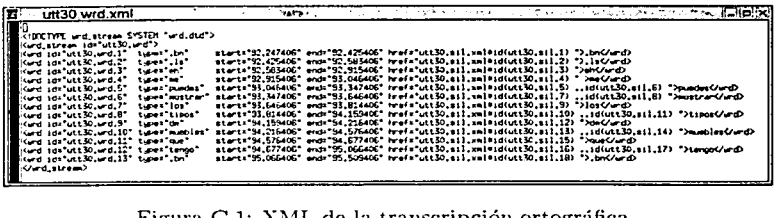


Figura C.1: XML de la transcripción ortográfica.

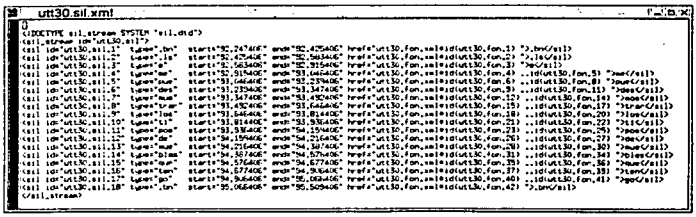


Figura C.2: XML de la transcripción silábica.

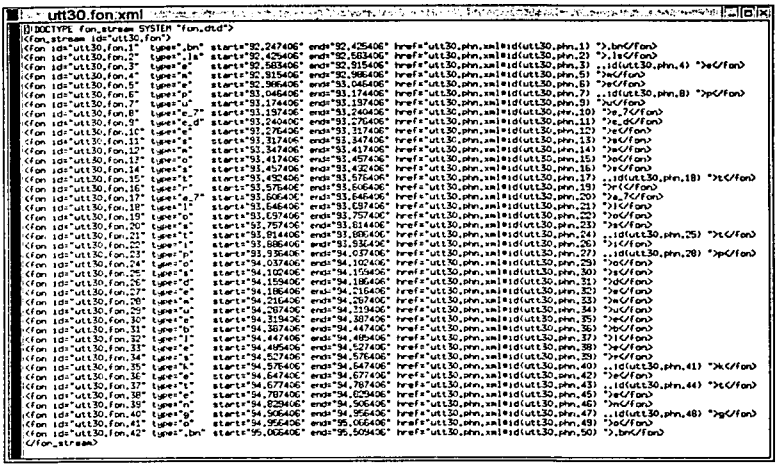


Figura C.3: XML de la transcripción fonológica.

utt30.phn.xml

```
<!DOCTYPE phn_stream SYSTEM "phn.dtd">
<phn_stream id="utt30.phn">
<phn id="utt30.phn.1"  type=".bn" start="92.247406" end="92.425406">.bn</phn>
<phn id="utt30.phn.2"  type=".ls" start="92.425406" end="92.583406">.ls</phn>
<phn id="utt30.phn.3"  type="e_?" start="92.583406" end="92.716406">e_?</phn>
<phn id="utt30.phn.4"  type="e"   start="92.716406" end="92.915406">e</phn>
<phn id="utt30.phn.5"  type="m"   start="92.915406" end="92.986406">m</phn>
<phn id="utt30.phn.6"  type="e"   start="92.986406" end="93.046406">e</phn>
<phn id="utt30.phn.7"  type="p_c" start="93.046406" end="93.137406">p_c</phn>
<phn id="utt30.phn.8"  type="p"   start="93.137406" end="93.174406">p</phn>
<phn id="utt30.phn.9"  type="w"   start="93.174406" end="93.197406">w</phn>
<phn id="utt30.phn.10" type="e_7" start="93.197406" end="93.240406">e_7</phn>
<phn id="utt30.phn.11" type="e_d" start="93.240406" end="93.276406">e_d</phn>
<phn id="utt30.phn.12" type="e"   start="93.276406" end="93.317406">e</phn>
<phn id="utt30.phn.13" type="z"   start="93.317406" end="93.347406">z</phn>
<phn id="utt30.phn.14" type="m"   start="93.347406" end="93.417406">m</phn>
<phn id="utt30.phn.15" type="O"   start="93.417406" end="93.457406">O</phn>
<phn id="utt30.phn.16" type="s"   start="93.457406" end="93.492406">s</phn>
<phn id="utt30.phn.17" type="t_c" start="93.492406" end="93.538406">t_c</phn>
<phn id="utt30.phn.18" type="t"   start="93.538406" end="93.576406">t</phn>
<phn id="utt30.phn.19" type="r(" start="93.576406" end="93.606406">r(</phn>
<phn id="utt30.phn.20" type="a_7" start="93.606406" end="93.646406">a_7</phn>
<phn id="utt30.phn.21" type="l"   start="93.646406" end="93.697406">l</phn>
<phn id="utt30.phn.22" type="o"   start="93.697406" end="93.757406">o</phn>
<phn id="utt30.phn.23" type="s"   start="93.757406" end="93.814406">s</phn>
<phn id="utt30.phn.24" type="t_c" start="93.814406" end="93.845406">t_c</phn>
<phn id="utt30.phn.25" type="t"   start="93.845406" end="93.886406">t</phn>
<phn id="utt30.phn.26" type="i_7" start="93.886406" end="93.936406">i_7</phn>
<phn id="utt30.phn.27" type="p_c" start="93.936406" end="94.007406">p_c</phn>
<phn id="utt30.phn.28" type="p"   start="94.007406" end="94.037406">p</phn>
<phn id="utt30.phn.29" type="O"   start="94.037406" end="94.102406">O</phn>
<phn id="utt30.phn.30" type="z"   start="94.102406" end="94.159406">z</phn>
<phn id="utt30.phn.31" type="D"   start="94.159406" end="94.186406">D</phn>
<phn id="utt30.phn.32" type="e"   start="94.186406" end="94.216406">e</phn>
<phn id="utt30.phn.33" type="m"   start="94.216406" end="94.287406">m</phn>
<phn id="utt30.phn.34" type="w"   start="94.287406" end="94.319406">w</phn>
<phn id="utt30.phn.35" type="e_7" start="94.319406" end="94.387406">e_7</phn>
<phn id="utt30.phn.36" type="V"   start="94.387406" end="94.447406">V</phn>
<phn id="utt30.phn.37" type="l"   start="94.447406" end="94.485406">l</phn>
<phn id="utt30.phn.38" type="e"   start="94.485406" end="94.527406">e</phn>
<phn id="utt30.phn.39" type="s"   start="94.527406" end="94.576406">s</phn>
<phn id="utt30.phn.40" type="k_c" start="94.576406" end="94.611406">k_c</phn>
<phn id="utt30.phn.41" type="k_j" start="94.611406" end="94.647406">k_j</phn>
<phn id="utt30.phn.42" type="e"   start="94.647406" end="94.677406">e</phn>
<phn id="utt30.phn.43" type="t_c" start="94.677406" end="94.757406">t_c</phn>
<phn id="utt30.phn.44" type="t"   start="94.757406" end="94.787406">t</phn>
<phn id="utt30.phn.45" type="e_7" start="94.787406" end="94.829406">e_7</phn>
<phn id="utt30.phn.46" type="N"   start="94.829406" end="94.906406">N</phn>
<phn id="utt30.phn.47" type="g_c" start="94.906406" end="94.936406">g_c</phn>
<phn id="utt30.phn.48" type="g"   start="94.936406" end="94.956406">g</phn>
<phn id="utt30.phn.49" type="o"   start="94.956406" end="95.066406">o</phn>
<phn id="utt30.phn.50" type=".bn" start="95.066406" end="95.509406">.bn</phn>
</phn_stream>
```

Figura C.4: XML de la transcripción fonética.

utt30.cg.xml

```
<!DOCTYPE cg_stream SYSTEM "cg.dtd">
<cg_stream id="utt30.cg">
<cg id="utt30.cg.1"   type=".bn"          start="92.247406" end="92.425406"  >.bn</cg>
<cg id="utt30.cg.2"   type=".ls"          start="92.425406" end="92.583406" >.ls</cg>
<cg id="utt30.cg.3"   type="interjeccion" start="92.583406" end="92.915406"  >int</cg>
<cg id="utt30.cg.4"   type="pronombre"    start="92.915406" end="93.046406" >pron</cg>
<cg id="utt30.cg.5"   type="verbo"        start="93.046406" end="93.347406" >verbo</cg>
<cg id="utt30.cg.6"   type="verbo"        start="93.347406" end="93.646406" >verbo</cg>
<cg id="utt30.cg.7"   type="articulo"     start="93.646406" end="93.814406" >art</cg>
<cg id="utt30.cg.8"   type="sustantivo"   start="93.814406" end="94.159406" >sust</cg>
<cg id="utt30.cg.9"   type="preposicion"  start="94.159406" end="94.216406" >prep</cg>
<cg id="utt30.cg.10"  type="sustantivo"   start="94.216406" end="94.576406" >sust</cg>
<cg id="utt30.cg.11"  type="pronombre"    start="94.576406" end="94.677406" >pron</cg>
<cg id="utt30.cg.12"  type="verbo"        start="94.677406" end="95.066406" >verbo</cg>
<cg id="utt30.cg.13"  type=".bn"          start="95.066406" end="95.509406" >.bn</cg>
</cg_stream>
```

Figura C.5: XML de la transcripción de las categorías gramaticales.

utt30.mor.xml

```
<!DOCTYPE mor_stream SYSTEM "mor.dtd">
<mor_stream id="utt30.mor">
<mor id="utt30.mor.1"  type="none"   start="92.247406" end="92.425406" href="utt30.cm.xml#id(utt30.cm.1)">none</mor>
<mor id="utt30.mor.2"  type="none"   start="92.425406" end="92.583406" href="utt30.cm.xml#id(utt30.cm.2)">none</mor>
<mor id="utt30.mor.3"  type="none"   start="92.583406" end="92.915406" href="utt30.cm.xml#id(utt30.cm.3)">none</mor>
<mor id="utt30.mor.4"  type="none"   start="92.915406" end="93.046406" href="utt30.cm.xml#id(utt30.cm.4)">none</mor>
<mor id="utt30.mor.5"  type="pued"   start="93.046406" end="93.276406" href="utt30.cm.xml#id(utt30.cm.5)" >pued</mor>
<mor id="utt30.mor.6"  type="es"     start="93.276406" end="93.347406" href="utt30.cm.xml#id(utt30.cm.6)">es</mor>
<mor id="utt30.mor.7"  type="mostr"  start="93.347406" end="93.606406" href="utt30.cm.xml#id(utt30.cm.7)">mostr</mor>
<mor id="utt30.mor.8"  type="ar"     start="93.606406" end="93.646406" href="utt30.cm.xml#id(utt30.cm.8)">ar</mor>
<mor id="utt30.mor.9"  type="none"   start="93.646406" end="93.814406" href="utt30.cm.xml#id(utt30.cm.9)">none</mor>
<mor id="utt30.mor.10" type="tip"    start="93.814406" end="94.037406" href="utt30.cm.xml#id(utt30.cm.10)">tip</mor>
<mor id="utt30.mor.11" type="os"     start="94.037406" end="94.159406" href="utt30.cm.xml#id(utt30.cm.11)">os</mor>
<mor id="utt30.mor.12" type="none"   start="94.159406" end="94.216406" href="utt30.cm.xml#id(utt30.cm.12)">none</mor>
<mor id="utt30.mor.13" type="mueble" start="94.216406" end="94.527406" href="utt30.cm.xml#id(utt30.cm.13)">mueble</mor>
<mor id="utt30.mor.14" type="a"      start="94.527406" end="94.576406" href="utt30.cm.xml#id(utt30.cm.14)">a</mor>
<mor id="utt30.mor.15" type="none"   start="94.576406" end="94.677406" href="utt30.cm.xml#id(utt30.cm.15)">none</mor>
<mor id="utt30.mor.16" type="ten"    start="94.677406" end="94.906406" href="utt30.cm.xml#id(utt30.cm.16)">ten</mor>
<mor id="utt30.mor.17" type="go"     start="94.906406" end="95.066406" href="utt30.cm.xml#id(utt30.cm.17)">go</mor>
<mor id="utt30.mor.18" type="none"   start="95.066406" end="95.509406" href="utt30.cm.xml#id(utt30.cm.18)">none</mor>
</mor_stream>
```

Figura C.6: XML de la transcripción morfológica.

utt30.cm.xml

```
<!DOCTYPE cm_stream SYSTEM "cm.dtd">
<cm_stream id="utt30.cm">
<cm id="utt30.cm.1"  type="none"   start="92.247406" end="92.425406">none</cm>
<cm id="utt30.cm.2"  type="none"   start="92.425406" end="92.583406">none</cm>
<cm id="utt30.cm.3"  type="none"   start="92.583406" end="92.915406">none</cm>
<cm id="utt30.cm.4"  type="none"   start="92.915406" end="93.046406">none</cm>
<cm id="utt30.cm.5"  type="raiz"   start="93.046406" end="93.276406">raiz</cm>
<cm id="utt30.cm.6"  type="sufijo" start="93.276406" end="93.347406">sufijo</cm>
<cm id="utt30.cm.7"  type="raiz"   start="93.347406" end="93.606406">raiz</cm>
<cm id="utt30.cm.8"  type="sufijo" start="93.606406" end="93.646406">sufijo</cm>
<cm id="utt30.cm.9"  type="none"   start="93.646406" end="93.814406">none</cm>
<cm id="utt30.cm.10" type="raiz"   start="93.814406" end="94.037406">raiz</cm>
<cm id="utt30.cm.11" type="sufijo" start="94.037406" end="94.159406">sufijo</cm>
<cm id="utt30.cm.12" type="none"   start="94.159406" end="94.216406">none</cm>
<cm id="utt30.cm.13" type="raiz"   start="94.216406" end="94.527406">raiz</cm>
<cm id="utt30.cm.14" type="sufijo" start="94.527406" end="94.576406">sufijo</cm>
<cm id="utt30.cm.15" type="none"   start="94.576406" end="94.677406">none</cm>
<cm id="utt30.cm.16" type="raiz"   start="94.677406" end="94.906406">raiz</cm>
<cm id="utt30.cm.17" type="sufijo" start="94.906406" end="95.066406">sufijo</cm>
<cm id="utt30.cm.18" type="none"   start="95.066406" end="95.509406">none</cm>
</cm_stream>
```

Figura C.7: XML de la transcripción de categorías de morfemas.

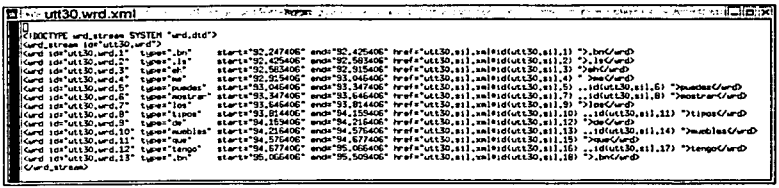


Figura C.8: XML de la Transcripción ortográfica.

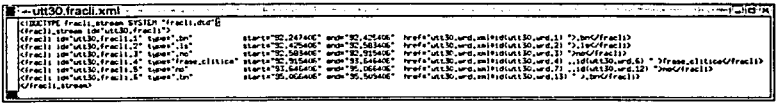


Figura C.9: XML de la transcripción de frases clíticas.

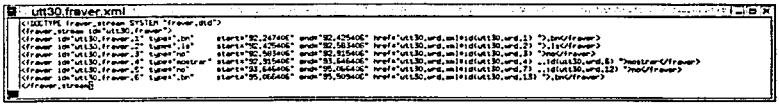


Figura C.10: XML de la transcripción de las frases con intención verbal.

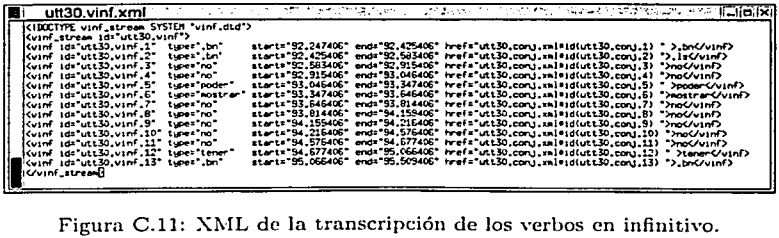


Figura C.11: XML de la transcripción de los verbos en infinitivo.

utt30.conj.xml

```
<!DOCTYPE conj_stream SYSTEM "conj.dtd">
<conj_stream id="utt30.conj">
<conj id="utt30.conj.1"  type=".bn"          start="92.247406" end="92.425406">.bn</conj>
<conj id="utt30.conj.2"  type=".ls"          start="92.425406" end="92.583406">.ls</conj>
<conj id="utt30.conj.3"  type="no"           start="92.583406" end="92.915406">no</conj>
<conj id="utt30.conj.4"  type="no"           start="92.915406" end="93.046406">no</conj>
<conj id="utt30.conj.5"  type="2s-pres-i"    start="93.046406" end="93.347406">2s-pres-i</conj>
<conj id="utt30.conj.6"  type="inf"          start="93.347406" end="93.646406">inf</conj>
<conj id="utt30.conj.7"  type="no"           start="93.646406" end="93.814406">no</conj>
<conj id="utt30.conj.8"  type="no"           start="93.814406" end="94.159406">no</conj>
<conj id="utt30.conj.9"  type="no"           start="94.159406" end="94.216406">no</conj>
<conj id="utt30.conj.10" type="no"           start="94.216406" end="94.576406">no</conj>
<conj id="utt30.conj.11" type="no"           start="94.576406" end="94.677406">no</conj>
<conj id="utt30.conj.12" type="1s-pres-i"    start="94.677406" end="95.066406">1s-pres-i</conj>
<conj id="utt30.conj.13" type=".bn"          start="95.066406" end="95.509406">.bn</conj>
</conj_stream>
```

Figura C.12: XML de la transcripción de la conjugación de verbos.

utts-dial6.xml

```
<turn id="turn.13" who="s">
<utt id="utt21">
okey
</utt>
<utt id="utt22">
ruido
</utt>
<utt id="utt23">
ahi esta bien
</utt></turn>
<turn id="turn.14" who="u">
<utt id="utt24">
ahi esta bien
</utt>
<utt id="utt25">
despues sil me puedes poner sil el el extractor de aire encima de la sil de la estufa
</utt></turn>
<turn id="turn.15" who="s">
<utt id="utt26">
okey
</utt>
<utt id="utt27">
ruido no-vocal asi esta bien
</utt></turn>
<turn id="turn.16" who="u">
<utt id="utt28">
si
</utt>
<utt id="utt29">
asi esta bien
</utt>
<utt id="utt30" href="utt30.wrd.xml#id(utt30.wrd.1)..id(utt30.wrd.13)">
no-vocal eh me puedes mostrar los tipos de muebles que tengo
</utt></turn>
<turn id="turn.17" who="s">
<utt id="utt31">
okey
</utt>
<utt id="utt32">
hay un catalogo de alacenas uno de estantes sil uno de estufas y extractores uno de frega
deros y maquinas lavatrastes sil uno de gabinetes uno de mesas sil uno de refrigeradores
sil y uno de sillas bancos y sillones
</utt></turn>
<turn id="turn.18" who="u">
<utt id="utt33">
no-vocal me puedes mostrar el catalogo de fregaderos y maquinas
</utt></turn>
<turn id="turn.19" who="s">
<utt id="utt34">
okey
</utt>
<utt id="utt35">
no-vocal este es el catalogo de fregaderos y maquinas lavatrastes
```

Figura C.13: Extracto del XML de la transcripción de las expresiones del diálogo 6.

Bibliografía

[Abiteboul, 1997] Abiteboul, S., (1997), *Querying Semi-Structured Data*, INRIA-Rocquencourt.

[Allen et al., 2000] Allen J., Choukri K., (2000), *Survey of Language Engineering needs: a Language Resources perspective.* Conferencia LREC 2000.

[Barbosa et al., 1998] Barbosa A., Cole R., Munive N., Serridge B., y Vargas A., (1998), *Creating a Mexican Spanish Version of the CSLU Toolkit.* ICSLP, Sydney Australia.

[Beristain, 2000] Beristáin H.,(2000), *Diccionario de Retórica y Poética.* Octava edición. Editorial Porrua. México.

[Bielawski et al., 1991] Bielawski L., Lewand R.,(1991), *Intelligent Systems Design.* Editorial Wiley Estados Unidos.

[Blecua et al.,1999] Blecua, J.M., Clavería, G., Sánchez, C., Torruella, J., (1999), *Filología e informática (Nuevas tecnologías en los estudios filológicos)*, Seminario de Filología e Informática, Departamento de Filología Española, Universidad Autónoma de Barcelona, España.

[Cieri et al., 2000] Cieri C., Liberman M., (2000), *Issues in Corpus Creation and Distribution: The Evolution of the Linguistic Data Consortium.* Conferencia LREC 2000.

[Dybkjaer et al., 2001] Dybkjaer, L., Berman, S., Kipp, M., Olsen, M. W., Pirrelli, V., Reithinger, N. y Soria C., (2001). *Survey of Existing Tools, Standards and User Needs for Annotation of Natural Interaction and Multimodal Data.* ISLE Deliverable D11.1. URL: http://isle.nis.sdu.dk

[Gibbon et al., 1997] Gibbon, D., Moore, R., Winsky, R., (1997), *HandBook of Standars and Resources for Spoken Language Systems*, alter de Gruyter Publishers.

[Krishnamurthy et al.,2000] Krishnamurthy R. y Orasan C., (2000), *An Open Architecture for the Construction and Administration of Corpora.*

[Mckelvie et al., 2000] Mckelvie D., Isard A., Mengel A., Baun M., Grosse M. and Klein M., (2000), *The MATE Workbench – an annotation tool for XML coded speech corpora.*

[Mei-Ling, 1995] Mei-Ling H., (1995), *Phonological Parsing for Bidirectional Letter-to-Sound/Sound-to-Letter Generation*, Tesis de Doctorado. Massachusetts Institute of Technology, Estados Unidos.

[Moreno et al., 2000] Moreno A., Comeyne R., Haslam K., Henk van den Heuvel., Höge H., Horbach S., y Micca G. 2000. *SALA: Speechdat Acroos Latin America. Results of the First Phase.* Conferencia LREC 2000.

[Mungía et al., 1998] Mungía I., Mungía M.E., Rocha G., (1998), *Gramática de la Lengua Española (Reglas y Ejercicios)*, Larousse, México.

[Pineda et al., 2001] Pineda L., Massé A., Meza I., Salas M., Schwarz E., Uraga E., Villaseñor L. *El proyecto DIME*, (2001), IIMAS-UNAM. En procedings of Second International Workshop on Spanish Language Processing and Language Technologies. SLPT2, España.

[Talley, 2000] Talley J., (2000), *The Establishment of Motorola´s Human Language Data Resource Center: Addressing the criticaly of language resources in the industrial setting.* Conferencia LREC 2000.

[Villaseñor et al., 2001] Villaseñor L., Massé A., Pineda L. *The DIME Corpus*, (2001), IIMAS-UNAM. En memorias del 3er. Encuentro Internacional de Ciencias de la Computación ENC01, México.