

Acceso a Servicios de Localización mediante una Interfaz de Voz

Objetivo: *Estudio, diseño e implementación de los componentes imprescindibles para permitir a un usuario acceder a servicios de localización mediante una comunicación bidireccional entre ambos en la cual el usuario sólo se expresa mediante descripciones habladas.*

Documento: **Informe Final**
Versión: 17/08/2002
Autor: Jorge Corral Areán (CI: 1987800-6)
Contexto: **Taller V**, INCO-FING-UDELAR
Tutores: Omar Viera (Depto. IO, FING),
Pablo Piperno (ICA)

0. Contenido

1. Introducción	Pág. 5
1.1 Estructura del Documento	Pág. 5
1.2 Contexto	Pág. 5
2. Definición del Problema	Pág. 7
2.1 Introducción	Pág. 7
2.2 Objetivo	Pág. 7
2.3 Resultados Esperados	Pág. 7
2.4 Módulos que componen el Sistema	Pág. 8
2.5 Detalle de los Módulos	Pág. 9
2.6 Plan de Trabajo Año 2001	Pág. 10
2.7 Plan de Trabajo Año 2002	Pág. 10
2.8 Documentación	Pág. 10
3. Requerimientos	Pág.11
3.1 Introducción	Pág. 11
3.2 Panorama General	Pág. 11
3.3 Clientes / Usuarios	Pág. 11
3.4 Metas	Pág. 11
3.5 Casos de Uso de Alto Nivel	Pág. 12
3.5.1 CU de Alto Nivel Consulta de Direcciones y Puntos de Interés	Pág. 12
3.5.2 CU de Alto Nivel Servicios Cercanos	Pág. 12
3.5.3 CU de Alto Nivel Ruteo	Pág. 12
3.5.4 CU de Alto Nivel Páginas Amarillas	Pág. 13
3.5.5 CU de Alto Nivel Publicidad Móvil	Pág. 13
3.6 Requerimientos Alternativos	Pág. 13
4. Propuesta de Arquitectura	Pág.15
5. Análisis & Diseño	Pág.17
5.1 Consulta de Direcciones y Puntos de Interés	Pág. 19
5.1.1 CU Expandido para Consulta de Direcciones y Puntos de Interés	Pág. 19
5.1.2 Diagramas de Secuencia para cada Curso de Eventos del CU	Pág. 19
5.1.3 Diagramas de Colaboración para cada evento del Diagrama de Secuencia	Pág. 20
5.1.4 Diagrama de Clases de Diseño	Pág. 23
5.2 Servicios Cercanos	Pág. 25
5.2.1 CU Expandido para Servicios Cercanos	Pág. 25
5.2.2 Diagramas de Secuencia para cada Curso de Eventos del CU	Pág. 25
5.2.3 Diagramas de Colaboración para cada evento del Diagrama de Secuencia	Pág. 26
5.2.4 Diagrama de Clases de Diseño	Pág. 28
5.3 Ruteo	Pág. 29
5.3.1 CU Expandido para Ruteo	Pág. 29
5.3.2 Diagramas de Secuencia para cada Curso de Eventos del CU	Pág. 30
5.3.3 Diagramas de Colaboración para cada evento del Diagrama de Secuencia	Pág. 30
5.3.4 Diagrama de Clases de Diseño	Pág. 32
5.4 Páginas Amarillas	Pág. 33
5.4.1 CU Expandido para Páginas Amarillas	Pág. 33
5.4.2 Diagramas de Secuencia para cada Curso de Eventos del CU	Pág. 33
5.4.3 Diagramas de Colaboración para cada evento del Diagrama de Secuencia	Pág. 34
5.4.4 Diagrama de Clases de Diseño	Pág. 35
5.5 Publicidad Móvil	Pág. 36
5.5.1 CU Expandido para Publicidad Móvil	Pág. 36
5.5.2 Diagramas de Secuencia para cada Curso de Eventos del CU	Pág. 36
5.5.3 Diagramas de Colaboración para cada evento del Diagrama de Secuencia	Pág. 37
5.5.4 Diagrama de Clases de Diseño	Pág. 38
5.6 Diagrama general de Clases de Diseño	Pág. 39

6. Implementación del Prototipo	Pág.40
6.1 Propuesta de Prototipo	Pág. 40
6.1.1 Acerca del Prototipo	Pág. 40
6.1.2 Motivación por un Prototipo Único Incremental	Pág. 40
6.2 Definición de las Versiones del Prototipo	Pág. 40
6.2.1 Versión 1	Pág. 41
6.2.2 Versión 2	Pág. 41
6.2.3 Versión 3	Pág. 41
6.3 Acerca de la Implementación del Prototipo	Pág. 42
6.3.1 Acerca del Prototipo en General	Pág. 42
Porqué Visual Basic 6.0 + MapObjects 2.1	Pág. 42
6.3.2 Versión 1 del Prototipo	Pág. 42
Explicación de Esta Versión y sus Funcionalidades	Pág. 42
Los Datos de los Shapefiles <i>Ejes y Direcciones</i>	Pág. 43
Utilización de un Documento de Texto para Almacenar las Calles	Pág. 43
6.3.3 Versión 2 del Prototipo	Pág. 44
Explicación de esta Versión y sus Funcionalidades	Pág. 44
Porqué se Necesita un Método como Soundex	Pág. 44
Ubicaciones Solamente dentro de Montevideo	Pág. 45
Problema de Reconocimiento de Calles	Pág. 45
Consideraciones Previas Acerca del Método Soundex	Pág. 47
Cómo Funciona el Método Soundex	Pág. 47
Mejoras al Método Soundex	Pág. 49
Problemas del Método Soundex	Pág. 50
Los Datos del Shapefile <i>Comercios</i>	Pág. 51
Problemática Acerca de los Rubros	Pág. 52
Proposición de un Diseño para los Datos del Shapefile <i>Comercios</i>	Pág. 53
Impacto Sufrido al Abordar la Presente Versión	Pág. 54
6.3.4 Versión 3 del Prototipo	Pág. 54
Explicación de Esta Versión y sus Funcionalidades	Pág. 54
Motivos para Agregar el Reconocimiento de Voz	Pág. 54
Problemática del Reconocimiento de Voz	Pág. 55
Propuesta para un Nuevo Software de Reconocimiento de Voz	Pág. 56
Porqué se Eligió el IBM ViaVoice v7	Pág. 56
Errores Introducidos por el Reconocimiento de Voz en “Forma Directa”	Pág. 58
Impacto Sufrido al Abordar la Presente Versión	Pág. 58
7. Testeo del Prototipo	Pág.61
7.1 Introducción	Pág. 61
7.2 Testeo de la Versión 1	Pág. 62
7.2.1 Cargar Capa	Pág. 62
7.2.2 Zoom In / Zoom Out	Pág. 62
7.2.3 Pan	Pág. 63
7.2.4 Full Extent	Pág. 63
7.2.5 Etiquetar Capa / Quitar Etiquetas	Pág. 63
7.2.6 Búsqueda por Intersección	Pág. 64
7.2.7 Búsqueda por Número	Pág. 65
7.3 Testeo de la Versión 2	Pág. 66
7.3.1 Función de Localización	Pág. 66
7.3.2 Función de Cercanías	Pág. 68
7.4 Testeo de la Versión 3	Pág. 69
7.4.1 Función de Localización	Pág. 69
7.4.2 Función de Cercanías	Pág. 71
8. Conclusiones	Pág.72
8.1 Objetivos Cumplidos	Pág. 72
8.2 Factibilidad del Sistema	Pág. 72
8.3 Planteo de Escenarios	Pág. 76
8.4 Flexibilidad de la Solución Propuesta	Pág. 77
8.5 Potencial de UML	Pág. 78
8.6 Conclusiones del Prototipo	Pág. 78

9. Trabajos Futuros **Pág.79****10. Referencias** **Pág.81**

Anexos

Anexo A: Tecnologías de Posicionamiento	Pág.84
Introducción	Pág. 84
Basadas en Terminales	Pág. 84
GPS	Pág. 84
E-OTD	Pág. 84
Basadas en Red	Pág. 84
COO	Pág. 84
TOA	Pág. 84
AOA	Pág. 84
Anexo B: GPS	Pág.85
Introducción	Pág. 85
¿Qué es el Sistema GPS?	Pág. 85
¿Cómo Funciona?	Pág. 85
Técnicas y Precisiones	Pág. 85
Posicionamiento Autónomo	Pág. 85
Corrección Diferencial	Pág. 85
Receptores Geodésicos	Pág. 85
Diferencias con los Métodos Tradicionales	Pág. 85
Diferentes Técnicas Utilizando la Tecnología GPS	Pág. 86
Anexo C: Estado del Arte en Reconocimiento del Habla	Pág.87
Introducción	Pág. 87
Últimos Avances	Pág. 87
Reconocimiento de Voz Natural	Pág. 87
Conversión Texto-Voz	Pág. 87
Verificación del Locutor	Pág. 88
Cómo Funciona un Reconocedor de Voz	Pág. 88
Diferentes Tipos de Reconocedores de Voz	Pág. 90
Según el Número de Locutores que Reconozcan	Pág. 90
Según el Tamaño del Vocabulario que Reconozcan	Pág. 90
Según el Canal	Pág. 90
Según el Tiempo de Respuesta	Pág. 90
Características de un Reconocedor de Habla Natural	Pág. 90
Independencia del Locutor	Pág. 90
Efectos del Canal y del Ruido	Pág. 91
Independencia del Dominio Semántico	Pág. 92
Características del Habla Espontánea	Pág. 93
Servicios de Voz	Pág. 93
Portales de Voz	Pág. 93
Directorios Vocales	Pág. 94
Servicios de Información	Pág. 94
Comercio Electrónico	Pág. 94
Asistentes Personales	Pág. 94

1. Introducción

1.1 Estructura del Documento

Se presentará primero una Definición del Problema que se estudia en el Capítulo 2. Luego, se mostrará los Requerimientos planteados para el presente proyecto en el Capítulo 3. Una vez sabido el problema, y los requerimientos ante los cuales nos enfrentamos, en el Capítulo 4 se plantea una posible Arquitectura para una solución general al problema planteado, dado los requerimientos ya analizados.

La arquitectura presentada se dice que es propuesta, ya que se propone utilizar tal arquitectura en el caso eventual de implementar el sistema en general. La idea de la arquitectura es que comunique a grandes rasgos los componentes del sistema, y sus interacciones a alto nivel. Además, sirve como punto de partida para el Análisis & Diseño de cada Caso de Uso.

Las importantes etapas de Análisis y Diseño de tal sistema aparecen juntas en el Capítulo 5, descritas en lenguaje UML, de todo lo que podría hacer un sistema de gran porte utilizando estas tecnologías. Esto viene de la letra del proyecto: “**Análisis y Diseño de un software**...e implementación de un prototipo...”. Es decir, que se cuenta con una sola sección dedicada al Análisis y Diseño, donde se trabaja por Caso de Uso. O sea, que se estudia el Análisis y el Diseño de un CU por vez, hasta cubrirlos todos.

El uso de UML se ve claramente en el análisis y diseño del sistema en general (Capítulo 5), mientras que el prototipo no sigue la metodología mostrada aquí, sino que solo muestra que se puede implementar tal sistema, mas allá de la metodología propuesta, aunque claro está que la presentada aquí creemos que es la mas conveniente.

Luego, en el Capítulo 6 se presenta una discusión de lo que ocurrió al implementar el Prototipo, es decir de lo que efectivamente se llevó a cabo. Esto viene de la letra del proyecto: “Análisis y Diseño de un software... **e implementación de un prototipo**...”. Este prototipo, *grosso modo* implementa dos Casos de Uso descritos para el sistema en general, los cuales se simplifican para ser implementados por el prototipo. Como se dirá luego, el prototipo no pretende seguir la forma de trabajo propuesta en este proyecto (Casos de Uso, Diagramas, Análisis & Diseño, etc.) sino que solo pretende mostrar la viabilidad de implementar un sistema de este porte. O sea que el prototipo no sigue los lineamientos del UML ni los lineamientos o metodología propuesta, por motivos de que fue hecho en paralelo junto con la presente documentación, porque el UML y su metodología ya fue mostrada para el sistema de gran porte, y porque el propósito del prototipo era mostrar la viabilidad de implementar tal sistema.

El Capítulo 7 presenta un Testeo efectuado sobre el prototipo junto con sus resultados. Las Conclusiones aparecen en el Capítulo 8 seguidas por los Trabajos Futuros planteados en el Capítulo 9.

Finalmente, se presentan los Anexos a los cuales se hará referencia en diferentes partes del documento.

1.2 Contexto

Es un hecho que la telefonía celular llegó para quedarse. Las cifras que muestran la cantidad de usuarios de éstos aumentan año a año, y no se requiere de un estudio para respaldar esta afirmación. Son sabidas las rebajas en los costos de los propios teléfonos, las facilidades de pago para adquirirlos, el mantenimiento de la infraestructura de la telefonía celular analógica, los nuevos planes disponibles para usuarios que apuntan a adquirir no sólo nuevos teléfonos sino paquetes de minutos de aire mensuales e incluso la duplicación de los minutos de aire con el mismo costo para algunos servicios.

Sin duda que el viejo modelo de negocios de las empresas operadoras de telefonía está cambiando, y mucho. Lejos estamos ya (tanto el mundo entero como el Uruguay) de aquel modelo de negocio inicial planteado en el cual el mayor tráfico transcurría sobre las líneas de telefonía fija, y, por ende, las grandes ganancias de los operadores. La década del 90 particularmente presencié dos cambios sobre este modelo de negocios. Por un lado, el advenimiento del tráfico por concepto de navegación en Internet, y, por otro lado, el aumento en el tráfico por concepto de llamadas a teléfonos celulares y las propias llamadas generadas en éstos.

Todo esto hace inducir que el negocio para los operadores telefónicos ya no es más el de los teléfonos fijos, y que la atención de éstos apunta cada vez más hacia el control del tráfico en Internet así como brindar contenidos y generar minutos de aire en los servicios de telefonía celular.

Por otro lado, encontramos a los servicios de localización¹ como parte de los sistemas de información geográfica². Esto es, aquellos servicios orientados a usuarios que les permiten consultar información relacionada con la ubicación de diferentes elementos en una región geográfica. Estos elementos se dicen georeferenciados sobre esa región. Así, un usuario de estos servicios de localización, tiene a su disposición tanto la información de los elementos georeferenciados así como la posibilidad de efectuar diferentes tipos de procesamientos con éstos y su propia ubicación.

Ejemplos de tales procesos son: obtener ciertos elementos geográficos de interés que se encuentren a cierta distancia de la ubicación del usuario; poder obtener la ruta más corta entre dos puntos geográficos o bien simplemente tener la posibilidad de recibir notificaciones o publicidades según la ubicación actual de la persona.

Estamos hablando aquí de servicios que, pensados en los términos actuales, conjugan elementos tan diversos como: las páginas amarillas de una guía; un servicio de consulta telefónico 0900; un conjunto de mapas temáticos tanto callejeros como aquellos que indiquen comercios y servicios; por mencionar algunos. Esto quiere decir que para un usuario que desee obtener el tipo de información que un servicio de localización le podría dar, debe reunir una enorme cantidad de recursos, incluidos los mencionados y sumado el tiempo, para poder obtener una respuesta.

También vale la pena notar que prácticamente todo tiene una componente geográfica, por lo que los servicios de localización cubren una amplísima gama de servicios. A su vez, también es de notar la dificultad (y el tiempo) que le insume a una persona poder obtener una respuesta a una consulta de tipo geográfica. Y la dificultad solo aumenta a medida que se complican las consultas planteadas.

Dados estos dos contextos, por un lado el de la telefonía celular, y por el otro, el de los servicios de localización, parece al menos razonable plantear un proyecto que busque acercarlos. Es decir que surge como una cuestión importante acercar servicios de localización en tiempo real a los usuarios de teléfonos móviles, los cuales interactúen con éstos a través de la voz.

Si a esto se le agrega la poca o ninguna disponibilidad en el mercado actual de soluciones de este tipo, que involucren a los servicios de localización y a los usuarios móviles, tanto a nivel mundial y particularmente en el contexto del Uruguay, se tiene como mínimo un problema que merece estudiarse, analizarse e incluso diseñar una posible solución, y en la medida de lo posible, presentar un prototipo que muestre la viabilidad de lo planteado.

A esto es a lo que apunta el presente trabajo.

¹ Por mayor información ver [ESRI, Maptuit].

² Por mayor información ver [GIS].

2. Definición del Problema

2.1 Introducción

La idea del taller es la de desarrollar un Sistema de Información Geográfica (SIG) que brinde servicios de localización. Esto se realiza mediante el uso de: diversas tecnologías tales como de telefonía móvil; diversas herramientas como por ejemplo algoritmos de decodificación de direcciones no estructuradas (GeoParser); diversos modelos de datos como Bases de Datos Geográficas y sus correspondientes vistas y capas de datos; algún lenguaje de programación que permita trabajar con todas éstas; y ciertas tecnologías específicas, como ser de localización automática (caso del GPS) y su interacción con el sistema en general.

Asimismo, se espera que la interfaz con el usuario sea en lenguaje natural hablado, por lo que el sistema debe proveer una herramienta mediante la cual reconozca los diálogos del usuario, y otra mediante la cual le comunique las respuestas al usuario.

2.2 Objetivo

Estudio, diseño e implementación de los componentes imprescindibles para permitir a un usuario acceder a los servicios de localización del producto en cuestión mediante una comunicación bidireccional entre ambos en la cual el usuario sólo se expresa mediante descripciones habladas.

2.3 Resultados Esperados

Se espera la realización de un exhaustivo Análisis de Requerimientos y Documento de Diseño en notación Unified Modeling Language (UML)³ del sistema en su totalidad.

A su vez, se deberá construir un prototipo que se enfoque en ciertos componentes (definidos en la Sección **6.1 Propuesta de Prototipo**) del sistema.

A continuación se presentan dos casos posibles, reales, de utilización del sistema que se va a **Estudiar a Nivel de Análisis y Diseño**, es decir, para los cuales se presentará un Análisis y un Diseño pero no una implementación:

- Dada la necesidad de un usuario de llegar a un destino que no conoce, le pregunta al sistema cómo llegar a partir del lugar donde se encuentra actualmente. El sistema le responderá, brindándole un “ruteo” desde su ubicación actual hasta su destino. Para esto, el sistema se servirá de descripciones escritas o habladas de cómo tomar cada calle, e incluso de características notables sobre la geografía del área en cuestión.
- Un usuario solicita un taxi al sistema. Basado solo con esta “intención” del usuario, y su ubicación (tomada por ejemplo a través de un GPS), el sistema enviará una solicitud de despacho de taxímetro hasta la ubicación actual del usuario.

A continuación se presentan dos casos posibles, reales a **Implementar a Nivel de Prototipo**, es decir, para los cuales sí se proveerá un prototipo que los implemente:

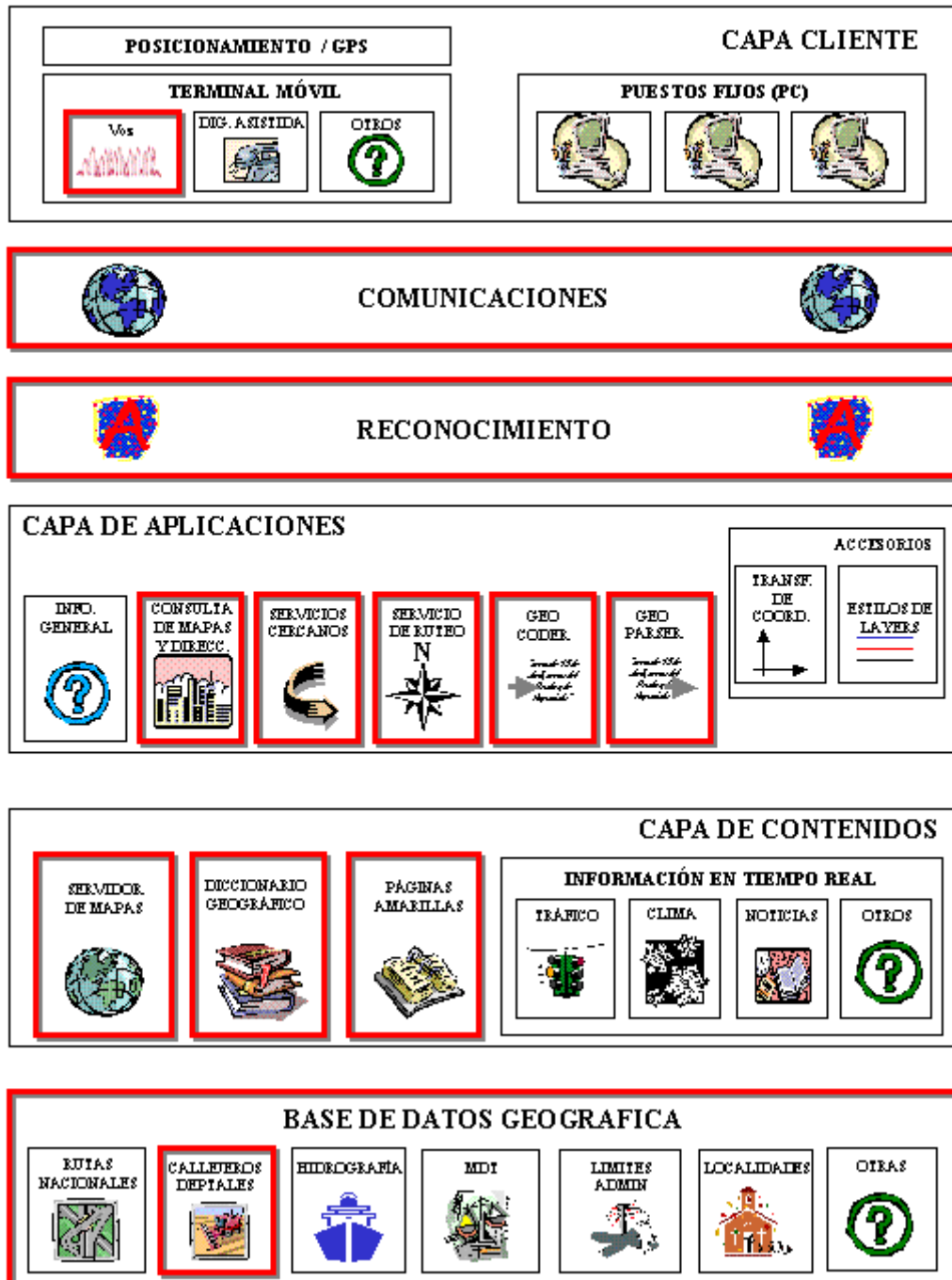
- Un usuario desde una computadora, y utilizando un micrófono, desea saber la ubicación exacta de una intersección de calles o de un lugar específico. Como respuesta a esto, el sistema le brindará la dirección exacta (o al menos una aproximación) de la ubicación en cuestión, a través del envío de un mapa, de una respuesta en pantalla o una respuesta de voz sintetizada.
- Otro caso de uso del sistema es si el mismo usuario desea saber qué farmacia se encuentra más cerca. En este caso, el sistema le respondería cuáles son las farmacias más cercanas a la ubicación dada por el usuario, dentro de cierto rango máximo de tolerancia, fuera del cual no se obtendrán resultados.

³ Por mayor información acerca del lenguaje UML en general, ver [Larman].

2.4 Módulos que componen el Sistema

A continuación se presenta un diagrama con los módulos del sistema. Dicho diagrama pretende solamente situar al lector en el contexto del presente proyecto. El diagrama no forma parte de ninguna especificación del problema (es decir que no forma parte de ningún diagrama válido del lenguaje UML), sino que sirve sólo a propósitos generales para comprender mejor el problema al cual nos enfrentamos y la posterior solución presentada.

Luego quedará clara cierta relación entre los módulos que aparecen en el siguiente diagrama y los componentes que efectivamente debe tener el sistema, pero debe quedar en claro que esto es “una casualidad”, y que tales componentes no necesariamente surgen a partir del presente diagrama.



2.5 Detalle de los Módulos

A continuación se comenta muy brevemente que se quiere decir y las características de cada uno de los módulos resaltados del diagrama anterior, los cuales se encuentran divididos en capas.

CAPA	MODULOS
Cliente	Terminal Móvil: <ul style="list-style-type: none"> • Voz: El usuario se comunica con el sistema a través de la voz. Por tal motivo surge la necesidad de una capa como la de Reconocimiento. • Localización: Se investigarán tecnologías tales como GPS, E-OTD, TOA, COO, AOA, etc. que le permiten al sistema conocer la ubicación exacta del usuario.
	Puesto Fijo: PCs conectadas a través de Internet por ejemplo, o directamente utilizando un micrófono interactuando con el sistema en el equipo al cual enfrentan.
Comunicaciones	Para los terminales fijos se puede pensar en utilizar Internet como la interfaz entre el cliente y capas anteriores. Para los terminales móviles se evaluarán diferentes protocolos que trabajan sobre plataformas móviles.
Reconocimiento	Permite convertir la voz de los terminales del cliente/usuario a texto, siendo este texto el dato de entrada para el sistema. A su vez, permite la operación inversa, convertir el texto resultante del sistema en voz para ser dictada al cliente.
Aplicaciones	Consulta de Mapas y Direcciones: Permite consultas directas sobre mapas así como el desplazarse dentro de los mismos y efectuar zooms. Para esto utiliza el Servidor de Mapas.
	Servicios Cercanos: Dada una ubicación permite la localización de diferentes servicios (como farmacias, estaciones de servicio, hoteles) próximos a la misma.
	Ruteo: Permite resolver un ruteamiento entre dos ubicaciones dadas, tomando en cuenta preferencias del usuario para resolver el ruteo.
	GeoCoder: Soluciona la georeferenciación de una dirección estructurada.
	GeoParser: Transforma una dirección no estructurada en una que sí lo es, para posterior ubicación, basándose en los datos introducidos por el usuario.
	Accesorios: Estos módulos podrían ser utilizados para transformaciones entre diferentes sistemas de coordenadas, así como disponer de diferentes estilos de capas de datos (layers) al momento de mostrar los datos al usuario. Por ejemplo, mostrar diferentes capas de datos dependiendo del dispositivo con el que cuenta el usuario.
Contenidos	Servidor de Mapas: Combina los contenidos de la capa anterior en una vista estándar de una zona geográfica específica, sirviendo de esta manera de un mapa para ser desplegado por las capas superiores.
	Diccionario Geográfico: Contiene datos que se basan en los de la BD Geográfica y que son usados directamente por el sistema. Brindan otro tipo de información al sistema, la cual puede adoptar la estructura deseada para así ser de gran utilidad para éste. También puede versele como una vista diferente de los datos contenidos en la BD Geográfica.
	Páginas Amarillas: Brinda información acerca de: servicios y empresas comerciales, sitios de interés, lugares turísticos, etc.
	Información en Tiempo Real: Estos módulos podrían ser utilizados para procesar datos en tiempo real, como ser datos del tiempo, noticias y tráfico. Tal análisis queda por fuera del presente proyecto, pero es de interés al menos plantear su existencia.
Base de Datos Geográfica	Callejero Departamental: Contiene la información de todas las calles dentro de un departamento o zona en general. Por ejemplo, la información de las calles de Montevideo. A su vez, una BD Geográfica puede contener datos acerca de la hidrografía, del terreno, de los límites administrativos, etc.

2.6 Plan de Trabajo para el Año 2001

MES	ACTIVIDAD
Marzo	Introducción al problema planteado en el proyecto.
Abril-Julio	Recopilación de material.
Agosto	Definición del prototipo a implementar.
Setiembre-Octubre	Investigación y familiarización con las herramientas necesarias para el desarrollo del prototipo (en particular el ESRI-MapObjects).
Noviembre-Diciembre	Implementación del prototipo y generación de documentos técnicos internos que mostrasen los resultados obtenidos en cada etapa del prototipo así como los detalles acerca de la implementación del mismo, los cuales luego conformarían parte del Capítulo 6 del Informe Final.

2.7 Plan de Trabajo para el Año 2002

MES	ACTIVIDAD
Enero-Marzo	Continuación de la etapa de implementación del prototipo y en forma paralela recopilación y generación de documentación preliminar.
Abril	Estudio del lenguaje UML y en forma paralela su utilización en los documentos.
Mayo-Julio	Generación del Documento de Análisis y Diseño en UML y en forma paralela confección del Informe Final.
Agosto	Finalización y presentación del Informe Final del proyecto.

2.8 Documentación

El Informe Final se realizará en forma incremental, es decir que los Informes Parciales se desarrollarán paulatinamente hasta conformar el Informe Final. Cada nuevo informe especificará los cambios y agregados sobre el anterior, para facilitar su lectura.

3. Requerimientos

3.1 Introducción

Esta Sección tiene como objetivo establecer los Requerimientos del Sistema. Estos son una descripción de las necesidades de un producto o sistema. Su meta es la de identificar y documentar lo que en realidad se necesita, en una forma lo suficientemente clara como para comunicársela a los usuarios (o clientes) del sistema.⁴

Los requerimientos comienzan por cinco diferentes secciones, destinadas a lograr el objetivo mencionado en el párrafo anterior. Las cinco secciones son:

- **Panorama General** (objetivos generales planteados en proyecto)
- **Clientes/Usuarios** (quienes serán las personas que interactuarán directamente con el sistema)
- **Metas** (metas del sistema y beneficios para los actores de cumplirse las metas)
- **Casos de Uso de Alto Nivel** (descripciones de los procesos más importantes del sistema en un lenguaje neutral)
- **Requerimientos Alternativos** (otras posibles utilizaciones o implementaciones del sistema en cuestión, en particular la interfaz entre el sistema y el usuario)

Son de notar los Casos de Uso, ya que el artefacto central mediante el cual plasmar los procesos que debe realizar el sistema, es a través de los Casos de Uso. Un Caso de Uso es una descripción narrativa textual de un proceso del sistema. Es decir, que los casos de uso sólo contienen frases en lenguaje natural y en idioma español, y su finalidad es establecer en forma escrita y a grandes rasgos, cuales son las funcionalidades que debe cumplir el sistema.

Vale notar que un Caso de Uso representa un *proceso*, mientras que una Funcionalidad representa una *operación* indivisible del sistema. Es decir, que un proceso contendrá una o mas funcionalidades.

3.2 Panorama General

Este proyecto tiene como objeto, como se mencionó en el Capítulo 2, el estudio, diseño e implementación de los componentes imprescindibles para permitir a un usuario acceder a servicios de localización mediante una comunicación bidireccional entre ambos en la cual el usuario sólo se expresa mediante descripciones habladas.

3.3 Clientes/Usuarios

Los clientes o usuarios del sistema son todas aquellas personas que dispongan de un teléfono celular y que se encuentren “afiliadas” al sistema. No olvidar que el otro actor participante en este “escenario”, son las empresas de telefonía celular, también generalmente denominadas operadores telefónicos. Debe quedar en claro que estas empresas no son los usuarios finales del sistema, sino que son el medio por el cual éstos utilizarán el sistema.

3.4 Metas

La meta del sistema es brindar acceso a los servicios de localización a usuarios móviles de manera de obtener los siguientes beneficios para todos los actores participantes (usuarios y empresas operadoras de telefonía celular):

- Disponibilidad en forma inmediata (dada por la utilización de dispositivos tales como GPS y por ser un sistema on-line) y desde cualquier lugar (dada la ubicuidad y movilidad de los teléfonos celulares) de información geográfica (en particular, servicios de localización).
- Facilidad de interacción usuario – producto gracias a la interfaz en lenguaje natural hablado que brinda una inigualable agilidad y comodidad para el usuario (es difícil pensar otra interfaz de comunicación que resulte más natural a un usuario que su propia voz).

⁴ Por mayor información acerca de la Etapa de Requerimientos, ver [Larman] Capítulo 5.-

- Introducirse a nivel regional en el uso avanzado de la tecnología celular así como volcar experiencia al mercado en la utilización de contenidos de tipo SIG. Un sistema con las características descritas en este documento no ha podido ser encontrado en ningún lugar del planeta, al menos a juicio del autor.
- Para los operadores de telefonía celular, ya que aumentarán el interés de los usuarios por dichos servicios (dado que incluyendo un sistema de este tipo estarían introduciendo un contenido muy atractivo para los usuarios) así como aumentar la cantidad de minutos de aire utilizados por los mismos (ya que se trata de comunicaciones telefónicas celulares corrientes, y la forma natural que surge de cobrar dicho sistema es a través de los minutos de aire, más una posible “suscripción”)

3.5 Casos de Uso de Alto Nivel del Sistema

A continuación se presentan los Casos de Uso de Alto Nivel correspondientes al sistema.⁵ Vale recordar que un Caso de Uso modela un proceso del sistema, ya no una funcionalidad, sino más bien un conjunto de funcionalidades que conforman un proceso.

Por estar dentro de la etapa de Requerimientos, los siguientes CU pretenden solamente ser informativos para el usuario, y simplemente describir en términos generales, las funcionalidades del sistema.

Luego, en secciones posteriores, más particularmente en el Capítulo de Análisis y Diseño, se discutirán los CU Expandidos para cada CU de Alto Nivel presentado aquí, los cuales muestran en detalle las interacciones entre el usuario y la interfaz del sistema, es decir su presentación.

CU de Alto Nivel:	1. Consulta de Direcciones y Puntos de Interés.
Actor:	Usuario.
Tipo:	Primario.
Descripción:	Este CU permite al usuario ubicar direcciones y puntos de interés en un mapa. Las direcciones pueden ser dadas de varias formas. También el usuario podrá buscar los puntos de interés de un mapa, como ser lugares notables, accidentes geográficos importantes o sitios muy conocidos. Como respuesta a esto, el sistema devolverá la ubicación exacta y la posibilidad de dar una ruta que tenga como origen la ubicación actual del usuario y como destino la dirección devuelta o el punto de interés buscado, utilizando para esto el CU Ruteo, una vez que se localizó exactamente la ubicación de destino. Podríamos comparar este servicio con el de un mapa “inteligente” al cual consultado sobre cierta dirección o punto de interés, el mismo nos mostraría donde éste se encuentra y hasta cómo llegar.
CU de Alto Nivel:	2. Servicios Cercanos.
Actor:	Usuario.
Tipo:	Primario.
Descripción:	Este CU permite al usuario ubicar los comercios (o en forma general, productos y servicios) más cercanos a una determinada ubicación. La misma puede ser tanto dada por el usuario (en el caso de que éste no se encuentre en tal ubicación) o puede considerarse la ubicación actual de éste. En este último caso, el sistema obtendrá automáticamente su ubicación a través de un dispositivo de localización y no se deberá ingresar la misma y solo se preguntará que servicios le interesa al usuario y a qué distancia de la ubicación dada/actual. Por lo tanto, el resultado del sistema se restringirá a los comercios del rubro dado por el usuario dentro de un radio también dado por el usuario de la ubicación dada/actual. En este caso, cabría comparar este servicio a consultar a una persona local, es decir del lugar, acerca de dónde conseguir cierto producto, servicio o comercio dentro de la zona.
CU de Alto Nivel:	3. Ruteo.
Actor:	Usuario.
Tipo:	Primario.
Descripción:	Este CU permite al usuario obtener la mejor ruta entre dos ubicaciones dadas por éste al sistema. La mejor ruta es calculada por el sistema en base a ciertas preferencias del usuario, ya que el concepto de “mejor” es distinto de usuario a usuario. Las ubicaciones que el usuario debe dar al sistema son dos: la de origen, y

⁵ Por mayor información acerca de la estructura de un Caso de Uso, ver [Larman] Capítulo 6.-

la de destino. Si la ubicación de origen es la ubicación actual del usuario, la misma puede ser obtenida automáticamente a través de un dispositivo de localización en cuyo caso el sistema solo preguntará la ubicación de destino. Nuevamente, podríamos comparar este servicio con una consulta a un mapa “inteligente” marcándole dos puntos (origen y destino), así como alguna preferencia (tiempo, distancia, consumo de combustible, etc.). El mismo mostraría la mejor ruta entre los dos puntos tomando en cuenta la preferencia dada.

CU de Alto Nivel:	4. Páginas Amarillas.
Actor:	Usuario.
Tipo:	Primario.
Descripción:	Este CU permite al usuario ubicar y obtener información acerca de comercios (de venta de productos y servicios) que cumplan con cierto criterio dado por el usuario. Esto es, ni más ni menos, que una consulta en una Guía de Páginas Amarillas electrónica, en donde, en lugar de buscar en forma secuencial el producto o servicio que el usuario desea, se ingresa un criterio de búsqueda que permite restringir la cantidad de resultados devueltos por el sistema. Por tanto, el usuario debe ingresar datos como la rama del comercio buscado y el rubro dentro de esa rama. No se debe confundir este CU con el de Servicios Cercanos, quien busca comercios que se encuentren a cierta distancia de una ubicación, mientras que Páginas Amarillas devuelve un conjunto de comercios que cumplen cierto criterio de búsqueda, no relacionado con la ubicación del usuario.

CU de Alto Nivel:	5. Publicidad Móvil.
Actor:	Usuario.
Tipo:	Primario.
Descripción:	Este CU permite al usuario recibir información acerca de ciertos productos o servicios de su preferencia que se encuentran disponibles en un comercio cercano a su ubicación actual. Cabe destacar que este CU no tiene al usuario como iniciador del mismo, sino que éste es notificado a través de un mensaje publicitario originado por su ubicación y basado en su perfil de preferencias previamente ingresado al sistema. De esta manera, se permite efectuar una publicidad móvil en forma pro-activa, ya que es el sistema quien toma la iniciativa de notificar al usuario, aunque únicamente sobre las preferencias de éste. El usuario no envía datos al sistema al momento de recibir la notificación, mientras que el sistema avisa del comercio adecuado a sus preferencias y su ubicación e incluso el camino de cómo llegar a tal destino tomando como origen la ubicación actual del usuario.

3.6 Requerimientos Alternativos

Esta Sección pretende mostrar requerimientos alternativos para el sistema. Esto es, analizar posibles alternativas con las cuales se podría haber pensado el presente sistema. La idea es solo presentarlas y discutir las similitudes, diferencias, ventajas y desventajas entre estas alternativas y la propuesta del proyecto. O sea que no se realizará un análisis o diseño de estas alternativas, sino que se mostrará su existencia y porqué se eligió la que se eligió.

El énfasis en los requerimientos alternativos se pondrá en la interfaz entre el sistema y el usuario, es decir en la forma de comunicación entre éstos. Debe quedar claro que la interfaz elegida fue la hablada, y que el usuario genera peticiones habladas al sistema y éste las contesta también en forma hablada. De ahí la fuerte utilización y dependencia de la telefonía celular actual, y que no se necesite ninguna tecnología especial de transmisión de datos (lo cual presenta una ventaja importante si se piensa en concepto del “Time to Market”, de vital importancia para que una empresa obtenga una ventaja competitiva sobre su competencia).

Una alternativa para la interfaz entre el sistema y el usuario es una en que el usuario escribe en forma de texto lo que le requiere al sistema, y éste le responde en forma de mapas con las ubicaciones o las rutas deseadas. Incluso es innegable que tal sistema presenta sus resultados de una manera mucho más útil y atractiva para el usuario, ya que en vez de hablarle las direcciones, comercios o rutas, se las mostraría en un mapa. Por ejemplo, si el usuario deseara conocer los restaurantes cercanos a su ubicación, le indicaría al sistema por medio de una opción de menú que desea utilizar el servicio de cercanías, y luego ingresaría el rubro restaurantes y la distancia máxima que está dispuesto a desplazarse hacia el más cercano. Como resultado a esto, el sistema le devolvería un mapa de la zona en donde se encuentra ubicado el usuario, indicando gráficamente exactamente donde éste se encuentra, y marcando los restaurantes que se encuentran a esa distancia de sí.

Incluso otra alternativa más atractiva sería una en que el usuario le hablara al sistema, y que éste le conteste en forma de mapas. Esto sería una mezcla de dos interfaces de comunicación, una hablada (por parte del usuario) y otra de datos en forma de mapas (por parte del sistema). La principal ventaja de un sistema de este tipo es justamente utilizar la forma de comunicar mas adecuada para una y otra parte. El usuario preferirá utilizar descripciones habladas, mientras que para el sistema será no solo más fácil, sino que mejor para el usuario, mostrar los resultados obtenidos en un mapa. No obstante, las desventajas que tal sistema conlleva tampoco son difíciles de ver. Todas estas se resumen en el siguiente cuadro.

Se incluye a continuación un cuadro comparativo de estas dos alternativas, más la adoptada, mostrando ventajas y desventajas de cada una de ellas.

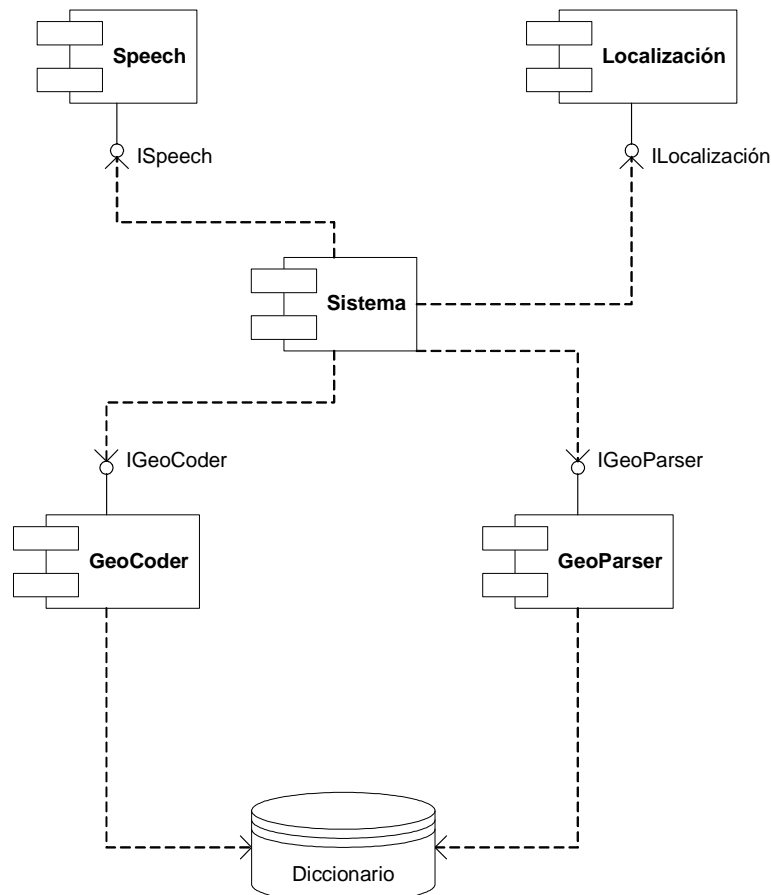
ALTERNATIVA DE INTERFAZ	VENTAJAS	DESVENTAJAS
Interfaz hablada entre usuario y sistema (la elegida)	<ul style="list-style-type: none"> •Facilidad técnica para implementar la solución ya que solo se transmite voz. •Interfaz hablada es la natural para todo ser humano. •No necesita ningún dispositivo especial, solo un teléfono celular. •Rapidez en la transmisión de la voz. •Costo de un teléfono celular con GPS es inferior (al menos en la actualidad) que el costo de un dispositivo tipo PDA (Personal Digital Assistant) con GPS. 	<ul style="list-style-type: none"> •El resultado hablado es inferior a devolver un mapa, aunque esta afirmación no está exenta de discusión. •La capacidad de procesamiento de un teléfono celular es casi nula frente a un dispositivo tipo PDA, por lo que éste último se vería beneficiado por su mayor poder de procesamiento, mientras que un teléfono celular estándar solo podría enviar su voz como una señal de voz. Y, en caso de necesitarse cierto procesamiento en el celular, debe disponerse de una nueva tecnología.
Interfaz de datos entre ambos (el usuario escribe y recibe un mapa)	<ul style="list-style-type: none"> •Un resultado devuelto en un mapa es superior a un resultado hablado. 	<ul style="list-style-type: none"> •Necesita un dispositivo capaz de desplegar un mapa, por ejemplo un PDA. •No se utiliza la interfaz hablada que es la natural para los humanos. •Mayor demora en la respuesta dado que se transmite un mapa. •Costo del dispositivo tipo PDA.
Interfaz “mixta” (usuario habla y recibe un mapa como respuesta)	<ul style="list-style-type: none"> •Interfaz hablada es la natural para todo ser humano. •Un resultado devuelto en un mapa es superior a un resultado hablado. •La capacidad de procesamiento de un dispositivo tipo PDA es mayor a la de un celular, pudiéndose realizar cierto pre-proceso en los datos que se envían. 	<ul style="list-style-type: none"> •Necesita un dispositivo capaz de desplegar un mapa, por ejemplo un PDA con capacidad de voz. •Mayor dificultad técnica ya que se debe transmitir datos y voz a través de un PDA. •Mayor demora en la respuesta dado que se transmite un mapa. •Costo del dispositivo tipo PDA.

Como se aprecia en el cuadro, cada alternativa presenta importantes ventajas y desventajas nacidas de las diferencias entre ellas. También se aprecia, que según las características analizadas en el cuadro, la opción elegida es la que presenta mas ventajas y menos desventajas, lo cual avala su elección.

En el Capítulo 8 (Conclusiones) se retomará brevemente esta discusión, y se harán referencias a los datos aquí expuestos.

4. Arquitectura Propuesta

A continuación se presenta un Diagrama de Componentes del lenguaje UML que muestra una posible arquitectura a adoptar al momento de llevar a cabo la implementación del presente proyecto en su totalidad.⁶



La misma comprende los siguientes componentes:

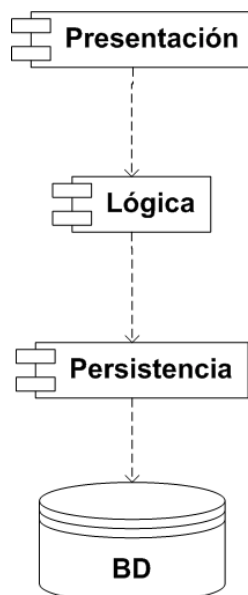
- **Speech:** El componente Speech representa a la parte del sistema encargada de las conversiones de voz a texto así como de texto a voz. Este componente debe exportar cierta interfaz (ISpeech) con métodos que permitan al implementador solucionar todos los problemas relacionados a la utilización de las tecnologías de reconocimiento de voz disponibles en dicho componente.
- **Localización:** El componente Localización representa a la parte del sistema encargada de ubicar geográficamente al usuario. Es decir, que brinda la posición del usuario dentro de un determinado mapa. Las formas particulares de localizar al usuario son varias, destacándose en este proyecto la utilización de un dispositivo de posicionamiento global (GPS). También debe exportar una interfaz (ILocalización) con métodos que permitan responder a la pregunta: ¿Dónde está ubicado el usuario en este momento?
- **Sistema:** Este componente representa al propio sistema. Sirve de guía para mostrar las diferentes interacciones entre los componentes. Por ser quien utiliza las interfaces de los componentes, de por sí no presenta interfaces.

⁶ Es decir que aquí, en este Capítulo, no se hace referencia en ningún momento al Prototipo, sino siempre a todo el sistema en toda su extensión.

- **GeoCoder:** El componente GeoCoder representa la parte del sistema encargada de ubicar un objeto en el mapa dada una descripción estructurada del mismo. Típicamente, dada una dirección estructurada (nombre de la calle, número de puerta, etc., todos con cierto formato particular), el componente GeoCoder debe ser capaz de ubicar geográficamente tal dirección. Como los anteriores, debe proveer una interfaz (IGeoCoder).
- **GeoParser:** El componente GeoParser representa la parte del sistema encargada de interpretar los datos que el usuario ingresa al sistema. Ya sea que el usuario desee obtener un servicio de localización, o de cercanías, o deba introducir nombres de calles o rubros, este componente es el encargado de convertir estos datos no estructurados y *a priori* sin valor, en datos estructurados y con sentido para el sistema. De alguna forma, es quien “parsea” lo que el usuario dicta, luego de ser convertida la voz a texto, de ahí su nombre. También debe proveer una interfaz (IGeoParser) de cuyos métodos se espera que brinden todos los servicios necesarios para cumplir con dicha tarea.
- **Diccionario:** El componente Diccionario representa la parte del sistema encargada de almacenar los datos. Esto implica guardar los datos que conforman los mapas y cualquier otro dato que sea de interés mantener por el sistema, obviamente que esté georeferenciado dentro del mapa. Se ha utilizado otro estereotipo (figura de una BD) tal como lo permite el UML.

La ventaja de utilizar interfaces y no directamente las diferentes operaciones brindadas en forma directa de los objetos, es que si en el futuro se desea reemplazar cualquier componente por otro que haga el mismo trabajo que el actual pero de manera mas eficiente, mas clara, mas rápida, o cualquier mejora que podamos imaginar, hay que cambiar solamente ese componente, y no hay que cambiar absolutamente nada de los demás componentes. Sí y solo sí el nuevo componente exporta una interfaz con las mismas operaciones que exportaba el componente actual. Es decir, si el nuevo componente dice que hace lo mismo que el actual, léase que exporta exactamente la misma interfaz, entonces para utilizarlo sólo es suficiente reemplazarlo, sin tocar nada de los demás.

A continuación, y en forma resumida, se muestra un diagrama de una arquitectura en tres capas, en donde cada capa se responsabiliza de una tarea en particular y muy bien diferenciada del resto. En particular, se mostrará el modelo de arquitectura con tres capas: Capa de Presentación, que es la encargada de manejar la interfaz del usuario y las interacciones que éste tiene directamente sobre el sistema; la Capa Lógica, que contiene los objetos que resuelven los problemas que el sistema está hecho para, y que no incluye ningún tipo de interfaz con el usuario, sino que se comunica solamente con su capa superior (Presentación) y su capa inferior (Persistencia); y por último, la Capa de Persistencia, quien es la encargada de almacenar todos los datos manejados por el sistema.



La motivación de introducir tal diagrama, es que en el presente proyecto se analizará el problema, y se le diseñará una solución (Capítulo 5) teniendo en mente esta estructura en tres capas que se acaba de exponer.

5. Análisis & Diseño

Introducción

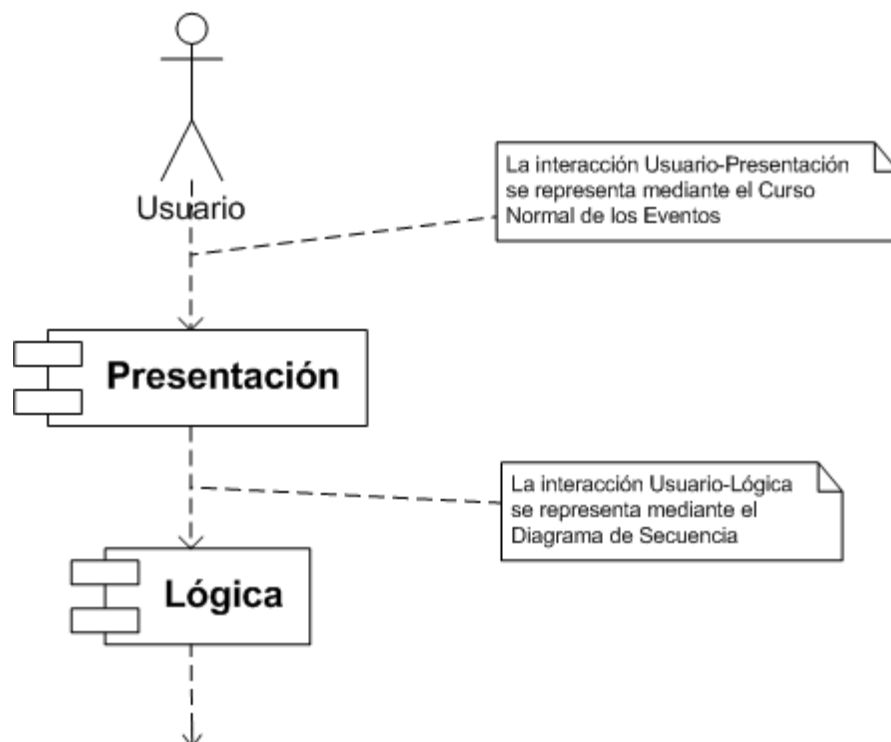
Este Capítulo contiene las etapas de Análisis y Diseño para cada Caso de Uso (CU) del Sistema. Por tanto, se discute cada CU junto con su Análisis y su Diseño en lenguaje UML. Todo lo referente a estas dos etapas de desarrollo se encuentra en este Capítulo.

Para cada CU, se comenzará describiendo su correspondiente **Caso de Uso Expandido** (Secciones 5.x.1, con $1 \leq x \leq 5$) los cuales están basados en los Casos de Uso de Alto Nivel descritos en la Sección 3.7.

Luego, se describen los **Diagramas de Secuencia**⁷ (Secciones 5.x.2, con $1 \leq x \leq 5$) del Sistema para cada posible Curso de Eventos del Caso de Uso, por lo que los diagramas dependen directamente de los Casos de Uso Expandidos, aunque representan únicamente la interacción entre la Capa de Presentación y la Capa Lógica del Sistema, y no la interacción entre el Usuario y la Capa de Presentación como lo hacen los Casos de Uso Expandidos.

Es importante aclarar este punto, ya que los mensajes que se verán en los Diagramas de Secuencia no se corresponden directamente con las interacciones mostradas en los Casos de Uso Expandidos, ya que éstos últimos representan las interacciones entre el usuario y la interfaz del sistema. Tal interacción, aunque es necesaria de detallar en los Casos de Uso, no es de interés al momento de encarar la lógica del sistema, es decir su Capa Lógica. Por lo tanto, son las interacciones entre la Capa de Presentación y la Capa Lógica lo que interesa al momento de confeccionar los Diagramas de Secuencia.

A continuación se presenta un diagrama que ilustra las interacciones del Usuario directamente con el Sistema, mediante la Capa Presentación, y las interacciones del Usuario indirectamente con el Sistema, mediante la Capa Lógica, y qué artefacto UML representa dichas interacciones. No se representa, por simplicidad, la Capa de Persistencia.



⁷ Por mayor información acerca de los Diagramas de Secuencia, ver [Larman] Capítulo 13.-

Luego, se definen los **Diagramas de Colaboración**⁸ (Secciones 5.x.3, con $1 \leq x \leq 5$) para cada Evento del Sistema, obtenidos éstos de los Diagramas de Secuencia del Sistema. Para los casos del presente proyecto, cada Diagrama de Secuencia posee un solo Evento del Sistema, por lo que los Diagramas de Colaboración comenzarán con este evento, para luego poder ver cómo los objetos de software interactúan entre sí a través de mensajes para poder llevar a cabo el evento original.

Luego, basándose en las colaboraciones, se elaboran los **Diagramas de Clases**⁹ (Secciones 5.x.4, con $1 \leq x \leq 5$) de Diseño del Sistema, los cuales muestran el diseño necesario de las clases que intervienen en ese Caso de Uso.

Finalmente, en la Sección 5.6, se elaborará el **Diagrama general de Clases de Diseño**, el cual representa un resumen de los diagramas individualmente obtenidos en el análisis y diseño de cada Caso de Uso.

A modo de resumen, para cada Caso de Uso, este Capítulo introduce cuatro artefactos del UML:

1. Caso de Uso Expandido
2. Diagrama de Secuencia
3. Diagrama de Colaboración
4. Diagrama de Clases

Además, para todos los Casos de Uso se introduce el artefacto de diagrama general de clases.

El primer Caso de Uso (denominado Servicio para darle un enfoque menos técnico) analizado y diseñado (Consulta de Direcciones y Puntos de Interés) se encuentra particularmente detallado por ser el primero al cual el lector se enfrentará.

Por lo tanto, se denota su importancia frente al resto ya que contiene explicaciones introductorias que se aplicarán al resto de los servicios analizados y diseñados.

Es sin duda el servicio que mayor atención requerirá para poder entender de la mejor forma los siguientes.

⁸ Por mayor información acerca de los Diagramas de Colaboración, ver [Larman] Capítulo 17.-

⁹ Por mayor información acerca de los Diagramas de Clases, ver [Larman] Capítulo 21.-

5.1 Consulta de Direcciones y Puntos de Interés

5.1.1 Caso de Uso Expandido

CU Expandido:	Consulta de Direcciones y Puntos de Interés.
Actor:	Usuario (iniciador).
Propósito:	Permitir ubicar direcciones y puntos de interés en un mapa y como alcanzarlos.
Resumen:	Permite consultar dentro de mapas urbanos y carreteros de una ciudad, dando la calle y el número de puerta, la intersección de dos calles, el nombre de un punto de interés dentro de la ciudad o cualquier descripción que permita al sistema ubicar dentro de un determinado contexto el lugar a buscar. A partir de estos datos, el sistema ubicará (o aproximará) la dirección, y le responderá al usuario ya sea cómo llegar a ella a partir de su ubicación actual; o qué elementos importantes de la geografía local se encuentran en las inmediaciones para poder ubicar con mayor facilidad la dirección ingresada.
Tipo:	Primario y esencial.

Curso Normal de los Eventos:

Se debe tener en cuenta que el presente (y los subsiguientes) Curso Normal de los Eventos representa la interacción entre el usuario y la interfaz con éste del sistema. Es decir, que muestra la interacción entre el usuario y la Capa Presentación del sistema. Por tanto, los pasos mostrados contienen un considerable nivel de detalle.

No se debe confundir con la interacción entre la Capa Presentación y la Capa Lógica. Ésta se representa mediante los eventos contenidos en los Diagramas de Secuencia.

Acción del Actor	Respuesta del Sistema
1. Se conecta al sistema	2. Se presentan los tipos de servicios en forma hablada
3. Inicia Consulta de Direcciones	4. Pregunta el elemento geográfica (la dirección o punto de interés) a buscar
5. Se ingresa la dirección o el punto de interés buscado y la respuesta esperada, ya sea cómo llegar a partir de la ubicación actual o qué otros elementos geográficos notables existen cerca	6. Se toman los datos
	7. Se intenta ubicar dicha dirección o punto
	8. Se calcula la respuesta en base a los resultados esperados por el usuario
	9. Se devuelve el resultado en forma hablada

5.1.2 Diagrama de Secuencia

La idea de un Diagrama de Secuencia es la de representar la interacción de mensajes entre la Capa de Presentación y la Capa Lógica. Por tanto, los mensajes aparecidos en los Diagramas de Secuencia se toman de los Casos Expandidos de Uso, en particular del Curso Normal de los Eventos y de los Cursos Alternativos más importantes, aunque no directamente, ya que como se explicó, éstos Cursos contienen la interacción detallada entre el Usuario y la Capa de Presentación del Sistema.

Por ésto, podría verse a los Diagramas de Secuencia como una abstracción de los mensajes más importantes entre Usuario y Sistema, olvidando los detalles de cómo se implementan las interfaces de usuario.

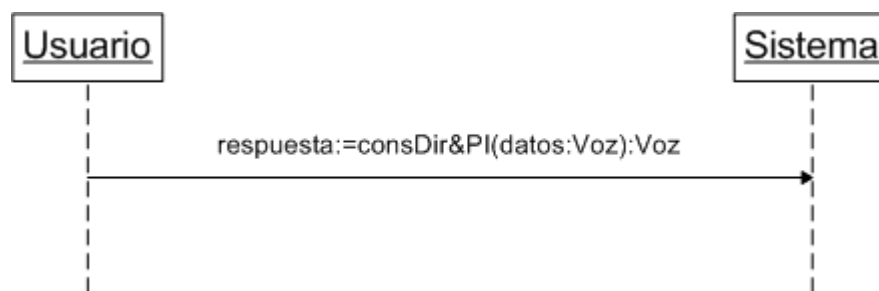
A modo de resumen, se explica la sintaxis utilizada para representar los mensajes:

```
<objeto_devuelto> :=
  <nombre_mensaje> ( <parámetro> : <tipo_parámetro> ) : <tipo_objeto_devuelto>
```

La siguiente tabla explica cada elemento de la sintaxis:

objeto_devuelto	Nombre del objeto que devuelve el mensaje.
nombre_mensaje	Nombre que identifica al mensaje.
parámetro	Parámetro de entrada que se le pasa al mensaje.
tipo_parámetro	Tipo de dato del parámetro de entrada.
tipo_objeto_devuelto	Tipo de dato del objeto que devuelve el mensaje.

A continuación se presenta el Diagrama de Secuencia para el Caso de Uso Consulta de Direcciones y Puntos de Interés.



En este caso en particular (consulta de direcciones y PI), vemos como el Usuario le pide al Sistema una consulta de direcciones o puntos de interés (`consDir&PI`), para lo cual debe de proveer ciertos datos en forma hablada (`Voz`), y cuya respuesta también será en forma hablada.

5.1.3 Diagrama de Colaboración

Un Diagrama de Colaboración pretende explicar gráficamente las interacciones que existen entre los objetos de un sistema. En particular, muestra las responsabilidades con las que debe cumplir cada objeto de manera de proveer de operaciones a los demás objetos que las utilizan.

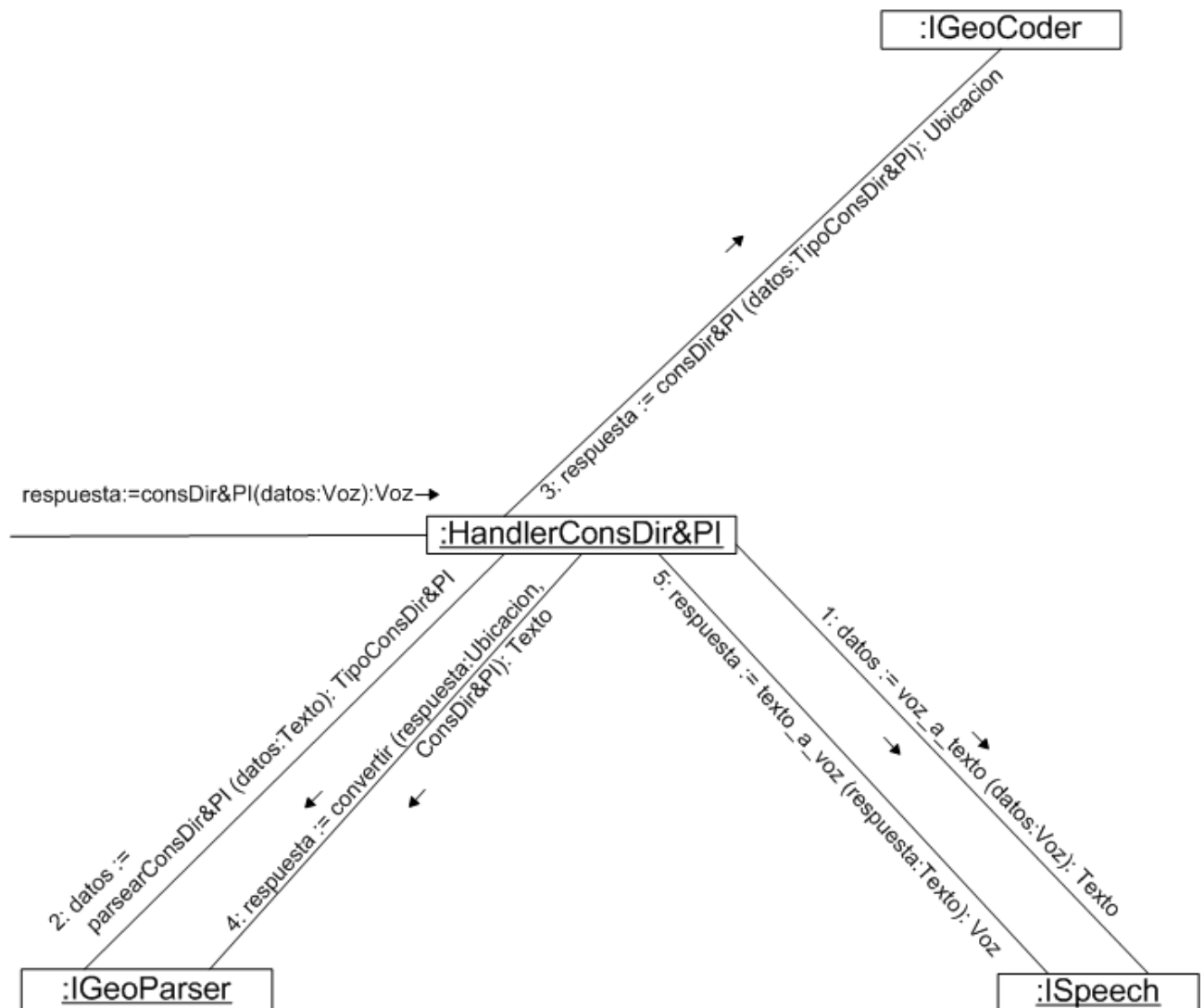
En nuestro caso, se hace uso de un objeto manejador de los demás, llamado `HandlerConsDir&PI` que es quién se encarga de la coordinación de las colaboraciones a través de mensajes. Existen otras formas dentro del UML de delegar responsabilidades, siendo ésta una de ellas.

A esta forma de asignar responsabilidades se la denomina Patrón Controlador (*Controller Pattern*) y viene de los patrones para asignar responsabilidades del UML, también conocidos como GRASP¹⁰ (General Responsibility Assignment Software Patterns). Esto quiere decir, que existirá un objeto UML encargado de coordinar y “controlar” todas las interacciones pertinentes a un caso de uso en particular, habiendo un objeto controlador por cada caso de uso del sistema.

En particular, estos objetos responden a la siguiente pregunta: ¿Quién deberá encargarse de atender un evento del sistema?. Para contestarla, es que utilizamos los controladores, quienes son objetos de interfaz no destinada al usuario, que se encargan de manejar un evento del sistema.

La convención de nombres seguida a lo largo de este Capítulo, será la de denominar a tales controladores como “Handler” seguido del nombre del caso de uso. Por ejemplo, `HandlerConsDir&PI`. Luego, es de esperar que tal objeto controlador se convierta en una clase que sea la que maneje todas las interacciones necesarias para llevar adelante el caso de uso pertinente, pero esto ya forma parte de la implementación del sistema, y queda en manos del implementador. Lo mismo se repetirá para todos los casos de uso.

¹⁰ Por mayor información acerca de los patrones GRAPS ver [Larman] Capítulo 18.



El presente Diagrama de Colaboración muestra las interacciones necesarias para poder llevar a cabo una consulta de dirección o punto de interés. A continuación se detallan y explican los pasos:

0. Todo Diagrama de Colaboración comienza con un mensaje originado del evento del Diagrama de Secuencia. En este caso, el usuario espera una `respuesta` en forma de `Voz`, por lo que se llama al mensaje `consDir&PI()` sobre el manejador de Consulta de Direcciones y Puntos de Interés (`HandlerConsDir&PI`), enviándole los datos en forma de `Voz` conteniendo lo que el usuario le dicta al Sistema. Este manejador concentra los esfuerzos de comunicación entre los diferentes objetos, por lo cual es éste quien invoca a los mensajes de éstos. Cabe destacar que la respuesta de tipo voz que devuelve este mensaje puede ser corroborada. Si se sigue la lógica de invocaciones de mensajes (mensaje 1, 2, ...) se debe advertir que el dato de voz original es convertido a una respuesta de voz final que contiene la respuesta del sistema en formato de voz que el usuario espera. De ahí que lo devuelto por este mensaje es el objeto `respuesta`, de tipo `Voz`.
1. El primer mensaje es enviado al componente `Speech` a través de la interfaz de reconocimiento de voz (`ISpeech`) pidiéndole que convierta los datos en forma de voz dictados por el usuario en datos de tipo `Texto`. Para esto se utiliza el mensaje llamado `voz_a_texto()` que recibe los datos en formato de `Voz` y los devuelve en formato `Texto`. Este mensaje siempre es llamado con el dato de voz como parámetro de entrada, y siempre devuelve el mismo dato convertido en texto, por lo que no es necesario pasar ningún parámetro extra indicando de qué servicio se trata, ni es necesario contar con varios mensajes. Con un mensaje solo es suficiente, ya que siempre recibe y devuelve el mismo tipo de dato.

2. Inmediatamente después, estos datos de tipo texto son enviados por medio de un mensaje (`parsearConsDir&PI()`) al componente `GeoParser`, a través de la interfaz del `GeoParser` (`IGeoParser`) esperando que devuelva los datos procesados, es decir, parseados. En esta etapa es que el `GeoParser` intentará reconocer nombres de calles, las estructuras formadas por las calles (calle esq. calle, por ejemplo), nombres de lugares notables o puntos de interés, etc. El tipo de dato devuelto por este mensaje es estructurado, ya que contiene los datos “útiles” para el servicio de consulta de direcciones y puntos de interés en particular, no sirviendo estos datos para los otros servicios, y que serán inmediatamente pasados al `GeoCoder`. En este caso en particular, el tipo de datos se definió como `TipoConsDir&PI`, y en cada servicio (Caso de Uso) se utilizará uno diferente y especial para cada servicio. Esto es necesario ya que los objetos devueltos por los servicios no son siempre los mismos. Por ejemplo, el servicio de Ruteo busca identificar en el texto de entrada una dirección específica, mientras que servicios cercanos busca identificar además de una dirección, un rubro de comercios. También es importante notar que el nombre del mensaje debe ser diferente para cada caso de uso, ya que el tipo de dato devuelto no es siempre el mismo.
3. Estos datos estructurados son pasados al componente `GeoCoder` a través de su interfaz (`IGeoCoder`). Este ubicará dentro del mapa los objetos geográficos encontrados dentro de los datos del usuario. Por eso los datos devueltos son de tipo `Ubicacion`. El mensaje utilizado para esto recibe el mismo nombre del servicio que lo utiliza. Así, para nuestro caso en particular, se utiliza el mensaje denominado `consDir&PI()`, pero esto varía de servicio en servicio ya que el tipo de dato esperado por el `GeoCoder` difiere de uno en otro. Por esto es que no se puede pensar en tener el mismo mensaje para todos los servicios, ya que el tipo esperado varía según el servicio. Además, la cantidad de parámetros también varía, esperando a veces un solo, y a veces dos parámetros.
4. Luego, esta respuesta del `GeoCoder` conteniendo la ubicación de la dirección o del punto de interés, es enviada al componente `GeoParser` a través de su interfaz para obtener una respuesta adecuada para ser devuelta al usuario. Este paso es el inverso al hecho en el punto 2, en donde mediante el mensaje `parsearConsDir&PI()` se tomaban los datos relevantes a cada servicio. Ahora, a partir de los datos relevantes del servicio, se construye una respuesta en lenguaje español que será luego devuelta al usuario, previa conversión a voz mediante. En esta etapa, se construye una respuesta en formato texto. El nombre del mensaje es siempre el mismo (`convertir()`), independientemente del tipo de servicio. Esta independencia se logra gracias a que cada vez que el mensaje es utilizado, se debe pasar como parámetro el tipo de servicio en que se encuentra. En este caso en particular, además de la respuesta conteniendo la ubicación de la dirección o del PI, se envía también como parámetro al enumerado `ConsDir&PI`. Es deseable que este mensaje siempre espere recibir como parámetro algún valor dentro de una enumeración de valores, entre los que se encuentran `ServCercano`, `PagsAmarillas`, `Ruteo`, `PublicidadMovil`, etc.
5. Finalmente, el componente `Speech` a través de su interfaz (`ISpeech`) convierte la respuesta devuelta del `GeoParser` en voz. Para esto se utiliza el mensaje llamado `texto_a_voz()`, quien recibe como parámetro la respuesta devuelta por el `GeoParser` que siempre será de tipo texto, y genera una respuesta audible para ser devuelta al usuario. Dado que siempre se recibe un texto y se devuelve un tipo `Voz`, el mensaje es siempre el mismo para todos los diferentes servicios ofrecidos por el sistema. A su vez, este paso es inverso al primero, en donde se utilizaba el mensaje `voz_a_texto()` para convertir en texto lo hablado por el usuario.

Como observación final acerca de este Diagrama de Colaboración: se puede ver que en todo momento, los mensajes (operaciones en la interfaz correspondiente) intercambian un solo dato, una sola respuesta es obtenida en cada momento.

Esto contrasta con lo que ocurrirá en futuros diagramas, en los cuales una invocación a un mensaje (operación) retornará un conjunto de valores, los cuales serán tomados uno a uno y procesados según corresponda.

5.1.4 Diagrama de Clases de Diseño

La idea de un Diagrama de Clases es la de describir gráficamente las especificaciones de las clases de software y de las interfaces de un sistema.

Normalmente contiene:

- Clases, asociaciones y atributos
- Interfaces y sus operaciones
- Tipos de los atributos
- Navegabilidad
- Dependencias

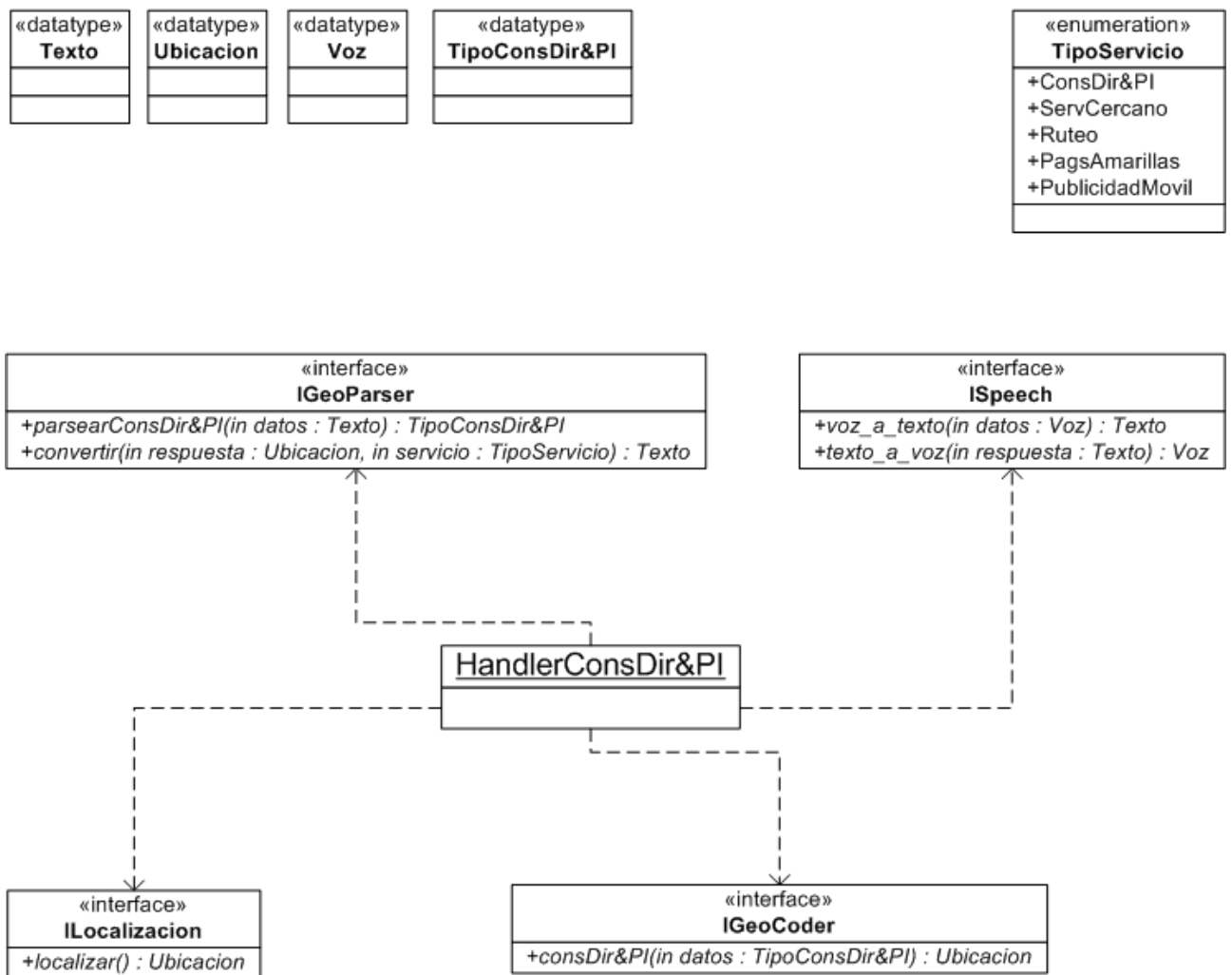
La principal diferencia con el Modelo Conceptual, es que un Diagrama de Clases contiene las definiciones de las entidades de software en lugar de conceptos del mundo real. Esto quiere decir que en este tipo de diagrama aparecen las especificaciones de las clases, tipos de datos e interfaces de software por primera vez.

En nuestro caso en particular, por tratarse de un Análisis & Diseño general, es que estos diagramas contendrán mas interfaces que especifican un determinado comportamiento que clases particulares. En otros proyectos donde se dispone de información más precisa se podrán ver más clases en estos diagramas.

Para confeccionar un Diagrama de Clases de Diseño, se deben tomar los mensajes que se llama sobre las interfaces definidas en los Diagramas de Colaboración, haciendo de cada mensaje una operación de tal interfaz. Los tipos de datos utilizados son ahora definidos.

Claramente, este tipo de artefactos forma parte del Diseño del sistema ya que involucra una toma de decisión en cuanto a que responsabilidades se asigna a cada objeto de software, al igual que en los Diagramas de Colaboración. La diferencia es que los primeros sirven como punto de partida para la codificación, es decir, la implementación en un cierto lenguaje de programación orientado a objetos.

A continuación presentamos el diagrama correspondiente al Caso de Uso actual.



Como se puede ver, cada interfaz del Diagrama de Colaboración es ahora un elemento `<<interface>>` en el Diagrama de Clases. Cada mensaje que era llamado a ejecutar la interfaz, es ahora una operación dentro de su correspondiente elemento de interfaz. A su vez, se han definido de forma concreta los tipos de datos utilizados, como ser los `<<datatype>>` `Texto`, `Ubicacion` y `Voz`.

También se ha definido un tipo enumerado (`<<enumeration>>`) llamado `TipoServicio` que contiene los diferentes tipos de servicios (o Casos de Uso) definidos en el sistema. Estos son utilizados cada vez que se invoca a la operación `convertir()` de `IGeoParser`, cada vez con su valor correspondiente.

Un tipo de dato que también se definió, es `TipoConsDir&PI`. Dentro de este tipo, el programador (o implementador) del sistema, es decir quien escribirá el código a partir de estos diagramas, debe decidir que datos particulares deberá tener este tipo de datos. Seguramente la decisión incluirá colocar un dato que contenga el nombre de la calle buscada, o su intersección con otra, o un punto de interés, o lugar conocido. Tal decisión se deja en manos del programador.

La idea es que luego de contar con todos los Diagramas de Clases de Diseño de todos los Casos de Uso, éstos sean “juntados” en uno solo. Ya que cada diagrama contiene las operaciones, tipos, etc. que ese Caso de Uso necesita, la combinación de todos los diagramas contendrá la información que el Sistema en forma global necesita. Seguramente habrán muchos elementos repetidos en los diferentes diagramas (como es el caso del tipo de dato `Ubicacion`), pero lo interesante de esta unión es que los diagramas se complementarán unos con otros, dando cada uno su visión de lo que necesita tener el sistema.

Un caso típico de complementación de interfaces se dará con `IGeoParser`. Esta interfaz define una operación diferente por cada Caso de Uso. Así, `Servicios Cercanos` necesita de `parsearServCercano()`, `Ruteo` necesita de `parsearRuteo()`, y sucesivamente. Por tanto, la `IGeoParser` final contará con todas estas operaciones, cada una dando soporte al Caso de Uso particular, y todas dando soporte al sistema.

Así, si se dota al sistema de todos los tipos de datos, enumerados, operaciones, interfaces, clases y atributos que cada Caso de Uso necesita, se contará con un diseño general del sistema en su conjunto.

Tal diagrama se confeccionará en la última Sección del presente Capítulo.

5.2 Servicios Cercanos

5.2.1 Caso de Uso Expandido

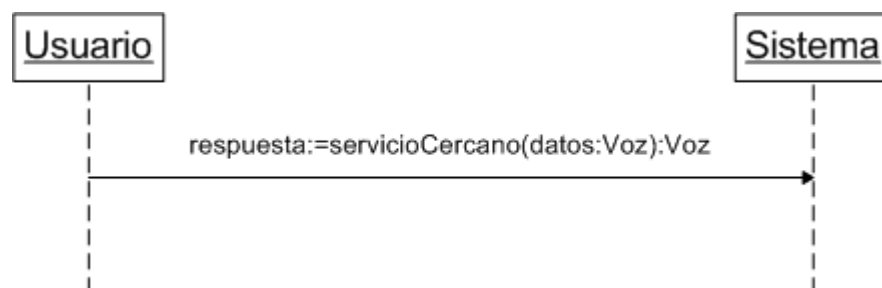
CU Expandido:	Servicios Cercanos.
Actor:	Usuario (iniciador).
Propósito:	Conjunto de comercios y servicios cercanos al usuario.
Resumen:	Permite ubicar los comercios (productos y servicios) más cercanos a una dirección ingresada por el usuario. Se incluyen las siguientes categorías de información: restaurantes, fast-food, hoteles, bancos y entidades financieras, agencias de pago, farmacias y entidades de salud (hospitales, sanatorios, clínicas), educación (universidades, escuelas y colegios), agencias de gobierno, entre muchos otros datos disponibles. El sistema retorna una lista de los N sitios más cercanos y opcionalmente con la información detallada de cada sitio: nombre, descripción, dirección, teléfono, etc.
Tipo:	Primario y esencial.

Curso Normal de los Eventos:

Acción del Actor	Respuesta del Sistema
1. Se conecta al sistema	2. Se presentan los tipos de servicios en forma hablada
3. Inicia Servicios Cercanos	4. Toma la ubicación del usuario
6. Ingresar el rubro, producto o servicio buscado y la distancia máxima a la cual lo desea	5. Pregunta sobre qué Servicio Cercano se desea efectuar la búsqueda y la distancia
	7. Se toman los datos
	8. Se calculan los servicios más cercanos a la ubicación dada, haciendo una lista en caso de haber más de uno
	9. Se devuelve el resultado en forma hablada

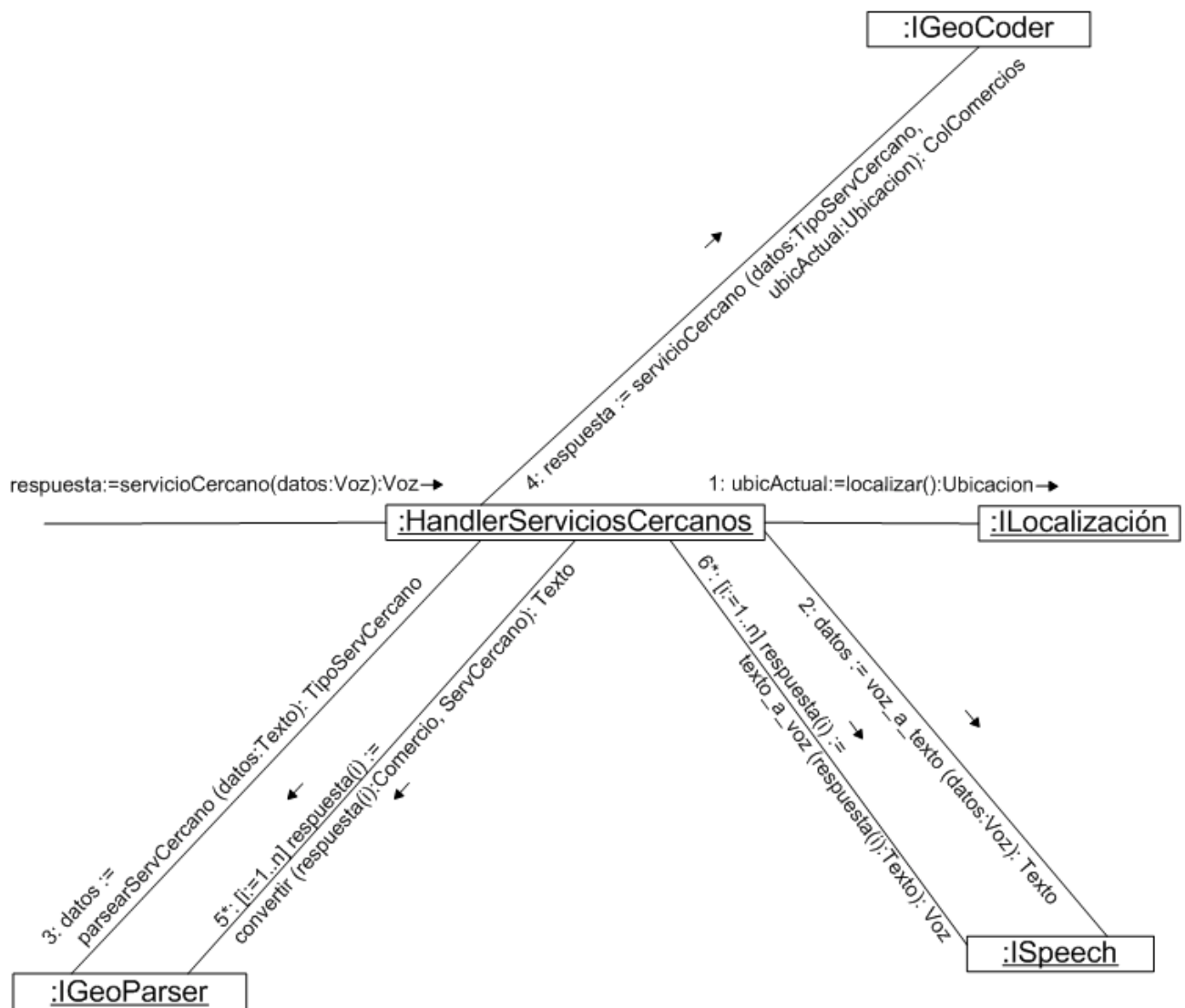
5.2.2 Diagrama de Secuencia

El Diagrama de Secuencia correspondiente con el Caso de Uso Servicios Cercanos muestra el evento `servicioCercano()`. Éste toma los datos del usuario en formato voz conteniendo lo que éste dijo, y devuelve una respuesta también en formato voz con el resultado de la búsqueda de comercios o servicios cercanos a la ubicación actual del usuario.



5.2.3 Diagrama de Colaboración

Aquí se presenta el Diagrama de Colaboración correspondiente al Caso de Uso de Servicios Cercanos. Como se dijo en la Sección 5.1.3, este artefacto del UML pretende mostrar las colaboraciones necesarias entre los objetos de software para llevar a cabo el evento con el cual se comienza y termina el diagrama.



Lo que sigue es la explicación de tales colaboraciones. Como ya se dijo antes, algunos conceptos fundamentales introducidos en el diagrama de la Sección 5.1.3 no serán explicados de nuevo.

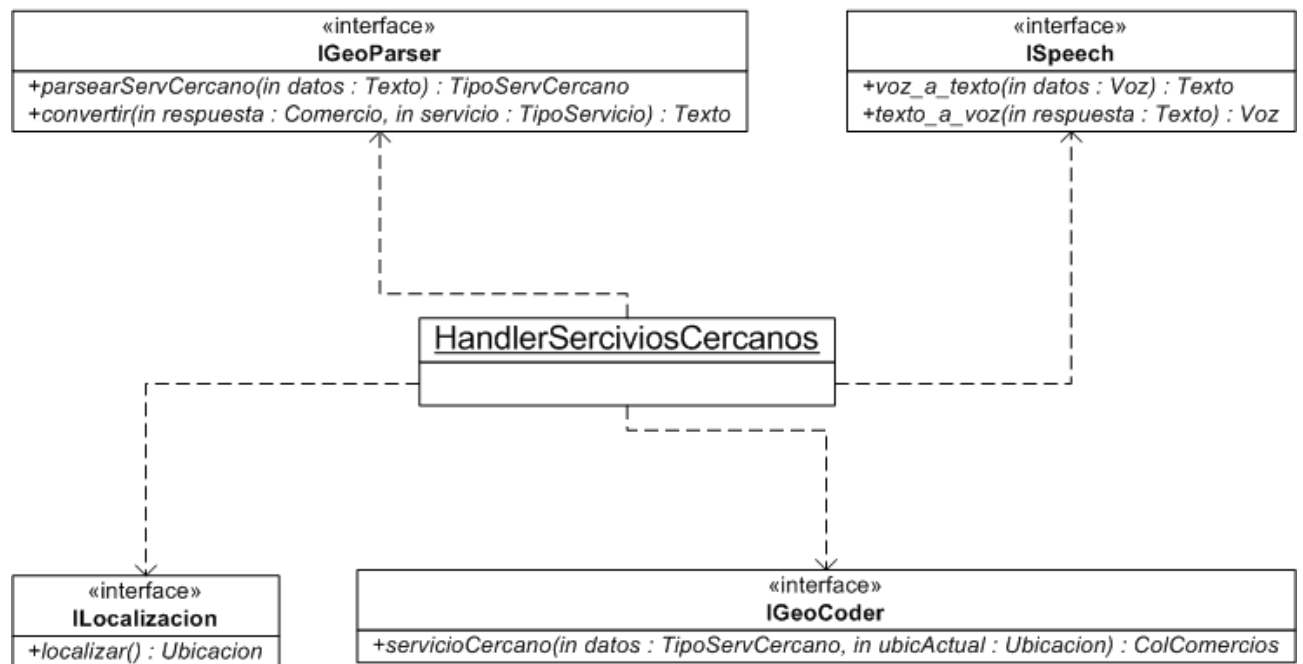
0. Como todo Diagrama de Colaboración, comienza con el evento obtenido del Diagrama de Secuencia del Sistema. En nuestro caso particular se trata del evento `servicioCercano()` recién comentado.
1. El primer paso para poder brindar este servicio, consta de ubicar al usuario. Es decir, obtener la ubicación “exacta” del usuario en un mapa en ese momento. Para esto se llama a la operación `localizar()` de la interfaz `ILocalización`. Esta interfaz es nueva, ya que no estaba presente en el diagrama anterior de colaboraciones. Este mensaje no recibe parámetro alguno, ya que lo único que hace es devolver la localización actual del usuario. Ésta, llamada `ubicActual` se considera de tipo `Ubicacion`.
2. El segundo mensaje es enviado a la interfaz del habla, `ISpeech`, pidiéndole que convierta a texto los datos hablados por el usuario. Al igual que en el diagrama del Caso de Uso anterior, esto se hace mediante el mensaje `voz_a_texto()` quien toma los datos hablados y los convierte a texto.

3. Inmediatamente después de haber convertido a formato texto los datos del usuario, éstos son pasados al componente `GeoParser` a través de la interfaz `IGeoParser` y su operación `parsearServCercano()`, que toma los datos textuales y devuelve un tipo estructurado (`TipoServCercano`) conteniendo los tipos de comercios (productos y servicios) que el usuario desea ubicar en su cercanía, y la distancia a la cual los prefiere. Notar que, como se dijo antes, existe una operación de parseo diferente para cada servicio.
4. Teniendo los datos útiles para el servicio cercano (rubro de los comercios buscados, distancia reunidos en el tipo de dato `TipoServicioCercano`), junto con la ubicación actual del usuario (`ubicActual`), es que se está en condiciones de invocar al mensaje `servicioCercano()` de la interfaz `IGeoCoder`. Este mensaje recibe los mencionados datos como entrada y devuelve, esta vez, una colección de elementos. ¿Elementos de que tipo? De tipo `Comercio`, a los que juntos se les denominó `ColComercios` (por Colección de Comercios).
5. Este punto reviste una particular importancia, o, mejor dicho, una peculiaridad nueva que introduce este Diagrama de Colaboración. Dado que el paso 4 devuelve una colección de comercios (`ColComercios`), lo que se debe hacer en este punto es convertir cada resultado devuelto en un texto pasible de ser “leído” por el sistema al usuario. Es decir, tomar el resultado devuelto por el `GeoCoder`, convertirlo en texto (ya no en comercios) y darle cierto formato o “cosmética” para ser pasado luego a la interfaz de habla y ser traducido a voz. Esta idea de tomar los comercios devueltos uno a uno, suponiendo para esto que son n-comercios (de 1 a n), se representa mediante el control de iteración definido para este mensaje (`i := [1..n]`). Además, se le agrega al número de mensaje el carácter asterisco (*), dando la idea de iteración, es decir que el mensaje es ejecutado consecutivamente. A su vez, al mensaje que se invoca (`convertir()`), se le pasan los comercios, uno a uno (`respuesta(i)`), más el indicador del servicio que se está utilizando (`ServCercano`) que es quien permite dar un formato adecuado a la salida de texto. Puede pensarse que para el caso de Servicios Cercanos el sistema forma un texto del estilo “Los comercios de tipo XXX a YYY metros de su ubicación son los siguientes: ...”, mientras que para otros servicios el mensaje generado será distinto. Por esto es que se utiliza el indicador de servicio `ServCercano`.
6. El último paso, como siempre ocurre, consiste en traducir las respuestas del `GeoParser` en voz. La diferencia en este caso particular es que esto ocurrirá tantas veces como comercios haya arrojado la búsqueda del `GeoCoder`. Por ello es que este mensaje es invocado iterativamente, y por esto es que se utilizó la sintaxis apropiada, ya comentada en el paso anterior.

5.2.4 Diagrama de Clases de Diseño

A continuación se presenta el Diagrama de Clases de Diseño para Servicios Cercanos.

Se obvian las explicaciones ya que se aplican las mismas que para el diagrama del Caso de Uso anterior.



5.3 Ruteo

5.3.1 Caso de Uso Expandido

CU Expandido:	Ruteo.
Actor:	Usuario (iniciador).
Propósito:	Solución de ruteamiento entre dos ubicaciones.
Resumen:	<p>Este servicio permite el cálculo de una ruta entre dos ubicaciones dadas por el usuario. En este tipo de problemas es de extrema importancia la velocidad con la cual se calculan las rutas entre dos puntos dados de un mapa ya sea a pie, en auto, en bus, etc. Al momento de calcular rutas, los usuarios tienen diferentes requerimientos, como ser: la ruta más corta, la que tenga menos tránsito, la más rápida, la que tenga menos peajes, etc.</p> <p>El ruteo no siempre tiene como objetivo una ruta en particular, sino que a veces se desea un estimativo del tiempo que lleva recorrer dicha ruta. Para lograr esto, se debe tener en cuenta el tráfico estimado sobre las calles consideradas, tarea nada trivial.</p> <p>Para recorridos con vehículos considera todas las restricciones de tránsito disponibles, como sentidos de circulación, giros permitidos, pasos a nivel, rampas de acceso en autopistas y calles inhabilitadas entre otras.</p>
Tipo:	Primario y esencial.

Curso Normal de los Eventos:

Acción del Actor	Respuesta del Sistema
1. Se conecta al sistema	2. Se presentan los tipos de servicios en forma hablada
3. Inicia Ruteo	4. Pregunta la ubicación de destino
5. Se ingresa la ubicación de destino	6. Se toman los datos
	7. Pregunta por características (preferencias) sobre el ruteo a obtener
8. Introduce su preferencia: menor tiempo, menor combustible, menor recorrido, etc.	9. Se toman los datos
	10. Se calcula la ruta tomando en cuenta las preferencias dadas
	11. Se devuelve el resultado en forma hablada

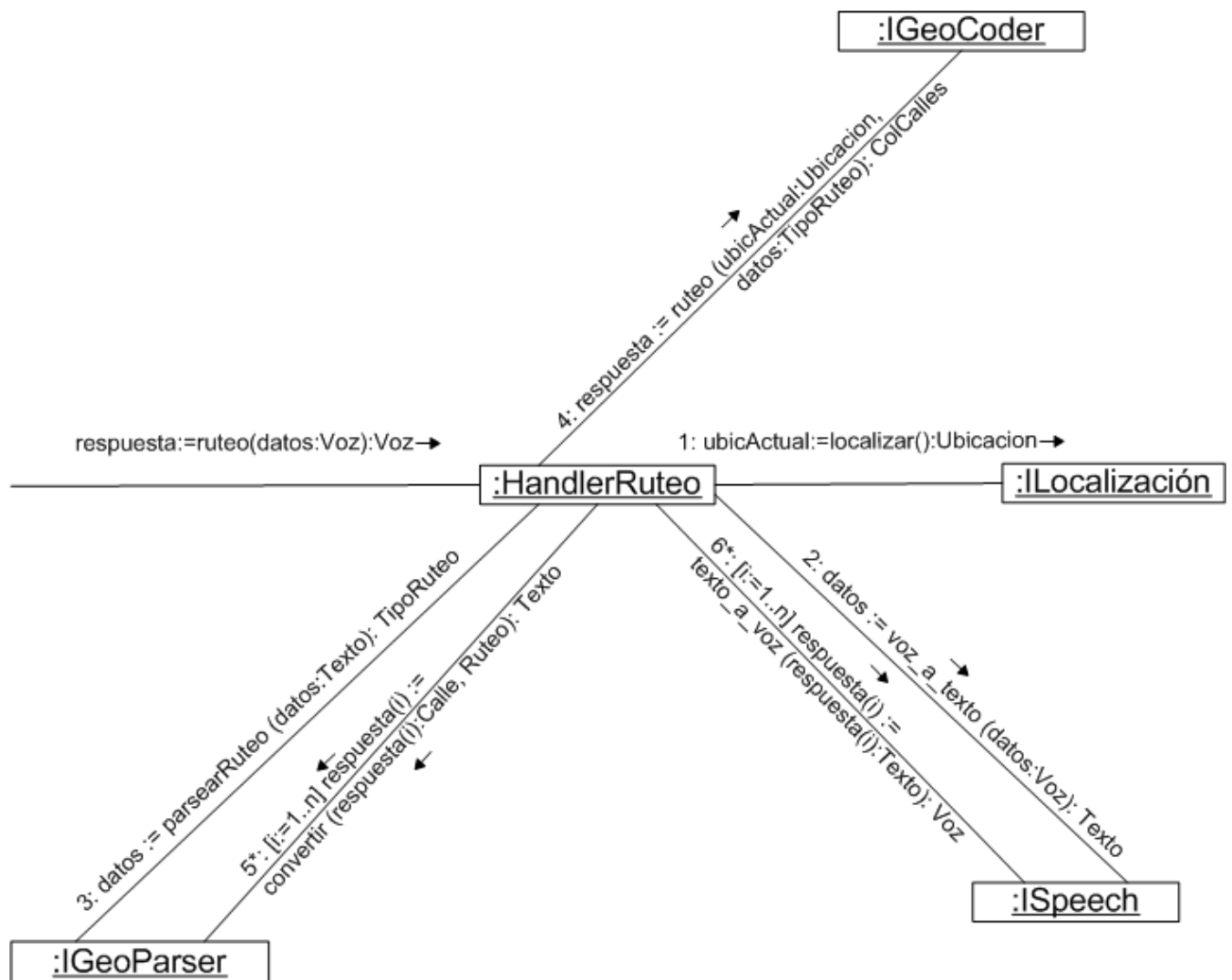
5.3.2 Diagrama de Secuencia

El Diagrama de Secuencia correspondiente con el Caso de Uso Ruteo muestra el evento `ruteo()`. Éste toma los datos del usuario en formato voz conteniendo lo que éste dijo, y devuelve una respuesta también en formato voz.



5.3.3 Diagrama de Colaboración

Aquí se presenta el Diagrama de Colaboración correspondiente al Caso de Uso Ruteo. Como se dijo en la Sección 5.1.3, este artefacto del UML pretende mostrar las colaboraciones necesarias entre los objetos de software para llevar a cabo el evento con el cual se comienza y termina el diagrama.

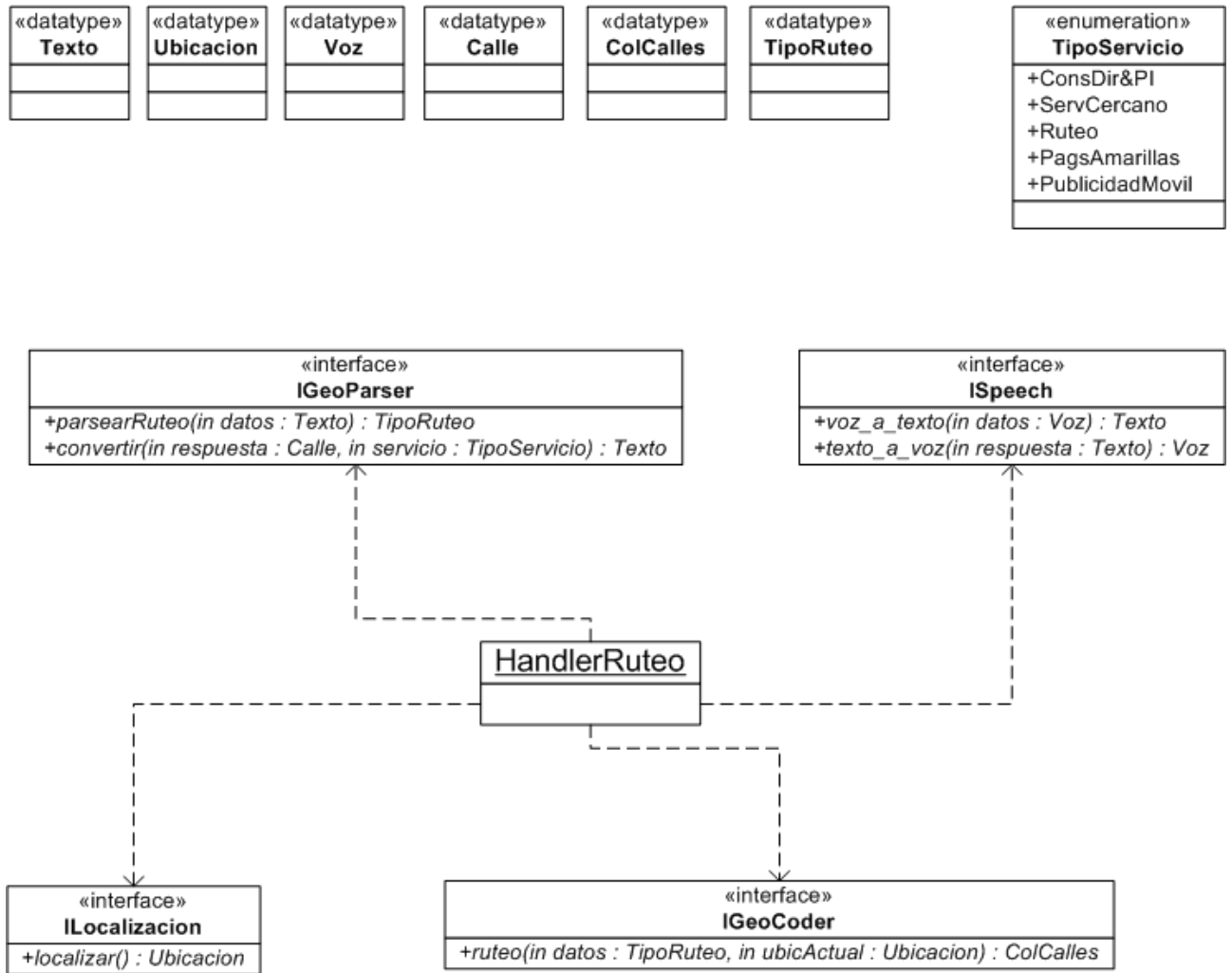


Lo que sigue es la explicación de tales colaboraciones. Como ya se dijo antes, algunos conceptos fundamentales introducidos en el diagrama de la Sección 5.1.3 no serán explicados de nuevo.

0. Como todo Diagrama de Colaboración, comienza con el evento obtenido del Diagrama de Secuencia del Sistema. En nuestro caso particular se trata del evento `ruteo()` recién comentado.
1. El primer paso para poder brindar este servicio, consta de ubicar al usuario. Es decir, obtener la ubicación “exacta” del usuario en un mapa en ese momento. Para esto se llama al mensaje `localizar()` de la interfaz `ILocalizacion`. Esta interfaz es nueva, ya que no estaba presente en el diagrama anterior de colaboraciones. Este mensaje no recibe parámetro alguno, ya que lo único que hace es devolver la localización actual del usuario. Ésta, llamada `ubicActual` se considera de tipo `Ubicacion`.
2. El segundo mensaje es enviado a la interfaz del habla, `ISpeech`, pidiéndole que convierta a texto los datos hablados por el usuario. Al igual que en el diagrama del Caso de Uso anterior, esto se hace mediante el mensaje `voz_a_texto()` quien toma los datos hablados y los convierte a texto.
3. Inmediatamente después de haber convertido a formato texto los datos del usuario, éstos son pasados al `GeoParser` a través de la interfaz `IGeoParser` y su operación `parsearRuteo()`, que toma los datos textuales y devuelve un tipo estructurado (`TipoRuteo`) conteniendo la ubicación de destino de la ruta y la función de costos o cualquier otro dato relevante al cálculo de la ruta.
4. Teniendo los datos útiles para el ruteo, junto con la ubicación actual del usuario (`ubicActual`), es que se está en condiciones de invocar al mensaje `ruteo()` de la interfaz `IGeoCoder`. Este mensaje recibe los mencionados datos como entrada y devuelve, esta vez, una colección de elementos. ¿Elementos de que tipo? De tipo `Calle`, a los que juntos se les denominó `ColCalles`. ¿Porqué se devuelve un conjunto de calles? Porque el servicio de ruteo devuelve una colección de calles como respuesta al usuario, indicándole, desde la ubicación actual, que calles tomar para llegar a destino. Esta es al menos una posible respuesta del sistema, aunque podría pensarse en otras.
5. Dado que el paso 4 devuelve una colección de calles, lo que se debe hacer en este punto es convertir cada resultado devuelto en un texto pasible de ser “leído” por el sistema al usuario. Es decir, tomar el resultado devuelto por el `GeoCoder`, convertirlo en texto (ya no en calles) y darle cierto formato o “cosmética” para ser pasado luego a la interfaz de habla y ser traducido a voz. Es el mismo comportamiento visto para el Caso de Uso anterior. Al mensaje que se invoca (`convertir()`), se le pasan las calles una a una (`respuesta(i)`), más el indicador del servicio que se esta utilizando (`Ruteo`) que es quien permite dar un formato adecuado a la salida de texto.
6. El último paso, como siempre ocurre, consiste en traducir las respuestas del `GeoParser` en voz.

5.3.4 Diagrama de Clases de Diseño

A continuación se presenta el Diagrama de Clases de Diseño para Ruteo.



5.4 Páginas Amarillas

5.4.1 Caso de Uso Expandido

CU Expandido:	Páginas Amarillas.
Actor:	Usuario (iniciador).
Propósito:	Conjunto de comercios y servicios con cierto criterio.
Resumen:	El usuario consulta los comercios de su interés ingresando por ejemplo el rubro del mismo o parte de su nombre. Esto dará como resultado un listado de comercios que poseen la característica ingresada en común, por lo cual el usuario deberá decidir por una de estas opciones. Cabe desatacar que el sistema devolverá una lista hablada con todos los comercios o servicios que cumplan los criterios, y en caso de ser necesario una descripción de alguno de ellos. En resumen, este servicio es una “guía de páginas amarilla” en formato digital. <u>No</u> debe confundirse con Servicios Cercanos, quien permite al usuario ubicar comercios de cierto rubro cerca de su ubicación, y no por cierto criterio, como es ahora el caso. Otra diferencia esencial es que este servicio no utiliza en absoluto la ubicación actual del usuario (ver Diagrama de Colaboración más adelante).
Tipo:	Primario y esencial.

Curso Normal de los Eventos:

Acción del Actor	Respuesta del Sistema
1. Se conecta al sistema	2. Se presentan los tipos de servicios en forma hablada
3. Inicia Páginas Amarillas	4. Pregunta criterios de búsqueda
5. Se ingresa/n el/los criterio/s de búsqueda	6. Se toman los datos
	7. Se calculan los comercios y servicios que se correspondan con los criterios ingresados, haciendo una lista en caso de haber más de uno
	8. Se devuelve el resultado en forma hablada

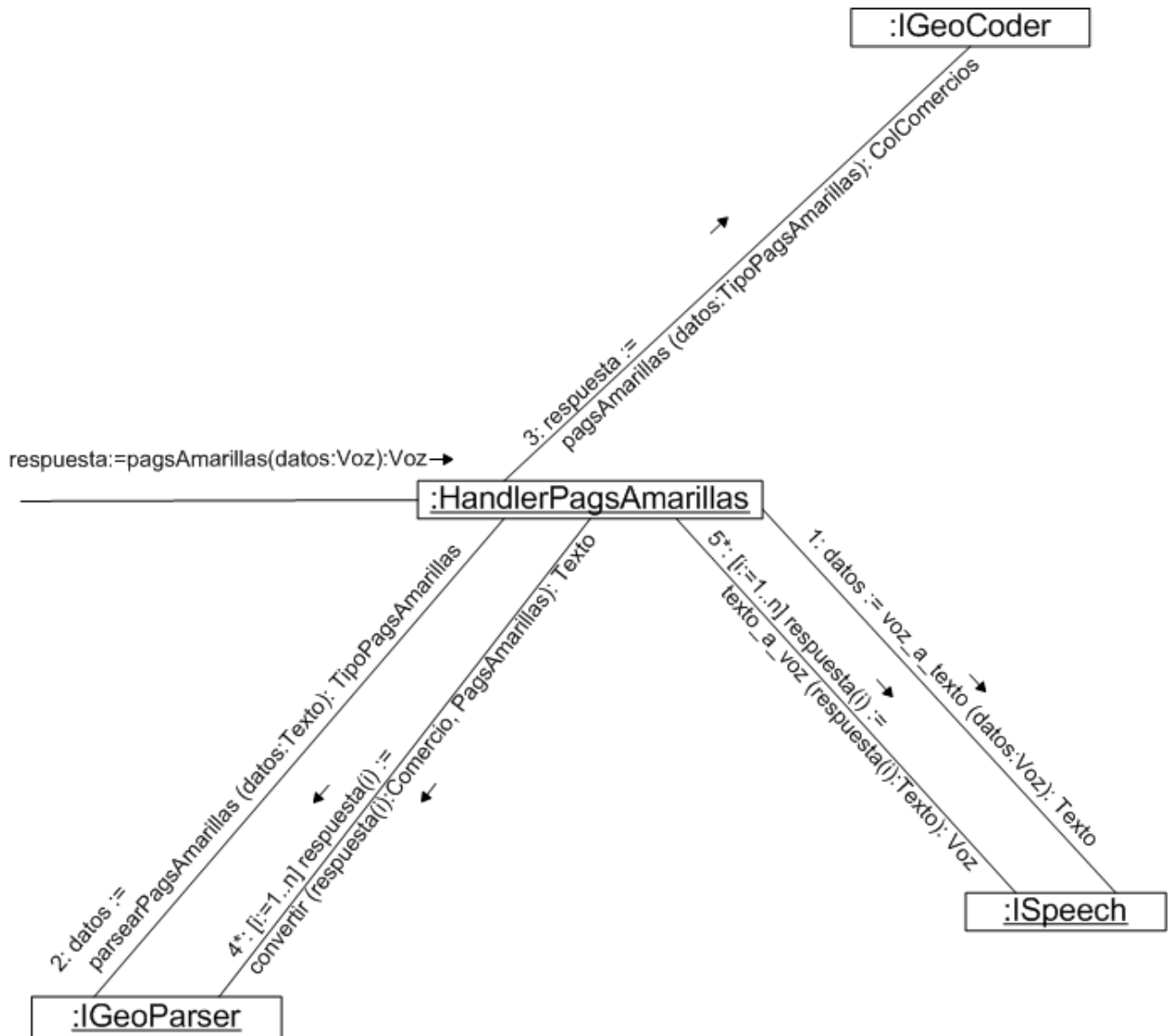
5.4.2 Diagrama de Secuencia

El Diagrama de Secuencia correspondiente con el Caso de Uso Páginas Amarillas muestra el evento `pagsAmarillas()`. Éste toma los datos del usuario en formato voz conteniendo lo que éste dijo, y devuelve una respuesta también en formato voz.



5.4.3 Diagrama de Colaboración

Aquí se presenta el Diagrama de Colaboración correspondiente al Caso de Uso Páginas Amarillas. Como se dijo en la Sección 5.1.3, este artefacto del UML pretende mostrar las colaboraciones necesarias entre los objetos de software para llevar a cabo el evento con el cual se comienza y termina el diagrama.



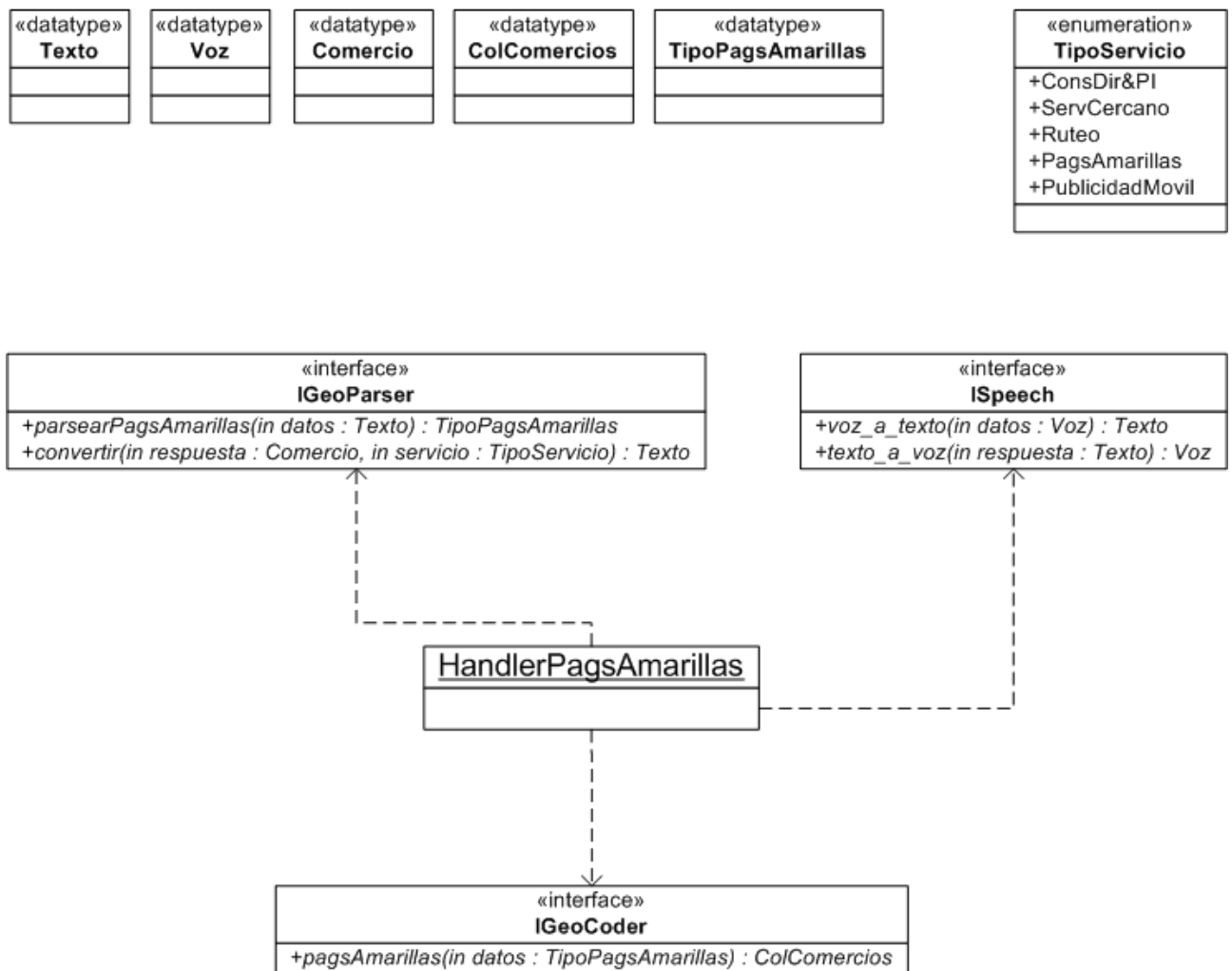
Lo que sigue es la explicación de tales colaboraciones. Como ya se dijo antes, algunos conceptos fundamentales introducidos en el diagrama de la Sección 5.1.3 no serán explicados de nuevo.

0. Como todo Diagrama de Colaboración, comienza con el evento obtenido del Diagrama de Secuencia del Sistema. En nuestro caso particular se trata del evento `pagsAmarillas()` recién comentado.
1. El primer paso para poder brindar este servicio difiere de los últimos dos vistos y es similar al del primer Diagrama de Colaboración analizado para el Caso de Uso Consulta de Direcciones y Puntos de Interés. El primer mensaje es enviado directamente a la interfaz del habla, `ISpeech`, pidiéndole a su componente (`Speech`) que convierta a texto los datos hablados por el usuario, mediante la operación ya conocida.
2. Después de haber convertido a formato texto los datos del usuario, éstos son pasados al `GeoParser`, que toma los datos textuales y devuelve un tipo estructurado (`TipoPagsAmarillas`) conteniendo el criterio de búsqueda de comercios, por ejemplo, aquellos ubicados en cierta calle, aquellos que contienen cierta palabra en su nombre, o dentro de cierto rubro, etc. El tipo particular de búsqueda es totalmente personalizable, y pueden pensarse muchas opciones diferentes.

3. El único parámetro necesario ahora para invocar al mensaje `pagsAmarillas()` de la interfaz `IGeoCoder`, es el recién comentado tipo estructurado. Este mensaje devuelve una colección de comercios que coincidan con el criterio de búsqueda ingresado por el usuario.
4. Dado que el paso 3 devuelve una colección de calles, lo que se debe hacer en este punto es convertir cada resultado devuelto en un texto pasible de ser “leído” por el sistema al usuario. Es el mismo comportamiento visto para el Caso de Uso anterior.
5. El último paso, como siempre ocurre, consiste en traducir las respuestas del `GeoParser` en voz.

5.4.4 Diagrama de Clases de Diseño

A continuación se presenta el Diagrama de Clases de Diseño para Páginas Amarillas.



5.5 Publicidad Móvil

5.5.1 Caso de Uso Expandido

CU Expandido:	Publicidad Móvil.
Actor:	Usuario.
Propósito:	Enviar publicidad a un usuario móvil basado en su ubicación y en su perfil de preferencias.
Resumen:	<p>Este servicio que forma parte de la tercera generación de servicios sensibles a la ubicación, tiene la capacidad de iniciar el servicio proactivamente sin que el usuario realice una petición explícita. Este tipo de servicio se dice que funciona en modo “gatillo”. Un “gatillo” es una condición que nace de una ubicación geográfica del dispositivo. Luego, este “gatillo” inicia automáticamente servicios y notificaciones.</p> <p>Es decir, que cuando el usuario se encuentra a determinada distancia de un lugar que brinda el servicio de publicidad móvil, éste es notificado automáticamente mediante publicidad.</p> <p>Los datos a publicitar dependerán de un perfil previamente creado para el usuario con sus preferencias, gustos y costumbres personales para así ajustar lo más posible el target de la publicidad.</p>
Tipo:	Primario y esencial.

Curso Normal de los Eventos:

Acción del Actor	Respuesta del Sistema
1. Previamente crea en el Sistema un “perfil” con sus preferencias de comprar y sus costumbres	
2. Debe estar en estado “disponible” para poder recibir publicidad	
	3. Ubica al usuario con respecto a la vecindad que lo rodea (que es de donde surge la publicidad)
	4. En caso de encontrarse alguna empresa que brinde el servicio y que coincida con su perfil, la misma es ingresada un una lista, la cual será enviada al usuario
	5. Se devuelve el resultado en forma hablada

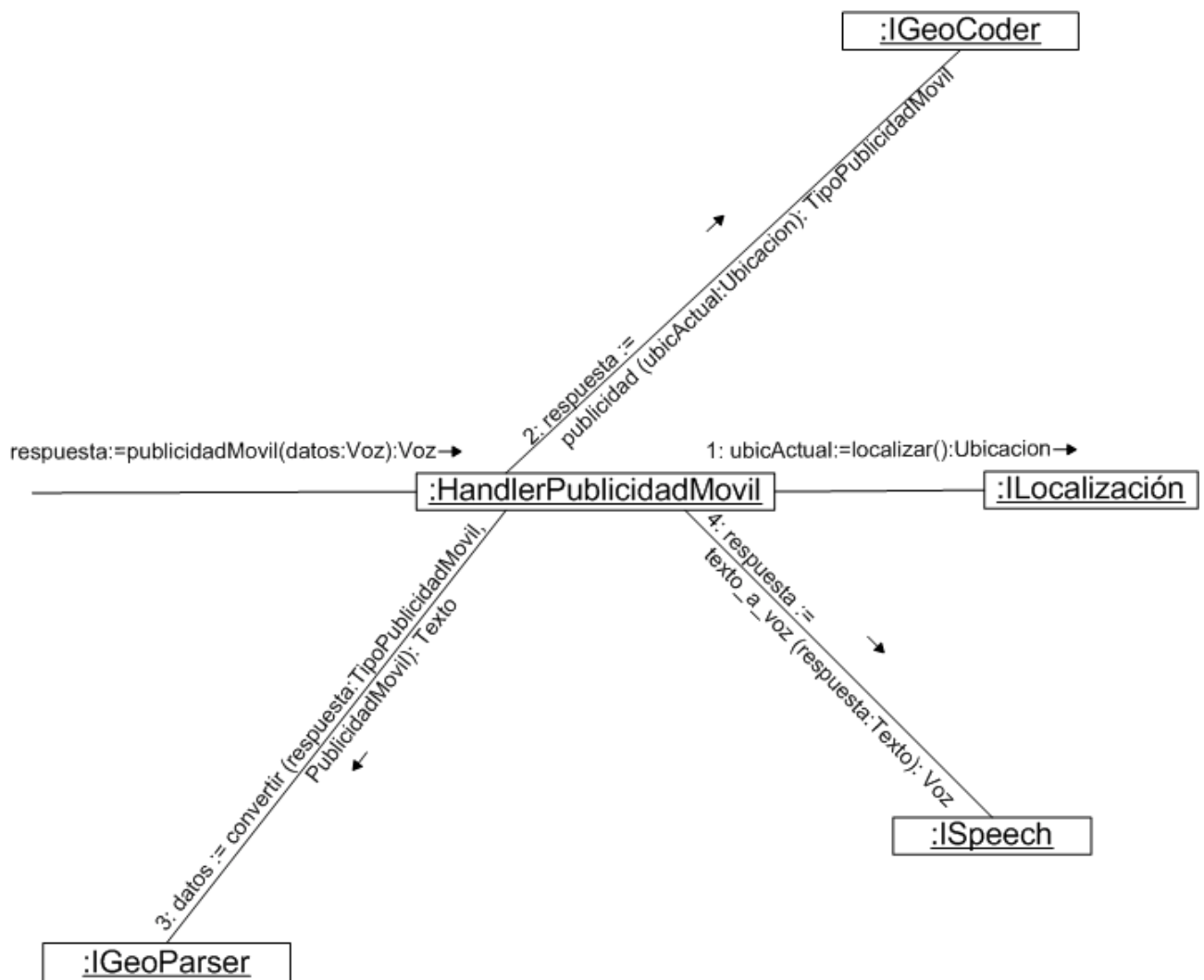
5.5.2 Diagrama de Secuencia

El Diagrama de Secuencia correspondiente con el Caso de Uso Publicidad Móvil muestra el evento `publicidadMovil()`. Éste toma los datos del usuario en formato voz conteniendo lo que éste dijo, y devuelve una respuesta también en formato voz.



5.5.3 Diagrama de Colaboración

Aquí se presenta el Diagrama de Colaboración correspondiente al Caso de Uso Publicidad Móvil. Como se dijo en la Sección 5.1.3, este artefacto del UML pretende mostrar las colaboraciones necesarias entre los objetos de software para llevar a cabo el evento con el cual se comienza y termina el diagrama.



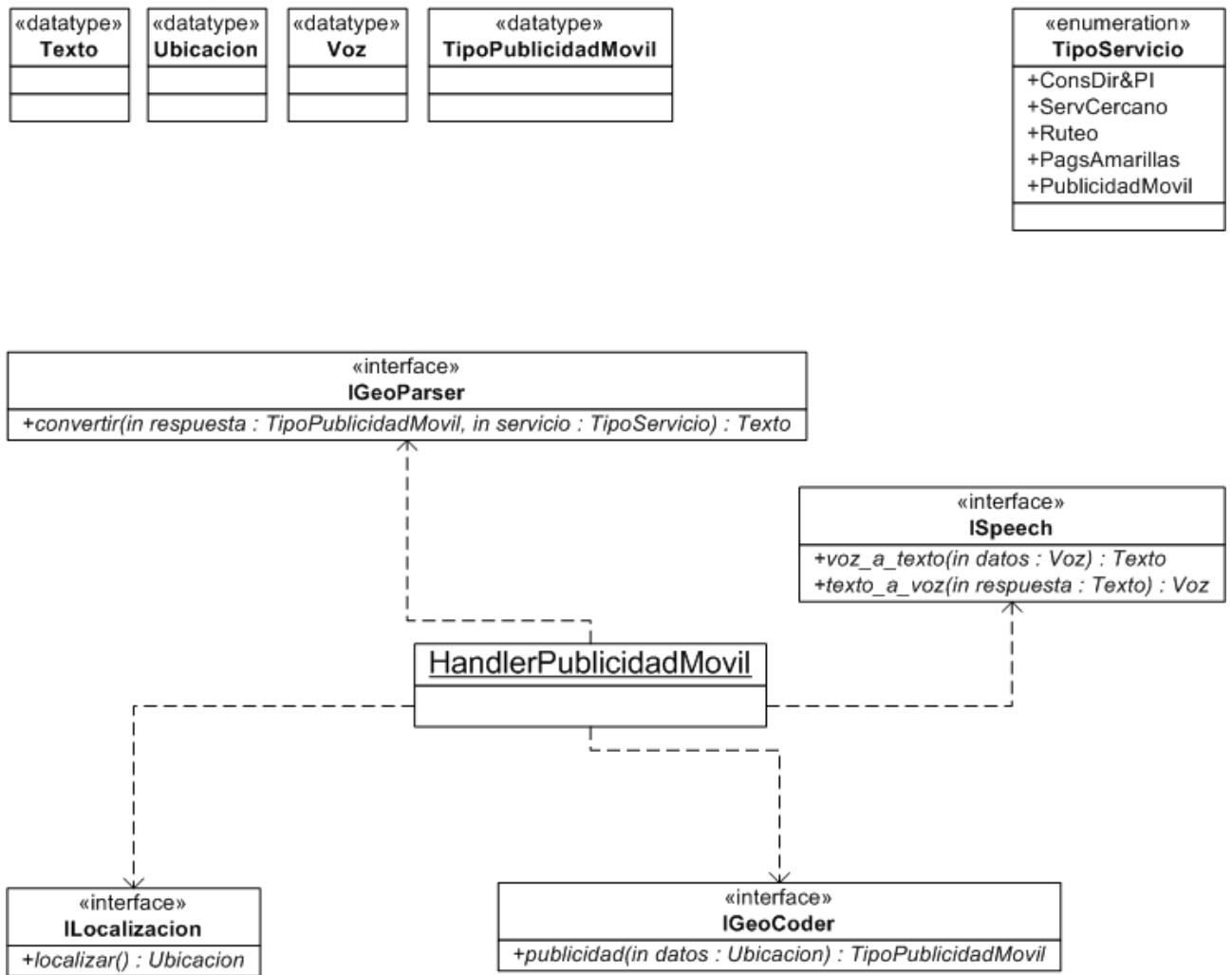
Lo que sigue es la explicación de tales colaboraciones. Como ya se dijo antes, algunos conceptos fundamentales introducidos en el diagrama de la Sección 5.1.3 no serán explicados de nuevo.

0. Como todo Diagrama de Colaboración, comienza con el evento obtenido del Diagrama de Secuencia del Sistema. En nuestro caso particular se trata del evento `publicidadMovil()` recién comentado.
1. El primer paso consta de ubicar al usuario, de forma análoga a lo ya visto. Este paso es particularmente importante en este servicio, ya que la publicidad dependerá de la ubicación actual del usuario, es decir, por dónde se está desplazando en todo momento. Por esto es que se dice que la publicidad es sensible (y personalizada) a la ubicación de éste.
2. En este caso, el segundo mensaje aparece en forma sorpresiva y apunta directamente al GeoCoder, solamente con la información de la ubicación. Esto es así, y es correcto, ya que para poder enviarle la publicidad verdaderamente sensible al lugar, el GeoCoder debe solo saber en que lugar está el usuario. Por eso tal parámetro. Obviamente, no hay que olvidar que el propio GeoCoder debe contar con las tablas de preferencias de los usuarios, indicando si el comercio actual que esta cerca, se corresponde con los gustos del usuario o no. Solo en caso de encontrarse cerca de un comercio de su preferencia, es que se lanzará un mensaje publicitario.

3. Lo devuelto por el mensaje anterior, se define como algo de tipo `TipoPublicidadMovil`. Podría pensarse en devolver solamente un texto con la publicidad deseada (en caso de que corresponda, como se vio en el punto anterior). Para diseñar una solución mas general, se decidió trabajar con un tipo estructurado especial, el cual seguramente, y por lo menos, contendrá el mensaje publicitario a devolver. Pero, podría pensarse también, de que el `GeoCoder` devuelva la dirección exacta de dicho comercio, ya que este módulo la conoce perfectamente, pero el usuario no. Aquí se hace evidente la necesidad de un tipo estructurado, ya que sino tendría que ser el propio `GeoCoder` el que concatene el texto publicitario con la dirección convertida a texto. Claramente esta no es tarea del `GeoCoder`, pero si del `GeoParser`.
4. El último paso, como siempre ocurre, consiste en traducir las respuestas del `GeoParser` en voz.

5.5.4 Diagrama de Clases de Diseño

A continuación se presenta el Diagrama de Clases de Diseño para Publicidad Móvil.



5.6 Diagrama General de Clases de Diseño

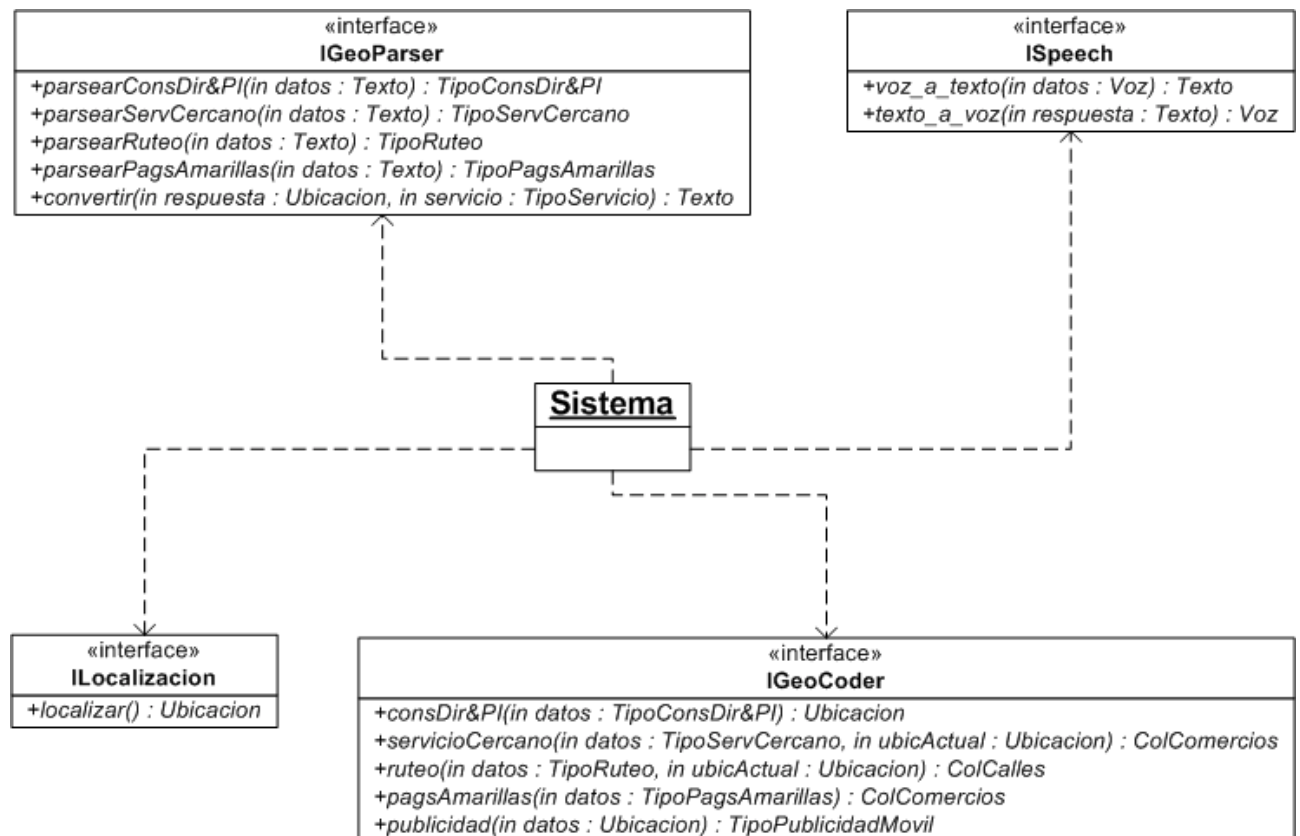
Como se comentó con anterioridad, luego de definir los Diagramas de Clases de Diseño para cada uno de los Casos de Uso, se debe confeccionar un diagrama de estos para el sistema todo.

En nuestro caso, esto se hace necesario para tener una visión general de las operaciones que deben brindar las diferentes interfaces que componen el sistema, ya que el análisis de cada CU arrojó una determinada cantidad de operaciones para cada interfaz, cada una con una lista de parámetros y sus tipos, así como un valor devuelto.

Por lo tanto, esta Sección tiene como objetivo mostrar un gran diagrama que resuma los cinco Diagramas de Clases de Diseño vistos uno a uno en las secciones anteriores en uno solo. Por cuestiones de espacio, simplicidad y ya que no representan lo esencial que se quiere mostrar, se omitirán los tipos de datos definidos por el usuario (<<DataTypes>>) y los enumerados (<<Enumeration>>).

De todos modos, para obtener el diagrama completo (incluyendo estos dos tipos de elementos), basta solo con agrupar todos los tipos de datos y enumerados diferentes utilizados en cada CU y colocarlos dentro del diagrama. Así se obtendrá una correspondencia para cada tipo de dato utilizado en las operaciones de las interfaces, por ejemplo, el tipo de dato `TipoRuteo` devuelto por la operación `parsearRuteo()`, se corresponderá con el <<datatype>> `TipoRuteo` agregado al diagrama.

El presente diagrama será de gran utilidad para el grupo de desarrolladores que eventualmente implementen el sistema aquí diseñado, ya que, como se dijo, presenta un enfoque global de los miembros que deben tener los diferentes componentes de software, y, en particular, sus interfaces.



Por lo tanto, para cualquier grupo de desarrolladores que implemente este proyecto en su totalidad, deberá partir de el presente Diagrama general de Clases de Diseño, ya que representa un resumen de los mensajes con los cuales los componentes de software interactuarán dentro del sistema para llevar a cabo cualquiera de los eventos del sistema.

Si bien el diagrama representa el candidato ideal, esto no significa que el mismo no podrá ser sujeto a modificaciones, siempre y cuando exista un análisis y diseño de la misma o mayor profundidad que el presente que avale la toma de tal decisión.

6. Implementación del Prototipo

Esta sección tiene el objetivo de mostrar lo que efectivamente se implementó. La idea de la presente implementación, es la de demostrar la viabilidad técnica de los planteos y soluciones presentados en este Informe Final.

A continuación se habla de la propuesta del prototipo, y luego muestra la separación en diferentes versiones del prototipo desarrollado.

En *Acerca de la Implementación*, se discuten diferentes consideraciones, toma de decisiones y problemas surgidos en la etapa de implementación.

6.1 Propuesta de Prototipo

6.1.1 Acerca del Prototipo

Para la confección del prototipo se siguió una metodología iterativa incremental. Cada versión de prototipo incluye la anterior y la extiende en una dirección, en un problema a la vez. Es decir, una versión será la extensión de la anterior en un solo aspecto, el cual será la única diferencia entre ambas, incrementando así su funcionalidad. La idea es ir encarando los diferentes niveles de dificultad de a uno por vez.

6.1.2 Motivación por un Prototipo Único Incremental

La metodología concuerda con mi opinión, ya que me parece una forma natural de ir encarando dificultades cada vez mayores y de forma que no tenga cursos de trabajo paralelos. Esto último es muy importante, ya que para obtener trabajos paralelos (o sea que una persona desarrolle cierto componente mientras otra desarrolla otro) naturalmente se debe contar con más de un alumno; y el presente trabajo contó con un solo alumno.

Otro punto a favor de tal elección, es que a lo largo de todo el prototipo desarrollado, aunque se encaran diferentes componentes, todos ellos tienen muchas cosas en común. Todos son desarrollados utilizando el mismo lenguaje de programación y herramientas, todos apuntan a un mismo “producto final”, todos tienen una complejidad similar (aunque no igual) y todos requirieron por parte de quien los hacía un conocimiento similar de cada tema. Es decir que todos los componentes incluidos en el prototipo tienen mucho que ver entre sí (son cohesivos). Y esto mismo favorece una metodología iterativa incremental, en donde no hay necesidad de abordar temas en paralelo.

No obstante, considero que esta metodología iterativa incremental no es la adecuada para desarrollar **todo** el sistema, es decir el sistema que tiene como requerimientos los expuestos en el Capítulo 3 y que luego es analizado y diseñado en el 5. Esta afirmación se basa en que cuando se desarrolle tal sistema (ya dejando de lado el “prototipo” y pasándose a “versiones finales del sistema todo”), dada la complejidad de ciertos componentes y la necesidad de abarcarlos en profundidad (lo cual no se hizo ahora), hace absolutamente necesario que más de una persona se dedique a ciertos componentes, mientras que otras lo hagan con otros.

No olvidar también la cantidad de componentes del sistema global. Esto introduce otra dimensión: ¿cuánto abarcar? Aquí, nuevamente necesitamos un grupo de personas, ya no solamente para estudiar en profundidad ciertos componentes sino para estudiar muchos otros, ya sea en profundidad o no.

A modo de resumen, considero que la metodología iterativa incremental es la adecuada para la confección de los presentes prototipos ya que hubo un solo alumno para hacerlos; pero no la considero adecuada para la confección de todo el sistema en general ya que la cantidad de temas a abordar (con sus disimilitudes) y la profundidad con la que se podrían abarcar son muy grandes.

6.2 Definición de las Versiones del Prototipo

Esta Sección tiene como objetivo mostrar las definiciones de las diferentes versiones del Prototipo, junto con el alcance que tendrá cada versión, los módulos que utilizará, e incluso las tecnologías, productos y programas que serán necesarios para su desarrollo (ver Secciones 2.4 y 2.5).

6.2.1 Versión 1

Prototipo que permita el ingreso de una dirección o intersección para luego ubicarla en un mapa y permitir la navegación en el mismo, es decir poder realizar zoom y desplazamientos. Como requerimientos extra pero esperados, se encuentran: agregar la cantidad que sea de capas (Shapefiles) al mapa; poder ver las características de una capa (etiquetas); poder ver y modificar la escala de un mapa; poder ver la totalidad de las capas agregadas; colorear las respuestas del programa, etc.

Módulos que se usan:

- GeoCoder (dado por MapObjects) – Capa de Aplicaciones
- Servidor de Mapas (dado por MapObjects) – Capa de Contenidos
- Base de Datos Geográfica (dada por los Shapefiles) – Capa de Datos Geográficos

Tecnologías, Programas y Productos necesarios¹¹:

- MapObjects v2.1
- Desarrollar aplicación en Visual Basic v6.0 que utilice MapObjects v2.1
- Shapefiles parciales facilitados por ICA

6.2.2 Versión 2

Prototipo que permita el ingreso de un texto **escrito** que, luego de parseado contra un diccionario permita ubicar la dirección u objeto en un mapa y la navegación en el mismo. Este proceso de parseado se basa en el reconocimiento de palabras claves almacenadas en el diccionario geográfico para luego reconocer frases compuestas por las palabras ya identificadas.

El objetivo de esta versión es el de dotar de cierta “inteligencia” al sistema como para que pueda ser lo más robusto ante posibles (y seguras) fallas por parte de los usuarios para brindar ubicaciones, direcciones, nombres de calles y otros.

Con respecto a los servicios cercanos, se deberá proporcionar al menos alguna forma para decir al usuario que rubros de comercios se encuentran en su cercanía, que sean los pedidos por éste, dentro del radio que éste especifique.

Módulos **agregados**¹² que se usan:

- **Servicios Cercanos** (dado por una interacción entre algoritmos programados y uso de las funciones de MapObjects sobre capas) – Capa de Aplicaciones
- **GeoParser** (dado por algoritmos propios) – Capa de Aplicaciones
- **Diccionario Geográfico** (dado por bases de datos o archivos con cierta estructura definida que contengan las palabras reconocidas como válidas por el sistema) – Capa de Datos

Tecnologías, Programas y Productos **agregados** necesarios:

- Desarrollar un programa que permita parsear un texto contra un Diccionario Geográfico. El programa deberá ser capaz de corregir errores en la entrada, sirviéndose de algoritmos o métodos existentes para dicha tarea (como es el caso del método Soundex, descrito más adelante).
- Además de esto, desarrollar funcionalidades propias que le permitan al usuario indicar que desea ubicar algún tipo de servicio que se encuentre cercano a su ubicación actual. Incluso puede pensarse una parametrización de forma que el usuario indique en metros que distancia tolera como máxima (para dar soporte a los Servicios Cercanos).

6.2.3 Versión 3

Prototipo que permita el ingreso en forma **hablada** (a través de un micrófono) de una dirección u objeto que, luego de parseado contra un diccionario permita ubicar la dirección u objeto en un mapa y la navegación en el mismo.

Para esta versión no se espera el desarrollo de algoritmos de reconocimiento de voz, dada la complejidad que estos envisten, sino que se pretenden observar los errores introducidos por ese paso más, el de la conversión Voz => Texto y como las soluciones propuestas para la Versión 2 soportan o no tales errores.

Módulos **agregados** que se usan:

- **Reconocimiento de Voz** (dado por el ViaVoice de IBM) – Capa de Reconocimiento

¹¹ La utilización de estos productos se justifica en la Sección 6.3.1.

¹² Se utilizará de ahora en más la palabra “agregados” ya que cada versión agrega un conjunto de funcionalidades por sobre las que ya estaban definidas.

Tecnologías, Programas y Productos **agregados** necesarios:

- Obtener programa de reconocimiento de voz que permita interoperar con la aplicación desarrollada en Visual Basic. Un posible candidato es el Speech API SDK v5.1 de Microsoft, que permite desde Visual Basic hacer reconocimiento de voz y otro es el IBM ViaVoice v7.0 que permite dictar directamente a la aplicación activa.

6.3 Acerca de la Implementación

Esta sección discute las decisiones tomadas al momento de implementar las diferentes versiones del prototipo, así como observaciones anotadas durante la realización de los mismos, discusiones acerca de los impactos de pasar de una versión de un prototipo a la otra, etc.

6.3.1 Acerca del Prototipo en General

Porqué Visual Basic 6.0 + MapObjects 2.1¹³:

Para todo el desarrollo del prototipo, se tomó Visual Basic v6.0 como lenguaje de programación y MapObjects v2.1 como herramienta para manejar los elementos geográficos.

Los motivos para elegir el lenguaje de programación son:

- Conocimiento del lenguaje por parte del alumno
- Buen manejo de componentes visuales (como lo es un mapa)
- Buena integración con herramientas de manejo de mapas (como MapObjects)
- Brindar rápido desarrollo y facilidades varias de su IDE (Integrated Development Interface)
- Disponibilidad del software por parte del alumno así como manuales

Los motivos para elegir la herramienta de manejo de mapas son:

- Conocimiento de la herramienta por parte del alumno (por la electiva Sistemas de Información Geográfica)
- Excelente integración con Visual Basic v6.0
- Muy buen conjunto de funcionalidades para manejo de mapas
- Buena integración de ShapeFiles (tipos de archivos de mapas disponibles en ICA)
- Disponibilidad del software (de evaluación) a través de Internet así como de manuales de referencia

En resumen, se preveía un buen desempeño de ambos productos juntos, lo cual evidentemente fue puesto a prueba al momento de comenzar a implementar la primera versión del prototipo, e incluso antes, en una etapa de instalación de los productos y de prueba preliminar.

Luego de esto, se confirmó el buen desempeño de ambos productos al trabajar juntos, ya que para el alumno, se podían cumplir los requerimientos planteados para el prototipo utilizando solamente estas dos herramientas, además claro de otro producto para el reconocimiento de voz.

6.3.2 Versión 1 del Prototipo

Explicación de Esta Versión y sus Funcionalidades:

La primer versión tiene como objetivos introducirse en el manejo de las herramientas (Visual Basic y MapObjects) y comenzar con una versión de prototipo pequeña pero que incluya funcionalidades necesarias para las demás versiones.

Además incluye un componente (antes mencionado) GeoCoder. Un GeoCoder es un componente de software implementado como una operación en este caso, capaz de localizar una dirección estructurada dentro de un mapa. Dentro de esta versión, la implementación de un GeoCoder se hace a través del propio MapObjects, ya que incluye una operación (funcionalidad) que permite buscar una calle y su número dentro de ésta. Con esto, devuelve una referencia geográfica de una dirección estructurada, mostrando en el mapa el resultado. Vale aclarar que se deben efectuar dos

¹³ Por mayor información ver [MapObjects].

búsquedas, primero por el nombre de la calle y luego por el número dentro de esa calle, y que estos resultados parciales deben luego ser combinados mediante un algoritmo y recién luego ser mostrados en el mapa.

También aquí se utiliza un componente (ya mencionado) Servidor de Mapas. El objetivo de éste es el de poder servir de un mapa con sus respectivos datos para poder ser utilizado por un programa. En nuestro caso, quien oficia de servidor de mapas es el MapObjects a través de los Shapefiles, es decir las capas que conforman el mapa. Por esto, el Mapa no es mas que un conjunto de capas (o Shapefiles) una sobre la otra. El diseño de cada capa se discute más abajo. MapObjects juega un papel fundamental en servir de estos mapas para la “manipulación” ya que sin él, Visual Basic no sabría como desplegar un mapa ni como acceder a sus datos y operaciones.

Por último, el componente de Base de Datos Geográfica está dado por los Shapefiles, ya que son ellos quienes representan cada capa del mapa, y son ellos quienes almacenan todos los datos necesarios para desplegar la capa como para buscar elementos dentro de esa capa. Por esto, son el siguiente tema a tratar.

Los Datos de los Shapefiles *Ejes y Direcciones*:

Dado que un Shapefile representa una capa dentro de un mapa (dentro de MapObjects), es de interés conocer como son los datos almacenados dentro de cada capa.

En primer lugar, para esta versión del prototipo, se utilizan dos Shapefiles: uno denominado *Ejes*, y otro denominado *Direcciones*. El de *Ejes* contiene los nombres de las calles junto con un código para cada una. El de *Direcciones* contiene el número de puerta y el código de la calle en la que se encuentra. Por lo tanto, dado el nombre de una calle ingresado en forma escrita por parte del usuario, este debe ser comparado con el nombre de alguna calle contenida en el ShapeFile *Ejes*. Y, los nombres deben ser exactamente iguales para que la búsqueda arroje algún resultado, ya que en esta versión no se tomaron en cuenta cuestiones como corrección de errores o robustez del prototipo (ver método Soundex de la Segunda Versión del Prototipo).

Shapefile:	Ejes
Campos:	Nombre (nombre de la calle)
	cod_calle (código de la calle)

Shapefile:	Direcciones
Campos:	num_puerta (número de puerta)
	cod_calle (código de la calle)

Una vez encontrado el nombre de la calle en *Ejes*, se busca el código para esa calle, y con éste, si es necesario (*Búsqueda por Número*), se busca en *Direcciones* el número de puerta dado que esté ubicado en la calle cuyo código sea el obtenido del Shapefile *Ejes*.

Se considera que el diseño de los datos de los Shapefiles tanto para calles como para direcciones es correcto, ya que se pudo utilizar con relativa facilidad, brinda el soporte necesario para realizar las búsquedas, y no da errores. Si bien existen otras soluciones, es decir, otras formas de modelar las calles y números de puerta de una ciudad, la elección de estas ha demostrado ser correcta.

Utilización de un Documento de Texto para Almacenar las Calles:

Para facilitar la búsqueda de las calles, y poder tener “a la vista” las mismas, se decidió crear un documento de texto que contenga todos los nombres de las calles contenidos en el Shapefile. Por lo tanto, las búsquedas no se hacen directamente sobre el Shapefile sino sobre este archivo de texto. Esto también permite una mayor flexibilidad, ya que, por ejemplo, en la siguiente versión puede ser conveniente guardar, junto con cada calle, información adicional, lo cual puede hacerse directamente sobre el archivo de texto.

La forma mediante la cual se volcaron los datos de los Shapefiles hacia un documento de texto, fue mediante una planilla electrónica (Microsoft Excel). Ya que el Shapefile contiene los nombres de las calles dentro de una tabla (archivo con extensión .dbf), la misma puede ser levantada desde una planilla. Luego, estos datos se copian hacia un nuevo documento de texto, obteniendo así una columna con los nombres de las calles, omitiendo sus códigos de calle ya que no son de interés para las futuras versiones.

Es importante considerar que en lugar de utilizar un documento de texto, se pudo haber utilizado una Base de Datos, e incluso la misma que trae el Shapefile. Esto no se hizo por simplicidad. Es mucho más fácil acceder a un archivo de texto desde Visual Basic, que acceder a algún tipo de tabla dentro de una Base de Datos. Esto último implica el uso de un controlador específico para la Base de Datos que se maneja (ya sea Microsoft Access, DBase, utilizando o no el Open DataBase Connectivity-ODBC, etc.). Además, no existen en este prototipo, consideraciones de performance, ya que el acceso a una Base de Datos puede ser más eficiente que el acceso a un documento de texto.

6.3.3 Versión 2 del Prototipo

Explicación de esta Versión y sus Funcionalidades:

El objetivo de la segunda versión es el de obtener un prototipo que sea capaz de parsear un texto escrito por el usuario de manera de poder encontrar direcciones no estructuradas, incluyendo algún método para corregir los errores introducidos en los nombres de las calles. También se brinda la posibilidad de ubicar comercios en las cercanías del usuario ayudándose del mismo método de corrección de errores pero ahora enfocado a los errores introducidos en los nombres de los rubros de comercios. Esta versión utiliza fuertemente las funcionalidades introducidas en la versión anterior.

La funcionalidad agregada llamada *Función de Localización*, tiene como idea permitir que el usuario ingrese un texto escrito y que a partir de éste se busquen direcciones no estructuradas, ya sean del tipo intersección de calles o del tipo calle y número de puerta. Por esto es que se utilizan las funcionalidades anteriores, ya que primero, se parsea el texto de entrada, y luego que identifica nombres de calles y números de puerta, utiliza las diferentes funcionalidades anteriores dependiendo si se encontró una pareja (calle, calle) o una pareja (calle, número).

Otra funcionalidad incorporada en esta versión, aunque en este caso aparece como oculta al usuario, quien solo ve su resultado, es la que utiliza el método Soundex. Esta se utiliza al momento de buscar una calle, lo que hace que el texto introducido por el usuario sea “corregido” antes de ser comparado contra las calles almacenadas en los ShapeFiles. El beneficio viene dado por la robustez que se le da a la solución, ya que el método Soundex está basado en como suenan fonéticamente las palabras y no en como se escriben. Más abajo se discute en profundidad este método.

El método Soundex es utilizado dentro de un algoritmo propio que hubo que desarrollar. Básicamente, este algoritmo toma como entrada los supuestos nombres de las calles ingresados por el usuario, y genera a partir de ellos un código Soundex, el cual es comparado contra todos los códigos Soundex almacenados correspondientes a los nombres de las calles. De esta manera, se ve que efectivamente, se “corrige” lo que el usuario ingresa, convirtiéndolo a un código, siendo este código el comparado y no lo ingresado por el usuario (más adelante se explicará en detalle su funcionamiento).

Porqué se Necesita un Método como Soundex:

Aquí se pretende mostrar porqué se hace necesario disponer de un método como el Soundex para la corrección de errores en la entrada del usuario. Vale aclarar que se comentará la necesidad de tal método en general, es decir la necesidad de utilizar cualquier método, no necesariamente el Soundex, aunque fue el elegido en este caso.

Las calles reciben un nombre. Este nombre es único dentro de un determinado contexto. Por ejemplo, la calle Dieciocho de Julio está presente en varias capitales departamentales, pero claramente se trata de diferentes calles en cada caso (ver más abajo). Ese nombre, por ser un nombre, no es una palabra que se encuentre en el diccionario de Idioma Español. Y que tiene que ver esto? Todo. Como no es una palabra del idioma, la forma de escribirla, dependerá de la persona que lo haga, y no tiene porqué existir consenso sobre como escribirla.

Esto no quiere decir que los nombres de las calles (como los de las personas) no tengan una forma precisa de escribirse. Generalmente sí la tienen. Mi nombre es Jorge Corral Areán, y así se escribe y no de ninguna otra manera. Pero, si alguien más fuera a escribir mi nombre, léase el nombre de una calle, existe una probabilidad no nula de que lo escriba con alguna falta ortográfica. Este tipo de faltas ortográficas, se ven aumentadas justamente por tratarse de palabras que no pertenecen al idioma español y que muchas personas no están acostumbradas a oír.

Un ejemplo son las palabras “difíciles” del diccionario. Como son palabras que rara vez se utilizan, las personas tienden a escribirlas mal. Algo similar sucede con los nombres (y apellidos obviamente) de las personas o calles. Por ser palabras que en este caso ni siquiera están en el diccionario, su escritura se dificulta. Ni que decir de los nombres en otros idiomas fuera del Español. Pensar en esas calles de nombres extranjeros que a todos dificulta escribir, si es que se las escribe correctamente! Teniendo esto presente, imaginemos el bagaje cultural que debe tener nuestro usuario del sistema si pretende escribir en forma perfectamente correcta absolutamente todos los nombres de calles que necesita hacer referencia, incluyendo los nombres en otros idiomas.

Entonces, ¿porqué es esto un problema para el sistema? Porque los nombres mal escritos, serían comparados directamente contra los nombres de calles almacenados en el documento de texto. Y, dado que la comparación se haría con alguna función de igualdad de Strings (cadenas de caracteres), en una buena parte de los casos Visual Basic diría que no encontró la calle, cuando en realidad no es el caso, sino que el usuario escribió mal su nombre.

Por esto es que se hace necesario algún método de corrección que se pueda aplicar a los nombres de las calles, para poder dotar de robustez al sistema, o al prototipo en este caso. Se deja libre la elección de tal método, e incluso

más abajo se comentarán varios posibles métodos, aparte del elegido (Soundex), pero debe quedar clara la necesidad de adoptar uno.

Ubicaciones Solamente dentro de Montevideo:

En este prototipo, NO se consideró el contexto en el cual ocurren los nombres de las calles. Es decir, que si el día de mañana se decide incorporar los datos (léase el Callejero Departamental) de otra ciudad, por ejemplo Minas (Lavalleja), el prototipo no sabrá distinguir entre las calles que tengan el mismo nombre dentro de Montevideo o dentro de Minas. En particular, por cómo fue diseñado, devolverá el primero que encuentra.

Para tener en consideración el contexto de cada calle, indefectiblemente se debe disponer de información adicional, aparte del nombre de la calle. Una posibilidad, es preguntar al usuario a qué departamento se refiere cuando le hace un pedido al sistema. Claramente esta posibilidad no suena muy convincente ya que requiere intervención extra por parte del usuario, lo que conlleva una pérdida de su tiempo.

Otra posibilidad es que se tome la ubicación actual del usuario (por ejemplo utilizando un GPS), y se decida en qué departamento está en ese momento el usuario, y a partir de esto, inferir que las futuras referencias a calles para esa sesión con ese usuario, serán de ese departamento. Esto tampoco parece una buena solución, ya que el usuario podría estar en Montevideo, pero preguntando como se llega a la calle Dieciocho de Julio de la ciudad de Minas. Aunque, probablemente sea una mejor solución si se la toma “por defecto”, es decir, asumir que se habla del departamento en el que se encuentra hasta que el usuario diga lo contrario.

Existe una tercer posibilidad (obviamente habrán muchas más) que es primero, intentar inferir el departamento en el que se encuentra el usuario, basándose solamente en la información dada por él. Por ejemplo, si el usuario desea ir a Dieciocho de Julio esquina Maldonado, el sistema podría darse cuenta (y de hecho lo hará), que en Montevideo, tales calles no se cortan, pero en la ciudad de Minas sí.

Cómo se aprecia, no es tan simple decidir, en función del contexto dado (o incluso pedir más información) determinar de qué departamento se está hablando. Lo mismo aplica para ciudades, países, regiones, etc.

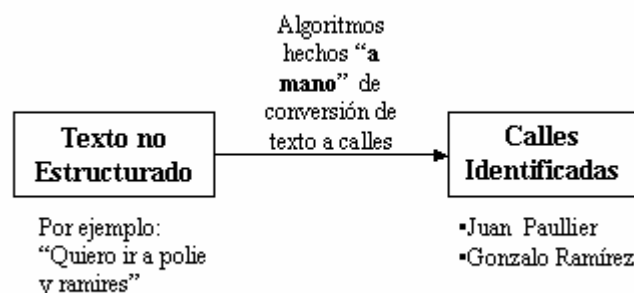
Problema de Reconocimiento de Calles:

El problema radica en poder identificar a qué calle hace referencia el usuario cuando escribe una palabra o frase. Por ejemplo, si el usuario escribe “juan polié” haciendo referencia a la calle “Juan Paullier” (tal como está almacenada en la BD), como hacemos para hacer una correspondencia entre lo que el usuario escribe, y lo que está almacenado en la BD?

Es decir, como establecemos una correspondencia única entre lo que el usuario escribe, y los nombres de calles almacenados en la tablas, para poder así recuperar solamente un registro de la tabla de calles que se corresponda con lo que el usuario escribió.

La solución a tal problema lejos está de ser trivial o exacta. Existen varios métodos para subsanar cierto porcentaje de estos “errores” o mas bien “problemas”. A continuación se estudiarán tres de estos métodos. Antes, se da una breve introducción al proceso que se da al momento de intentar identificar una palabra dicha por el usuario.

El proceso es:



Por tratarse aquí de la segunda versión, se omiten detalles del paso previo, es decir cuando el usuario dicta (habla) al sistema y luego un conversor de voz a texto traduce esto a un texto. Se deja tal análisis para la tercer versión del prototipo.

Una vez que contamos con el texto en palabras escritas de lo que el usuario dijo, o al menos contamos con lo que el conversor de voz a texto creyó que el usuario dijo, hay que parsear (analizar) palabra a palabra ese texto en busca de nombres de calles. (así como de número de puerta, rubros de comercios, etc, pero la búsqueda de NOMBRES es la parte más compleja).

Aquí es donde tenemos la libertad de aplicar cualquier método que consideremos adecuado, por dos motivos: uno es que no existen aplicaciones comerciales accesibles (las hay pocas y muy caras y son algoritmos propietarios), y dos porque la complejidad del problema es abordable desde el presente taller. A continuación hablaremos de los tres métodos que se mencionaron antes para solucionar los problemas derivados de la suma de errores en las dos conversiones.

A mi consideración, existen (al menos) tres métodos aplicables para resolver estos problemas:

1. **Por extensión:** Este método consiste en almacenar en una tabla, cada calle junto con un conjunto de posibles interpretaciones de esa calle, es decir un grupo de palabras que bien podrían considerarse como referidas a la calle en cuestión. O sea, se guarda cada calle y su conjunto de errores (las calles mal escritas pero que hacen referencia a la misma). De este modo, se almacenaría la calle Paullier junto con: Paulier, Polie, Pollié, Polyé, etc. De ahí su nombre (ya que se describe un conjunto por extensión). Cuando se parsea el texto de entrada, se compara cada palabra de entrada con todas las de la tabla. Si alguna coincide con una calle mal escrita que esta guardada en la tabla, el algoritmo busca la calle correcta, y la reemplaza en el texto de entrada, habiendo así identificado esa calle como encontrada en la BD (o diccionario).
2. **Por preprocesamiento:** Aquí se guarda en la tabla solamente las calles bien escritas. En este método, las palabras no encontradas (es decir que no estén exactamente igual que en la tabla) son preprocesadas, generando así nuevas palabras hasta obtener una de las calles de la tabla. Las palabras nuevas que se generan a partir de la incorrecta, son pequeñas variaciones de ésta. Por ejemplo, si se busca la palabra "Paulie" y no se encuentra (porque la correcta es Paullier), se la preprocesa generando palabras como "Paullie", "Paullié", "Paullier", etc. hasta que una de ellas (la última) sí es encontrada en la tabla. Si la palabra incorrecta distara mucho de cualquier palabra del diccionario (tabla), se podría acotar a sólo generar cinco palabras, o sea habiendo probado con cinco alteraciones de la palabra incorrecta. O diez, o veinte, o lo que se desee. Esto podría incluso parametrizarse.
3. **Por Soundex o similar:** Este método guarda cada calle correcta junto con un código (una letra seguida de tres dígitos). Este código es el código Soundex para esa calle. El mismo se genera a partir de un algoritmo (open source) que se basa en como SUENA la palabra para generar un código Soundex que represente ESA o CUALQUIER otra palabra que suene similar. Así, a cada palabra ingresada por el usuario se le asigna un código Soundex y se busca si ese código generado coincide con algún código de la tabla. Si es así, quiere decir que la palabra ingresada es muy similar (fonéticamente hablando) que la palabra almacenada (la calle). En caso de no encontrar el mismo código, puede parametrizarse el algoritmo para que busque palabras del diccionario que estén a cierta distancia del código de la palabra ingresada. Por ejemplo, si el código Soundex de "Polié" es P400, y el de "Paullier" es P402, ambos distan entre sí en dos unidades Soundex, por lo que podría decirse que aunque no son el mismo código, son palabras que suenan muy similar, y podría decirse que se hizo un matching correcto.

A continuación se presenta un cuadro comparativo de los tres métodos:

MÉTODO	VENTAJAS	DESVENTAJAS
Por extensión	- Muy simple y rápido de implementar.	- Usa la fuerza bruta, lo que podría llevar a tiempos de respuesta inadecuados. - Hay que ingresar un GRAN número de posibles errores para cada calle si se desea que sea confiable. (hablamos aquí de <u>decenas de miles</u> en total.) - Utiliza gran espacio en la tabla ya que guarda todos los errores por cada calle existente. - No aparece como una solución “elegante”, o de corte ingenieril.
Por preprocesamiento	- Método inteligente que usa reglas de idioma y de fonética para generar palabras que sean buenas candidatas a ser encontradas en la tabla. - No almacena los errores en la tabla.	- La complejidad de generar palabras “buenas” ya que se requiere de conocimiento en las áreas de idioma y fonética. (para nada trivial).
Por Soundex o similar	-Brinda un método alternativo que es tanto inteligente en su concepción, -Ampliamente utilizado a nivel mundial, -Con una solución open source que permite conocer su funcionamiento, -Rápida respuesta (buena performance), -No almacena los errores en la tabla, -Es de una complejidad abordable.	- Hay desventajas bien documentadas (once en total) pero que han sido analizadas una a una y a nuestro criterio las mismas pueden ser o pasadas por alto por buenos motivos, o directamente no se aplican, dadas las características de esta aplicación.

Por estos motivos, de ventajas y desventajas, es que se cree apropiado el uso del método de reconocimiento de nombres de calles por Soundex, y así es como se encuentra implementada la segunda versión del prototipo.

Consideraciones Previas Acerca del Método Soundex:

Lejos de considerar que el mismo es la mejor opción o el mejor método, considero que es el mejor que se encontró disponible durante la etapa de investigación realizada. Dados los tres métodos considerados, por los motivos expuestos más arriba, considero que el método Soundex, si bien tal vez no sea el mejor, es apropiado, y, lo que es más importante, y como se muestra en el **Capítulo 7** (Testeo), el mismo funciona perfectamente a los propósitos y alcance del presente prototipo.

Es decir, que cumple con los requisitos de robustez deseados al momento de introducir dicho método. Recordar que la robustez de la solución era lo que hacía notar la necesidad de un método como éste.

Además, cabe destacar que el propósito del prototipo no es tener implementada una excelente solución a los requerimientos planteados en la **Sección 6.2**.

Por el contrario, el objetivo global del prototipo es mostrar la viabilidad de incorporar todas las diferentes tecnologías comentadas en una única solución, y mostrar que es posible hacer el sistema general cuyos requerimientos aparecen en el **Capítulo 3**.

Cómo Funciona el Método Soundex:

Esta sección describe en detalle cómo funciona el método Soundex. Originalmente fue desarrollado en los primeros años del siglo XX. Su objetivo era ayudar al llenado manual de los registros de censo de los Estados Unidos. Pese a sus remotos orígenes, el método seguramente es hoy en día la alternativa más utilizada para “matchear” cuando se involucra los nombres en búsquedas automáticas y sistemas de recuperación de datos¹⁴. Desde sus orígenes ha sufrido cambios, hasta su actual configuración que se estudia a continuación.

Este método tiene varias características que lo hacen recomendable: no es propietario (algo muy común en esta área, según lo investigado), es rápido de calcular y de operar con, eficiente y generalmente efectivo para ciertos tipos bien conocidos de errores y por último, es gratis. Incluso, Soundex viene como un método incluido (built-in) en muchos

¹⁴ Por mayor información ver [Soundex, Whose].

sistemas manejadores de bases de datos (DBMS), lenguajes de programación y herramientas de manejo de datos en general.

Sin embargo, el método Soundex no carece de problemas, los cuales son documentados, explicados y discutidos si se aplican o no al presente taller en una sección más abajo.

El método funciona de la siguiente manera. Se describirá la idea general del mismo, aunque en la actualidad se dispone de una enorme cantidad de variantes al método original patentado en 1918 por Robert C. Russell. En pocas palabras, el método convierte una palabra en un código, llamado código Soundex. El código está compuesto por una letra y tres dígitos, por ejemplo C251. La letra es la primer letra de la palabra, mientras que los tres dígitos son una codificación de las primeras tres consonantes de la palabra, en caso de tener tres consonantes. Si no las tiene, simplemente se rellena con ceros hasta completar tres dígitos.

Como se ve, las vocales son omitidas por el método, por considerar que introducen más errores que las consonantes y que son estas últimas las que usualmente hacen a como “suenan” una palabra. Recordar que el método no está exento de críticas o problemas, los que se discuten más adelante.

El criterio primigenio para asignar dígitos a las consonantes es el siguiente:

LETRA	CÓDIGO
B, F, P, V	1
C, S, G, J, K, Q, X, Z	2
D, T	3
L	4
M, N, Ñ	5
R	6

Las demás letras se ignoran (todas ellas).

Como se puede apreciar, el método es en extremo simple, e incluso puede parecer demasiado simple como para funcionar correctamente. Según se mostrará en el **Capítulo 7** de Testeo, el método funciona perfectamente para las necesidades del presente prototipo.

También se aprecia claramente que la forma que tiene el método para codificar, es capturar, a grandes rasgos vale decir, la esencia de la palabra en cuestión. Es decir, cuáles son las consonantes que hacen sonar así la palabra.

Por esto, es que las consonantes “fonéticamente” similares se agrupan y se les asigna un mismo código Soundex. De esta manera, si dos palabras suenan muy similar pero se escriben diferente de manera que la letra en que difieren pertenece al mismo grupo, ambas palabras recibirán el mismo código ya que en definitiva “suenan” similar, más allá de que se escriban diferente y de que cualquier algoritmo “directo” tradicional (por ejemplo de comparación de Strings) devolvería que ambas palabras son diferentes.

Aquí está la esencia del método Soundex: hacer que palabras diferentes se consideren iguales porque según su calificación, ambas suenan parecido.

A continuación se muestran algunos ejemplos.

PALABRA	CÓDIGO
Paullier	P460
Pollier	P460
Poulyer	P460
Juan	J500
Jorge	J620
Blandengues	B453
Dieciocho	D220
Julio	J400
Jalea	J400
Jules	J400

Como se puede ver, a las primeras tres palabras se les asigna el mismo código Soundex, cuando claramente se trata de la misma calle. También se aprecia que a las últimas tres palabras las considera como la misma, cuando claramente hacen referencia a cosas diferentes. Podría pensarse un ejemplo con nombres de calles que de este problema.

Mejoras al Método Soundex:

Se puede pensar de muchas mejoras plausibles de ser aplicadas al método Soundex. Se pretende listar algunas, aunque no todas, a continuación¹⁵.

1. **Mayor número de grupos de consonantes:** Esto redundaría en una mayor precisión al momento de definir a qué grupo pertenece una consonante. El método original propone 6 grupos, pudiéndose alcanzar hasta 8 grupos aproximadamente. El objetivo de esta mejora es el de aumentar la precisión del método en distinguir sonidos diferentes.
2. **Códigos más largos:** Esto permite codificar más de 3 consonantes, por ejemplo 5. El objetivo de esta mejora es el de impedir que palabras prefijos de otras sean indistintas (por ejemplo: con un código de 3 dígitos, Jorges tendría el mismo código que Jorgeson, ya que sólo se codifica la R, la G y la S, fuera de la primera letra, dejándose la N de lado).
3. **Agrupaciones especiales:** Introducir una agrupación especial de consonantes permite definir que ciertas consonantes (o letras en general), al estar juntas, suenan como una sola, perteneciente a un sólo grupo (por ejemplo CH puede ser considerada perteneciente al grupo de la T, o en la palabra Pesce por estar la S y la C juntas, y pertenecer a un mismo grupo, se asigna sólo un dígito a ese par). Se debe tener especial cuidado con ciertas combinaciones aceptadas por el idioma español (por ejemplo en la palabra Pescado no pueden juntarse la S y la C en un solo código).
4. **Repetición de consonantes:** Esto permite definir que la aparición de más de una ocurrencia de determinada consonante se le asigne solamente un grupo (por ejemplo en Petterson la TT se tomaría como una sola T, generando así un solo dígito y no dos). Se debe tener cuidado de no infringir las reglas del lenguaje que permiten que, por ejemplo, la doble ele (LL) tenga un sonido propio. En este caso no se podría asignar un solo dígito Soundex de la L porque el par no suena como L.
5. **Agrupar la primera letra:** El código Soundex original depende muy fuertemente de la primera letra, ya que esta es tomada textualmente como parte del código final (recordar que el código está compuesto por una LETRA -la primera- y tres DIGITOS). Agrupar la primera letra permite que dos códigos Soundex que comiencen con letras diferentes, pero del mismo grupo, sean considerados el mismo código (por ejemplo las palabras Coreanos y Koreanos tienen códigos Soundex iguales salvo por la primera letra, pero por ser ambas del mismo grupo (la C y la K) se considera que ambas palabras son iguales). El objetivo de esta mejora es el de no tener una dependencia tan fuerte sobre la primera letra de la palabra.
6. **Búsqueda exacta la primera vez:** El código Soundex fue pensado para ser aplicado directamente, es decir, se toma una palabra, se la codifica y se busca si ese código ya se encuentra en la BD. Una posible mejora es la de primero, y antes que nada, buscar la palabra en la BD tal y como fue ingresada. En caso de tener éxito y encontrar una palabra en la BD que sea exactamente igual (carácter a carácter), se considera que la búsqueda fue un éxito. En caso de no encontrarse, se aplica el método Soundex. Esto no tiene por qué enlentecer el sistema, ya que la BD solamente almacena las palabras y sus códigos Soundex. En el caso de que hayan muchas palabras almacenadas, puede considerarse armar una estructura de diccionario (un árbol en el que no se repiten prefijos) lo cual aumentaría considerablemente la performance del sistema. El objetivo de esta mejora sería el de no utilizar el método si no es necesario, ya que el mismo no es infalible, por lo que su aplicación podía generar errores. Evitando su uso cuando es posible, puede ser considerado como una mejora.
7. **Diferentes estructuras en las calles:** Qué sucede si en la BD está almacenada la calle Luis Alberto de Herrera, y el usuario ingresa Alberto de Herrera. Primeramente, como ambas palabras (mas bien calles) no son exactamente iguales, se utiliza el método Soundex, obteniendo tres códigos para Alberto de Herrera, contra cuatro códigos para Luis Alberto de Herrera. Obviamente, visto de esta forma, nunca podría llegarse a buen puerto. Pero, una vez que el sistema no encuentre la calle, comenzaría a buscar si la calle ingresada forma parte de alguna otra. Es decir, si Alberto de Herrera es parte del nombre de alguna calle. Naturalmente, esto daría como resultado que SI lo es. Aquí mismo podría considerarse terminada la búsqueda, y decir que la calle ingresada fue encontrada. Pero que sucede si esa calle ingresada forma parte de dos calles? Por ejemplo, de José Alberto de Herrera Lussich, otra calle diferente pero que también la tiene como parte del nombre. En este caso, ya que no

¹⁵ Como se puede apreciar de los ejemplos anteriores, el método Soundex aplicado a ellos incluye la mejora listada bajo el número 4.

hay elementos para decidir, se debe buscar en el contexto. Es decir, si se preguntó por una esquina, verificar cuál de ambas calles se corta con la última calle encontrada. O si se está en el contexto de un barrio determinado, buscar que calle pertenece al barrio, o a la ciudad. En caso aún de no poder discriminar, se le pregunta al usuario que identifique a cuál de las dos calles hace referencia, si a Luis Alberto de Herrera o si a José Alberto de Herrera Lussich.

8. **Calles prefijos de otras:** Esto implica intentar encontrar nombres de calles en el texto ingresado por el usuario de manera tal de tratar primero de buscar las calles de nombre más largo y luego las de nombre más corto. Por ejemplo, si el usuario ingresa "...Carlos Maria Morales..." intentar primero de buscar la calle Carlos Maria Morales. En caso de no encontrarse, intentar con Carlos Maria, y en caso de no encontrarse intentar con Carlos. El objetivo de esta mejora es el de identificar correctamente una calle aún cuando haya otra que sea prefijo de ésta. Supongamos que en el ejemplo la BD contuviera las calles: Carlos Maria Morales y Carlos Maria. Obviamente la respuesta correcta es la de encontrar la calle Carlos Maria Morales, lo cual sucede gracias a este método. Pero que pasaría si se buscara desde el principio? Se buscaría primero la calle Carlos...no está, luego la calle Carlos Maria...si está! Se termina la búsqueda, y se asume que Carlos Maria Morales es la calle Carlos Maria. Error.
9. **Distancia entre códigos:** Ya que los códigos Soundex están compuestos por una letra seguida de tres dígitos, podemos considerar estos dígitos como un número entero de tres cifras (o de tantas cifras como consonantes se deseen considerar). Luego, considerando que dos palabras pueden ser muy similares pero aún así tener un código Soundex diferente, se puede definir una "distancia" entre ambas palabras como la resta algebraica entre los números sacados de sus códigos. Así, por ejemplo, una palabra con un código de P453 podría ser muy similar a una de código P454, y podría interesar considerarlas como iguales. Cabe destacar que la distancia permitida entre ambas palabras (en el ejemplo 1 unidad) debe ser motivo de estudio, ya que de otra manera se podría estar agregando errores.

Cabe señalar que para el caso del Prototipo desarrollado, se tomaron en consideración los las siguientes mejoras: 4 (Repetición de Consonantes), 5 (Agrupar por la Primer Letra), 8 (Calles Prefijos de Otras) y 9 (Distancia entre Códigos).

Problemas del Método Soundex:

Así como se puede pensar de muchas mejoras al método Soundex, también se puede pensar muchos problemas que de la aplicación de éste surgen. A su vez, es interesante ver, para cada problema, dada su naturaleza, si es o no aplicable al presente prototipo¹⁶.

También se debe tener presente que el ingreso, en última instancia (Versión 3 del prototipo) será en forma hablada, por lo que algunos errores pueden ser descartados de antemano.

1. **Dependencia sobre la primer letra:** Los autores indican la fuerte dependencia del método sobre la primer letra, ya que es la que se toma como primer letra del código Soundex. Por esto, la mejora de agrupar la primer letra sobreviene este inconveniente.
2. **Intolerancia al ruido:** Este método es intolerante a la transposición de letras, típicamente venida de errores en el tipeo. Dado que Soundex apunta a semejanzas fonéticas y las transposiciones producen diferencias fonéticas, el método no sirve. Dado que en nuestro caso el ingreso es vía oral, dictada, podemos asumir (no sin considerar sus riesgos) que el problema de la transposición de letras no se da.
3. **Sistemas de transcripción diferentes:** Lenguajes que no son de escritura Romana, pueden introducir errores al pasar de la lengua nativa a lenguajes romanos. Por ejemplo al traducir del Chino al Español o Inglés, seguramente habrá más de una posibilidad de escribir un mismo nombre. Dado el alcance del nuestro proyecto, consideramos a este problema como no aplicable a nuestro caso.
4. **Nombres conteniendo partículas:** En muchas culturas, los nombres puede contener o no elementos opcionales tanto al principio como al final de los mismos. Un ejemplo es el prefijo "Al" en lenguajes árabicos. La misma consideración que el caso anterior.
5. **Diferencias de percepción:** Es el caso en que la forma en que se escribe un nombre no es fácilmente percibible. Por ejemplo, en el nombre ruso "Tkachev" puede no percibirse la T (o no percibirse la K).

¹⁶ Por mayor información ver [Soundex1, Soundex2].

En el nombre “Pfeiffer” puede no percibirse la P. Este tipo de problemas es aplicable al método Soundex en el caso de nuestro estudio. Cabría investigar la presencia de tales nombres en el callejero en cuestión. También es de notar que justamente este problema hace del reconocimiento de calles un problema no trivial de resolver en términos generales, fuera ya del método Soundex.

6. **Consonantes silenciosas:** Es un caso particular del caso anterior, en el que las consonantes mudas (o silenciosas) no se perciben en el nombre. Típico es el caso de la letra H. Caben las mismas consideraciones que para el problema anterior.
7. **Variación sintáctica de nombres:** Diferentes estructuras de nombres son utilizados por diferentes culturas. Por ejemplo, el modelo nombre-apellido no siempre tiene porqué ser válido, ya que en otras culturas puede encontrarse el modelo apellido-nombre. Este problema consideramos que no se aplica por dos motivos: primero, que las diferencias culturales (modelos de sintaxis) en los nombres no representa un problema por tratarse de nombres casi en su totalidad del lenguaje español o bien traducidos a las estructuras del lenguaje español; segundo porque no sería común que un usuario, hablando naturalmente, se refiera primero al apellido de una calle y luego al nombre. Si sería a considerar si los datos introducidos fuesen en forma de texto y no en forma hablada.
8. **Equivalencia de nombres:** Algunos nombres pueden ser “sinónimos” de otros, como es el caso de los sobrenombres. Tanto en el lenguaje inglés, Peggy podría ser considerado sobrenombre de Margaret, como en el lenguaje español, Pepe podría ser considerado sobrenombre de José. Este caso, al tratarse de un universo medianamente normalizado y medianamente formal (el de los nombres de calles), no es de esperarse este tipo de problemas, por lo que son descartados. Otro tipo de problemas menos obvio pero que también se descarta, es que en ciertas culturas (típicamente orientales) nombres tales como “Wu” y “Ng” pueden ser considerados como iguales.
9. **Iniciales:** El problema con las iniciales radica en ingresar nombres de calles utilizando iniciales y tener almacenados los nombres con sus nombres completos; o viceversa. Este problema puede considerarse dentro de las mejoras propuestas al método soundex (mejora 7) la que refiere a que las diferentes estructuras que el nombre de una calle puede tener. Es decir, que podría descartarse la inicial, solamente utilizada en caso de ambigüedad.
10. **Respuestas sin ordenar:** Ya que el objetivo es agrupar palabras que suenen similar, soundex no provee un mecanismo para “rankear” los resultados. Esto puede paliarse definiendo -como sugiere la mejora 9- una distancia entre los resultados. Aún así, en lo que concierne a nuestra aplicación, no interesa dar un conjunto de resultados rankeados, sino mas bien interesa devolver un único resultado con altas probabilidades de ser correcto, o a lo sumo dos resultados y pedir la interacción del usuario para que decida.
11. **Pobre precisión:** El método Soundex reduce las diferencias entre letras a tal punto que palabras muy disímiles pueden converger en el mismo código. Esto, es claramente uno de los principales problemas del método que no tiene una solución trivial, y que también adolece el prototipo de este taller, tal como se mostró en los ejemplos más atrás. A su vez, el método se vuelve peor a medida que la cantidad de datos crece, ya que aumenta la probabilidad de que el nombre de una calle tenga el mismo código que el nombre de otra. Este problema se presenta como infranqueable en este prototipo, aunque puede llegar a relativizarse mucho viendo los resultados obtenidos.

Aquí cabe destacar que de los problemas analizados, los siguientes son los que afectan realmente al desarrollo del prototipo: 5 (Diferencias de Percepción), 6 (Consonantes Silenciosas) y 11 (Pobre Precisión).

Los Datos del Shapefile Comercios:

Esta sección discute los datos contenidos en el Shapefile *Comercios* suministrado por la empresa ICA y sus desventajas en cuanto a su estructura. Los datos de esta capa, incluyen únicamente la ubicación y el rubro de los comercios dentro de la zona del mapa brindado. Es decir, que cada registro de la tabla conteniendo los datos del Shapefile, tienen dos campos: la ubicación dentro del mapa de dicho comercio y el rubro al cual pertenece.

Shapefile:	Comercios
Campos:	direccion (ubicación del comercio)
	rubro (rubro del comercio)

Este diseño es, en principio, adecuado hasta lo que se ha discutido hasta ahora. Pero, ¿que sucede con la categorización de los rubros? Es decir, ¿cómo se clasifican los rubros contenidos dentro del Shapefile *Comercios*?

La clasificación contenida dentro del ShapeFile consta, como se mencionó, simplemente de un campo asignado al rubro de cada comercio. Por lo tanto, no existe ninguna posibilidad de “anidar” clasificaciones, es decir, clasificaciones dentro de clasificaciones, obteniendo así una jerarquía de rubros.

No solamente no existe tal jerarquía, sino que los rubros contenidos en el campo rubro, son múltiples. O sea, que cada comercio pertenece en verdad no a un único rubro, sino a un conjunto de rubros. A dicho conjunto de rubros múltiples los denominaremos “super-rubros” por tratarse de varios rubros juntos conteniendo en realidad subrubros dentro.

Esta clasificación, o mas bien, agrupación de rubros, se estima que se adoptó por simplicidad, y para mantener el conjunto de rubros bastante acotado. Así, un bar pertenece al super-rubro “cafeteria/bares/confitería”, al igual que una cafetería o una confitería. Claramente puede verse una necesidad de depuración de tal agrupamiento, pues no faltan ejemplos en los que no cabe comparar un bar con una confitería.

Otro problema visto en los datos de dicho Shapefile, es la forma en que se ingresaron los nombres de los rubros. Por ejemplo, el super-rubro visto antes que nuclea a los bares, confiterías y cafeterías, ni siquiera aparece como se mencionó aquí entre comillas, sino que aparece como “cafeteria/bares/confi”. Notar la palabra (rubro) “confi”. Es de suponer que significa “confitería”. Otro ejemplo de este tipo de errores, que seguramente sean errores de tipeo los cuales no han sido pasados por ningún proceso de depuración, es el de los super-rubros “juegos elect./pool” y “juegoelec./pool”. Aquí, si no se efectúa un preprocesamiento por parte del programa a desarrollar, los dos super-rubros aparecerán como distintos, cuando claramente se tratan del mismo. Es decir, que queda por parte del desarrollador, primero darse cuenta que estos errores ocurren, y segundo, tomar medidas para de alguna forma “juntar” estos dos super-rubros en uno solo, aunque sea lógicamente dentro del programa.

Un último tipo de problema visto en los datos de muestra, son los tildes. En ocasiones aparecen, se utilizan, pero en otras no. Un ejemplo es el super-rubro ya visto “cafeteria/bares/confi”, donde se omite el tile en cafetería, mientras que en “papelería/librería” se los incluye.

Lejos de intentar convertirse en una discusión acerca de los datos de muestra suministrados, lo que se pretende es mostrar los errores típicos de los que están plagados los datos que se manejan en este tipo de sistemas, y las correcciones que los desarrolladores deben tener presente.

Más abajo se propone un nuevo diseño para el Shapefile *Comercios*, pero antes se discute los problemas para corregir los errores introducidos en los rubros. Este orden viene de que la discusión que sigue menciona los super-rubros, por lo que debía estar luego de la presente sección que los introduce.

Problemática Acerca de los Rubros:

Los rubros no son nombres. Esto plantea una pregunta: ¿Será conveniente considerar a los rubros como nombres, para así utilizar el método Soundex el cual está especialmente diseñado para nombres?

Una primera respuesta sería afirmativa: Si, conviene considerar a los rubros como nombres, y aplicarles el método Soundex. Es decir, que cuando el usuario ingrese algún rubro, este será convertido a un código el cual se buscará en la BD.

Si bien este planteo parece razonable, no carece de asperezas. Por ejemplo, que los rubros, al no ser nombres propios como los de las calles, no tienden a ser expresados con tanto error como pasa con éstas. Es decir, ya que son palabras que se encuentran en cualquier diccionario de Español, difícilmente el usuario escriba mal palabras como “bar”, “peluquería”, “panadería”, etc. No nos olvidamos aquí que igualmente podrán contener cierto grado de error al ser ingresadas, pero es de esperar que éstos no sean tan frecuentes como en los nombres de las calles.

Otra consideración es que dada la forma en que están almacenados los rubros en la BD (cosa que damos como dada para el prototipo pero que se propone un nuevo diseño, ver más abajo), los mismos se encuentran agrupados. Es decir, que rubros tales como “fiambrería” y “quesería”, se encuentran en un solo rubro llamado “fiambrería/quesería”. Esto ocurre con muchos otros. Esto imposibilita que el rubro pronunciado por el usuario sea cotejado directamente contra estos rubros almacenados en la BD. Algún procesamiento extra es requerido, para de alguna forma “separar” los diferentes rubros que estén agrupados dentro un “super-rubro” en rubros individuales.

Aún otra consideración es que si bien los nombres de las calles pueden ser pronunciados con un alto grado de error, siempre se refieren a un mismo nombre. Por ejemplo, “Paulie”, “Paullier” y “Polié” hacen referencia a la misma calle, y como se ven son palabras MUY similares. Por otro lado, mientras algunas personas llaman “frutas y verduras” al rubro que se dedica a la venta de éstas, otras podrán llamarle “feria”, “mercadito”, “frutería”, “verdulería”, etc.

Aquí se ve claramente que la aplicación del método Soundex para resolver estos conflictos es inviable, ya que el mismo se dedica a palabras que suenen similar, y obviamente éstas no suenan similar.

A modo de resumen, hay que tener en cuenta las siguientes consideraciones:

1. Por no ser nombres, y ser palabras del idioma Español, no suelen tener un alto grado de error.
2. Existen “súper-rubros”, los cuales requieren de algún tipo de preprocesamiento.
3. Palabras muy diferentes pueden hacer referencia a un mismo rubro.

La solución propuesta consta de dos etapas. Primero, separar los “súper-rubros” en rubros individuales. Esto es absolutamente necesario como parte de ese “preprocesamiento” al que se hacía referencia más arriba, ya que el usuario no se espera que diga “Quiero una fiambrería/quesería cerca de mí”. Con esta forma de Extensión (ya que hacemos por extensión una división del súper rubro) estaríamos atacando la segunda consideración mencionada anteriormente.

Pero además, podemos atacar también la tercer consideración con mínima modificación. Cuál modificación? La de agregar como parte de esa extensión (separación) de rubros, otras formas bajo las cuales los usuarios pueden hacer referencia a un mismo rubro. De esta forma, el originalmente rubro “fiambrería/quesería” se convertiría en los rubros “fiambrería”, “quesería”, “chacinería”, “fiambres” y algún otro que se desee.

Cabe notar que más que separar los súper rubros, lo que se hace es construir una nueva tabla con los rubros individuales (más otras posibles formas de mencionarlos) donde cada rubro apunta al súper rubro que lo originó, ya que el mapa de la aplicación sólo conoce esos súper rubros.

También es importante observar que en este caso, el método de Extensión es implementable, ya que hay una cantidad finita, acotada y pequeña de nombrar a los rubros. No pasa aquí como en las calles, que hay muchas maneras diferentes de decir su nombre. De todos modos no es un trabajo menor el de ingresar en una BD todos los posibles nombres.

La segunda etapa, es utilizar el método Soundex para atacar la primer consideración, que hablaba de los posibles errores con los que se pronuncian los rubros. Es decir, lo que primero se hace, es tomar el rubro ingresado por el usuario y convertirlo en un código Soundex, el cual es cotejado contra todos los rubros que por extensión se obtuvieron de los originales. Esto soluciona problemas como pronunciar “pisería” en lugar de “pizzería” (ya que la las dos Z suenan como una, y que la S suena como una Z, y los tildes no importan por estar en vocales).

A modo de resumen, se presenta un ejemplo que hace buen uso de ambas etapas de la solución: “Quiero inportadores a 200 metros de mi ubicación actual”. Aquí, la palabra “inportadores” no sólo está mal pronunciada, sino que es parte de un súper rubro (mayorista/importador). Por lo tanto, primero se utiliza Soundex para obtener que lo que efectivamente el usuario quería decir era “importador”, para luego ver que “importador” es una extensión de un rubro individual que apunta al súper rubro “mayorista/importador”. Recién ahora, la aplicación está en condiciones de buscar todos los “mayorista/importador” dentro de un área de 200 metros de la ubicación dada por el usuario.

Proposición de un Diseño para los Datos del Shapefile Comercios:

Las dos secciones anteriores explicaron el diseño actual de los datos contenidos en el Shapefile *Comercios*, y de cómo cuidarse para utilizarlos.

Esta sección, fuera de esto, propone un nuevo diseño de los datos para poder, en una futura versión, lograr una mejor solución al momento de clasificar y buscar los rubros de los comercios. La propuesta es clasificar los rubros, donde cada clasificación contenga un solo rubro. La cantidad de clasificaciones seguramente sea un tema de discusión por sí solo, pero podríamos estar hablando de dos o tres niveles de anidamiento, es decir clasificaciones de clasificaciones.

De esta manera, evitamos primero, el problema de tener super-rubros, ya que cada comercio pertenece a un único rubro, el cual a su vez, pertenece a otro único rubro mas amplio, y el cual a su vez pertenece a otro único rubro mas general, el cual a su vez... tantas veces como se desee.

Segundo, evitamos, o mejor dicho, introducimos una jerarquía evitando el “desorden” previo. Así, teniendo más información de cada comercio (ya no a qué super-rubro pertenece sino a que dos o tres rubros pertenece) podemos brindar mejores servicios. Por ejemplo, se puede pensar que el usuario pregunte por “comercios dentro de la rama de la construcción”. Claramente, esa rama (o rubro) contiene varios sub-rubros como “materiales de obra”, “carpintería”, “albañilería”, etc. Así, el sistema devolvería todos los comercios cuyo “gran” rubro sea “construcción”, para cualquier rubro particular.

Otra mejora propuesta, aunque obvia, no es menos importante: la de ingresar los datos utilizando cierto estándar, de manera de utilizar siempre los tildes, o no utilizarlos nunca, abreviar de cierta manera los rubros o nunca abreviarlos, cuando separar palabras y cuando no, etc. y por sobre todo, someter a los datos a una constante supervisión.

Impacto Sufrido al Abordar la Presente Versión:

El propósito de esta sección es el de comentar el impacto sufrido al pasar de la primer versión del prototipo a la segunda. El motivo por el cual esta sección no se incluyó en la primer versión del prototipo es que no había ninguna versión anterior.

Con respecto a las funcionalidades implementadas en la primer versión, como ya se comentó, algunas fueron utilizadas tal cual fueron hechos, y otros (*Búsqueda por Intersección* y *Búsqueda por Número*) además de haber sido incluidas tal cual, fueron incluidos en la incorporación de una nueva funcionalidad (*Función de Localización*).

Por esto, podemos decir que no se experimentó ningún impacto, o al menos ninguno negativo, ya que el código se utilizó y se reutilizó de manera satisfactoria.

La incorporación del GeoParser (componente que analiza el texto ingresado por el usuario) fue realizada sin problemas extras fuera de los propios ya comentados (la dificultad de encontrar los nombres de calles dentro del texto introducido por el usuario).

La incorporación de la funcionalidad de localización, no significó cambios en lo ya hecho, sino mas bien agregados al código existente.

La incorporación del Shapefile *Comercios* tampoco presentó nuevos problemas.

Por todo esto, podemos decir que no existen comentarios interesantes con respecto a cambios o problemas surgidos de la implementación anterior (Versión 1) bajo la luz de los nuevos agregados de la presente implementación (Versión 2).

6.3.4 Versión 3

Explicación de Esta Versión y sus Funcionalidades:

El objetivo de esta versión, es el de dotar de un sistema de reconocimiento de voz al Prototipo que lo haga capaz de interpretar un texto que el usuario hable (dicte) y convertirlo en un texto escrito, para poder así utilizar íntegramente las funcionalidades de la Versión 2 del Prototipo sin modificación alguna (ver sección acerca del impacto sufrido para ver detalles).

Todas las funcionalidades vistas para las versiones 1 y 2 del Prototipo, continúan utilizándose de la misma manera, con la única diferencia que ahora reciben en forma **indirecta**, un texto hablado por el usuario. El énfasis en la palabra indirecta viene de que en realidad, ya que la herramienta utilizada para el reconocimiento de voz devuelve un texto con lo que interpreta del usuario, este mismo texto es pasado a la funcionalidad elegida por el usuario, ya sea de localización o el de cercanías.

El motivo para incorporar una herramienta de reconocimiento surgió ya que sería muy importante mostrar que este Prototipo, además de mostrar la viabilidad de la solución lógica, que es posible incorporar la voz a la solución.

Ya que por ahora se asume (y en el **Anexo C** se muestra) que la voz es la interfaz más natural para el ser humano, y ésta debía ser por lo tanto la interfaz para el sistema en cuestión, no podía faltar justamente la parte del sistema que reconoce el texto escrito del usuario y lo manda en forma de texto a las demás funcionalidades previamente implementadas.

Motivos para Agregar el Reconocimiento de Voz:

Esta sección pretende enumerar una serie de motivos y ventajas, aplicables a nuestro prototipo claro está, que vienen de la implementación del módulo de reconocimiento de voz. Algunas ya fueron expresadas en la sección anterior, y otras se agregan aquí:

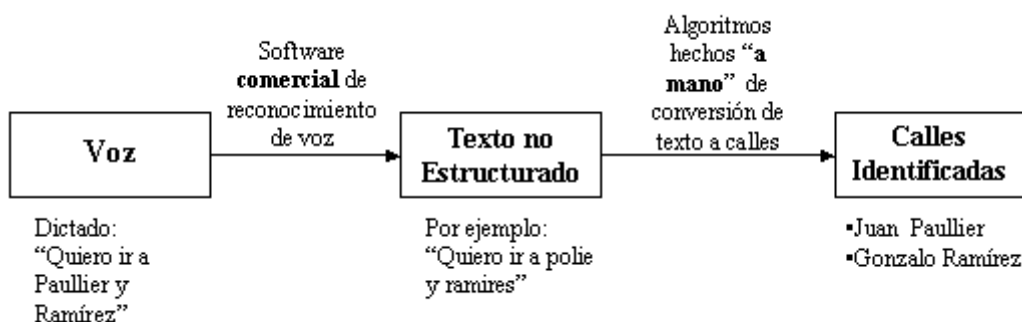
- Poder mostrar una parte central del proyecto, que es que el mismo es implementable incluyendo el reconocimiento de voz,

- Utilizar la interfaz por naturaleza del ser humano,
- Permitir así la futura utilización del prototipo a través de un teléfono, ya sea celular o de telefonía fija,
- Para el caso anterior, permite utilizar la ubicuidad del teléfono, es decir su disponibilidad y su masiva utilización a nivel mundial, extendiendo así su aplicabilidad y masividad,
- Facilitar el uso del prototipo, y hacer mas amigable su interfaz mediante el uso de la voz como entrada de datos,
- Poder ver el impacto sufrido de haber agregado un software de reconocimiento de voz sobre la implementación de la versión 2 (ver sección de impacto sufrido),
- Haber podido estudiar los diferentes software existentes y estar en condiciones de proponer un nuevo tipo de software de reconocimiento de voz que mejoraría el uso del presente prototipo, al menos a los efectos del presente proyecto.

Problemática del Reconocimiento de Voz¹⁷:

En esta sección se pretende continuar con la discusión comenzada secciones atrás, en la versión 2 del Prototipo, donde se comentaban los problemas para el reconocimiento de calles. La diferencia es que aquí debemos introducir el concepto del reconocimiento de voz.

Por esto es que comenzamos nuestra discusión con el diagrama completo del proceso de reconocimiento de calles, el cual complementa el visto anteriormente:



La conversión de voz a texto es un proceso completamente automático del cual solo podemos confiar (o esperar) que sea hecho lo mejor posible, no pudiendo modificar de forma alguna los algoritmos utilizados. Los principales motivos de esto (aplicables a nuestro caso) son:

- Los programas de reconocimiento de voz, en todos los casos analizados por quien escribe, aplican soluciones propietarias, es decir que los algoritmos utilizados no se encuentran disponibles. No son de código abierto. Esto es entendible, ya que el tiempo y costo de realizar tales algoritmos no son para nada despreciables.
- La complejidad que revisten las soluciones (por ende los algoritmos también) de tales programas, exceden los límites del presente taller, ya que para implementar uno, se necesitaría experticia en varias áreas: fonética, lingüística, idioma y una gran inversión de tiempo.

Por lo tanto, debemos limitarnos a estudiar las soluciones disponibles, evaluarlas, y decidir cuales de ellas se ajusta mejor a las necesidades del prototipo y utilizarla en el mismo, justificando fuertemente su adopción y discutiendo, en caso de existir, cuál sería una mejor solución mas ajustada al presente proyecto.

También debemos tener en cuenta que en esta conversión de voz a texto se pueden introducir algunos errores, del tipo fonético. En nuestro diagrama de ejemplo, el conversor devolvió "polie" a lo que dijo el usuario, pero bien podría haber devuelto "polyé" o incluso "pollié". Luego veremos que sólo se devolverá una palabra en particular (la más probable dentro del diccionario).

A modo de resumen, si bien en esta conversión solo somos espectadores, sabemos los tipos de errores que podrían generarse si se utilizara un tipo de reconocedor de voz diferente a los habituales. Estos errores se discuten más

¹⁷ Ver Anexo C.

abajo en una sección aparte, ya que requiere conocer al menos *grosso modo* como es que funciona un software típico de reconocimiento de voz y cómo podría funcionar uno que no fuera de los estándares.

Propuesta para un Nuevo Software de Reconocimiento de Voz:

Todos los software de reconocimiento de voz funcionan de forma muy similar. Esto es, que todos devuelve palabras contenidas dentro de su diccionario, las que tienen la mayor probabilidad de ser las que el usuario dijo.

Imaginemos por un momento que esto no fuera así. O sea, que en lugar de que el software de reconocimiento devolviera la palabra del diccionario que mayor probabilidad tenga, devolviera una palabra que se acercara a lo que el usuario dijo, y no a la palabra del diccionario que más se acerque según la función de probabilidad utilizada. Llamemos a tal sistema, un Reconocedor en “Forma Directa”, ya que devolvería directamente lo que el usuario dice, y no la palabra del diccionario que supuestamente el usuario dijo.

Esto implicaría, que cuando el usuario diga una palabra que esté en el diccionario, naturalmente se devuelve la palabra del diccionario. Con “esté” queremos decir una altísima probabilidad, imaginemos, mayor al 90 por ciento para fijar ideas. Y, cuando el usuario diga una palabra que no esté en el diccionario, imaginemos una probabilidad menor al 90 por ciento, el sistema devolverá una palabra o frase que más se acerque a lo que el usuario dijo, sabiendo que tal palabra o frase no pertenece a su diccionario.

Fuera de discutir si esto daría un mejor resultado o no, parece más que interesante el propio planteo de la posibilidad! Ahora, porqué es interesante tal planteo? Que introduce o cambia con respecto al prototipo? La respuesta es la siguiente: De esta forma, nos evitaríamos la etapa de “enseñarle” al sistema todo el callejero departamental de Montevideo, por citar tal vez el caso más obvio.

Ya que los nombres de las calles de Montevideo son en su mayoría nombres y apellidos de personas, estos nombres y apellidos obviamente no estarán en el diccionario del reconocedor. Podrían estar algunos nombres, pero hasta allí y con suerte!

Entonces, como se hizo para la presente versión del prototipo, se tuvo que enseñar al reconocedor elegido (ViaVoice, ver secciones más abajo) los nombres de las calles de Montevideo. Para el caso del reconocedor elegido, el mismo pide para cada calle, como es que “suena” esa calle, lo que denomina la “palabra fonética” y además, pide al usuario que diga en voz alta una vez el nombre de tal calle, así puede reconocerla en el futuro.

Todo este procedimiento de enseñanza podría evitarse en caso de utilizar un reconocedor en “Forma Directa”. Aquí se introdujo el concepto del nuevo reconocedor de voz en forma directa. En la sección que habla de porqué se eligió el IBM ViaVoice se continúa con la discusión.

Otra modificación que se le podría agregar al nuevo reconocedor en forma directa, es la de acotar el diccionario. Es decir, acortar la cantidad de palabras presentes en el diccionario.

El motivo de tal modificación es muy simple: si el diccionario no contiene palabras que el usuario no utiliza, y contiene solamente aquellas que el usuario sabe que utilizará, las palabras que ya no están no serán tenidas en cuenta como posibles candidatas, por lo que estaríamos aumentando la probabilidad de las palabras que quedaron.

Imaginemos el siguiente ejemplo. El diccionario original contiene solamente las siguientes palabras: {“hola”, “ola”, “que”, “tal”}. El usuario dice: “hola que tal”. El reconocedor debe asignar probabilidades a las palabras “hola” y “ola”, para ver cual de las dos es más parecida a como el usuario dijo “hola”. Pero, si quitamos la palabra “ola” del diccionario, es fácil ver que el proceso de reconocimiento se vuelve no solo más rápido porque hay menos palabras, sino que más exacto, ya que las palabras con nula posibilidad que el usuario diga no están presentes en el diccionario.

Por lo tanto, la propuesta de un nuevo reconocedor, sería uno que introdujera estas dos modificaciones.

Porqué se Eligió el IBM ViaVoice v7:

La idea de esta sección es argumentar la adopción del software de IBM para reconocimiento de voz, el ViaVoice Millennium versión 7.¹⁸

¹⁸ Por mayor información ver [ViaVoice].

En primer lugar, y antes de las argumentaciones de rigor, hay que decir que el software elegido era el único candidato con posibilidades reales de ser útil al presente proyecto. Los productos analizados en esta etapa fueron los siguientes:

- IBM ViaVoice Millennium
- Microsoft Speech SDK v5.1
- Dragon Dictate

Motivos para la adopción del ViaVoice:

- Es un reconocedor de voz en idioma Español,
- Se pudo conseguir relativamente fácil a un costo accesible en el mercado actual,
- Tiene todas las prestaciones que se puede esperar de un software de esta especie:
 - Interacción con otras aplicaciones,
 - Capacidad de entrenamiento del locutor para mejorar porcentaje de aciertos,
 - Capacidades de “enseñarle” nuevos términos y agregarlos a su diccionario,
 - Buena capacidad de dictado en lo que a fluidez se trata,
 - Excelente nivel de aciertos luego de su correspondiente entrenamiento por parte del locutor,
- En una etapa de prueba integrándolo a los prototipos, satisfizo las expectativas.
- Es muy simple de instalar, usar y mantener.

Porqué no se adoptó el Microsoft Speech SDK v5.1:¹⁹

- No se dispone de un Language Pack en idioma español,

Ventajas del Speech SDK contra el ViaVoice:

- Se integra perfectamente con Visual Basic, pudiendo programar directamente desde ese lenguaje utilizando las capacidades de reconocimiento del habla,
- Se baja de Internet, por lo que su disponibilidad es inmediata y sin costo.

Con respecto al Dragon Speech Recognition²⁰, sus similitudes con el IBM ViaVoice lo colocan a la par de éste a la hora de elegir. Pero, hay que tener en cuenta:

- La disponibilidad del software de la empresa Dragon no es buena en nuestro mercado actualmente.

Discusión Acerca del Reconocimiento De Voz En “Forma Directa”:

Recordemos que por reconocimiento de voz en “Forma Directa” entendemos un software de reconocimiento del habla que enfrentado con una palabra que tenga una alta probabilidad de no pertenecer a su diccionario, decide devolver la palabra o frase corta que más se acerque a lo que entendió.

Con esto, se logrará obtener las “palabras fonéticas” de cada calle de Montevideo, ya que si las mismas no son agregadas al diccionario, al ser habladas, el software se dará cuenta que hay una muy alta probabilidad de que la palabra efectivamente no está en su diccionario, e intentará escribir la palabra que más se le parezca, sabiendo que está devolviendo algo que no pertenece a su diccionario. Esta es la idea básica de un reconocimiento de voz en Forma Directa.

A falta de encontrar tal software, es evidente que nunca fue una opción. No obstante, se pueden sacar conclusiones interesantes si se imagina el uso de tal software. Tales conclusiones pueden resumirse en:

- Debido a que el software de reconocimiento no forzará su salida a una palabra del diccionario, se obtendrán los nombres de las calles (y de toda palabra que no entienda) tal y como se dijeron, por lo que se necesitará hacer un uso extensivo de los algoritmos que implementan el método Soundex (o cualquier otro método que se considere apropiado) para lograr comprender lo dicho.
- Se tendría más control sobre la corrección de errores en los nombres de las calles, debido a que ahora sobre quien recae la responsabilidad de “entender” lo que el usuario dice, es el programa, no el software de reconocimiento. Por tal, se pueden elaborar rutinas y algoritmos tan complejos como se desee para lograr un mejor o peor entendimiento de las palabras que no están en el diccionario.

En resumen, sería en extremo interesante probar la integración de tal herramienta con el Prototipo en funcionamiento, para probar o refutar las (prematuras) conclusiones aquí expuestas.

Cuadro comparativo:

¹⁹ Por mayor información ver [MSSpeech].

²⁰ Por mayor información ver [Dragon].

HERRAMIENTA	VENTAJAS	DESVENTAJAS
IBM ViaVoice	<ul style="list-style-type: none"> • Idioma • Buena disponibilidad y bajo costo • Totales prestaciones • Buen desempeño en general 	<ul style="list-style-type: none"> • No se integra con Visual Basic
Dragon Recognition	Idem ViaVoice	Idem ViaVoice
Microsoft Speech SDK	<ul style="list-style-type: none"> • Excelente integración con Visual Basic 	<ul style="list-style-type: none"> • Idioma (solo en inglés)
Reconocedor en “Forma Directa”	<ul style="list-style-type: none"> • Uso extensivo de algoritmos de corrección • El control lo tiene el programa desarrollado, no el software de reconocimiento 	<ul style="list-style-type: none"> • No se encontró ninguna implementación.

Errores Introducidos por el Reconocimiento de Voz en “Forma Directa”:

Ahora que se introdujo el concepto del reconocedor en forma directa, y como se retrasó la discusión de los errores posibles introducidos por un reconocedor en forma directa, es que ahora se los comenta.

Discutir los errores introducidos por un reconocedor estándar casi pierde sentido, ya que más que errores el reconocedor estándar basado en palabras del diccionario, aumenta la precisión del sistema, con la desventaja obvia e importante del entrenamiento que se le debe dar. Por esto es que tal discusión se omite. Además, el único error que podrá cometer el reconocedor es devolver una palabra o frase que se asemeje a lo que el usuario dijo, pero sin serlo, lo cual, en un diccionario acotado, y con la precisión del elegido, resulta difícil.

En el contexto del uso del reconocedor en forma directa es que podemos considerar la introducción de errores por parte del propio reconocedor, ya que cuando el usuario ingrese el nombre de una calle que no pertenece al diccionario, el reconocedor devolverá en forma escrita una palabra o frase que más se asemeje fonéticamente a lo que el usuario dijo.

Claro está, que dicha palabra o frase puede estar lejos de ser el nombre de la calle a la que el usuario intentaba hacer referencia. Por lo tanto, nos encontramos con errores. Tales errores pueden ser de diferentes tipos: falta de letras, transposición de letras, omisiones de tildes, agregado de letras, falta de espacios en blanco, etc.

Pero, no olvidar que el método Soundex, no puede dejarse fuera de esta discusión. Porque? Porque justamente fue diseñado para reconocer palabras que “suenen” similar a otras, y no necesariamente se escriban igual. Este sería exactamente el caso en cuestión. El reconocedor devolvería palabras fuera del diccionario, que seguramente estén mal escritas, es decir que no sean exactamente el nombre correcto de la calle, pero que seguramente suene muy similar al nombre de la calle. Exactamente lo que está esperando el método Soundex.

Entonces, ya que como se verá a continuación el uso del método Soundex pierde el sentido ante la presencia de reconocedores estándares (porque las calles o son perfectamente escritas porque fueron sacadas del diccionario o se reconoce cualquier otra palabra que no tiene nada que ver), ahora toma todo el sentido del mundo utilizar en conjunción el método Soundex junto con un reconocedor en forma directa.

Impacto Sufrido al Abordar la Presente Versión:

La idea de esta sección, es documentar el impacto sufrido sobre la solución de la Versión 2 del Prototipo al avanzar hacia la Versión 3, o sea el impacto sufrido en la adopción del reconocimiento de voz.

El problema era reconocer los errores (o nuevos problemas) surgidos al introducir el software de reconocimiento de voz. En realidad, se vio que tal introducción de software, más que problemas, aportó simplicidad a la solución. Esto viene por los siguientes motivos.

El primer motivo viene de que, en la Versión 2, cuando el texto se ingresaba manualmente al programa, se vio que surgían varios problemas:

- Transposición de letras,
- Falta de algunas letras,
- Diferentes pronunciaciones.

Algunos de estos problemas, al momento de decir oralmente las palabras, dejan de ser importantes. El caso más claro es el de la transposición de letras. Es difícil imaginar a una persona diciendo “Dieciocho de Julio” en vez de “Dieciocho de Julio”. Cuidado que aquí no se está diciendo que estos errores no se den cuando se habla, sino que su frecuencia se espera que sea mucho menor que cuando se escribe.

El segundo motivo viene del propio software de reconocimiento. Debido a que el IBM ViaVoice funciona de manera tal que devuelve la palabra contenida en su diccionario que es más parecida a lo que el usuario habla, nunca va a devolver algo fuera del diccionario. O sea, que cuando el usuario diga una palabra y la misma sea cotejada contra todas las fonéticamente similares del diccionario, y el software decida que la más parecida es una de las calles de Montevideo, se devolverá el nombre de dicha calle correctamente escrito.

Esto es muy importante, ya que si el software de reconocimiento reconoce la calle dictada, devolverá su nombre correctamente escrito ya que pertenece a su diccionario. Y obviamente las palabras del diccionario están correctamente escritas. En caso de que no reconozca la calle dictada, reconocerá alguna palabra o frase corta que sea lo más parecido fonéticamente a lo que el usuario dijo. En este caso, se hace imposible reconocer la palabra como válida, y se devolverá un error por parte del programa.

Vale la pena notar, volviendo a lo comentado en la sección acerca de porqué se eligió el IBM ViaVoice, que si el ViaVoice escribiera directamente lo que oyera, estaríamos en condiciones de tomar esas palabras y aplicarles el método Soundex hasta obtener nombres de calles. Pero, ya que el ViaVoice escribe la palabra más parecida de SU diccionario, y ésta se escribe bien (obviamente cuando se la encuentra), pareciera no tan interesante el uso de Soundex.

Es decir, que contando con una herramienta como el IBM ViaVoice, no se utilizan al máximo los algoritmos de corrección de errores desarrollados para la Versión 2 (los que implementan el método Soundex).

Si se contara con una herramienta que escribiera tal cual lo que está oyendo del usuario, esto daría la posibilidad de tomar esas palabras y pasarlas al programa para que haga un uso extensivo de tales algoritmos de corrección de errores.

Esto muestra que de alguna forma la introducción del software de reconocimiento de voz hace que él mismo solucione ciertos problemas de dictado que otrora los solucionaba el método Soundex. Se reitera que esto NO se da en forma intencional (el ViaVoice no está hecho para corregir nombres de calles mal dichas), sino únicamente porque en el esfuerzo de reconocer lo que el usuario habla, se escribe la palabra que sea más parecida del diccionario.

Lo mismo ocurre con los rubros. Aquí, esta “corrección” que hace el ViaVoice se hace más evidente ya que se trata de palabras del idioma español. Si bien las calles no tienen porqué serlo, los rubros sí lo son. Por lo tanto, el ViaVoice reconoce más fácilmente estos rubros como palabras de su diccionario, devolviéndolo bien escrito.

Claro que aquí se hace absolutamente necesario el procesamiento del que se habla en la sección acerca de consideraciones de rubros, en cuanto a que una vez obtenido un rubro, hay que encontrar su super-rubro asociado, ya que son éstos últimos los que se encuentran almacenados en el mapa. El uso de estos algoritmos es imposible de evitar en este caso.

Soundex: No Está Todo Perdido:

Aunque se haya dicho que la introducción del reconocimiento de voz opacó el previo desarrollo de algoritmos que implementan el método Soundex, no está todo perdido.

Estos algoritmos siguen siendo útiles ya que hacen que la solución sea independiente del software de reconocimiento de voz elegido. Si el día de mañana, se opta por un software comercial de reconocimiento de voz que opere de la forma directa, es decir que escriba tal cual lo que oiga sin obligar a devolver una palabra del diccionario si no la encuentra, entonces éstos algoritmos serán de vital importancia para la robustez del programa.

Asimismo, estos algoritmos que implementan el método Soundex, también hacen que la solución de los prototipos sean lo suficientemente flexible que permita el ingreso de texto escrito por parte del usuario, en casos en los que no se le pueda enviar texto hablado al programa.

No es difícil imaginar tales situaciones: la señal de voz tiene demasiado ruido, el usuario dispone de un terminal del tipo PDA (Personal Digital Assistant) o Handheld (computadora de bolsillo) con radio MODEM pero no con micrófono, en el caso que una operadora atienda ciertas llamadas en forma personalizada, etc.²¹

²¹ Ver Sección 3.6.

A continuación se presenta una breve discusión acerca del impacto sufrido en la operativa del programa que lleva a utilizar correctamente el prototipo.

Para poder hacer funcionar de forma aceptable (que reconozca todas o casi todas las calles) el ViaVoice, se hace necesario “enseñarle” ciertas palabras. Qué palabras? Los nombres de calles que no pertenezcan a su diccionario en idioma español. Por ejemplo, la calle “Juan Paullier” consta de dos palabras, una que si pertenece al diccionario (Juan) y otra que no (Paullier). Por lo tanto, la primera no es necesario enseñársela, pero la segunda si. Aquí estamos implícitamente hablando de dos etapas: primero, cómo saber cuáles palabras sabe y cuáles no sabe; y segundo cómo enseñarle.

Para saber cuáles palabras sabe y cuáles no, utilizamos una herramienta llamada “Analizar mis Documentos”. Mediante esta herramienta, propia del ViaVoice, se le pasa un documento de texto conteniendo una cantidad de palabras, que en nuestro caso son nombres de calles de Montevideo. Luego, el software separa todas las palabras del documento en dos grupos: las que sabe, y las que no sabe. El paso final es mostrarle al usuario todo el conjunto de palabras que no sabe para que el usuario se las “enseñe”.

Para enseñarle las palabras que no sabe, se deben hacer dos cosas: primero, escribir la “palabra fonética”, es decir, escribir en español como suena la palabra. Por ejemplo, en el caso de “Paullier”, la palabra fonética sería “Polié”. Esto le permite al ViaVoice tener una noción de qué sonidos (fonemas) esperar para tal palabra. Segundo, se debe decir en voz alta la palabra nueva para que el ViaVoice la grabe. De esta manera, ya sabe cómo ha de escribirla (tal cual como estaba en el documento de texto) y ya sabe como se dice (dada la misma palabra en su forma fonética y la grabación de la voz del locutor diciéndola).

Este procedimiento debe repetirse para cada palabra nueva en su diccionario, ya que de no hacerlo, el ViaVoice nunca devolverá la palabra, por no pertenecer a su diccionario.

Obviamente que esto implica tener que gastar un tiempo considerable en “enseñarle” al software como debe escuchar los nombres de las calles de Montevideo, pero es absolutamente necesario. También es justo decir que una buena cantidad de palabras ya las conocerá, porque están en su diccionario.

De todos modos, éste es un gran problema, aunque tal vez el mayor sea la dependencia del locutor. Es posible imaginar a un empleado de una empresa operadora de telefonía celular “enseñarle” una vez al programa los nombres de las calles, y que luego, mediante un reconocedor independiente del locutor, los usuarios lo utilicen con éxito.

Lo que sí es difícil de imaginar, es que cada usuario del sistema le “enseñe” todas las calles al programa.

7. Testeo del Prototipo

7.1 Introducción

Este Capítulo mostrará los diferentes testeos efectuados sobre el prototipo desarrollado en el presente trabajo, el cual ha sido extensamente examinado en el Capítulo 6.

Cabe destacar que lo único que interesará testear de manera exhaustiva, como lo indica el título del Capítulo, son las funcionalidades del prototipo, no así la documentación del presente proyecto ni los métodos o metodologías utilizadas.

Además, habiéndose buscado normas para seguir en el presente testeo, el autor prefirió seguir su propia motivación, por no encontrar ninguna norma o guía a su criterio adecuada a las circunstancias. El presente testeo representa en forma efectiva todas y cada una de las características que se pretendían mostrar y analizar del prototipo.

La forma de abordar el testeo ha sido la siguiente: primero se encara el testeo de la Versión 1, luego de la Versión 2 y finalmente de la Versión 3. Para cada versión, se estudia de forma separada cada funcionalidad agregada por esa versión. El motivo de tal separación es que, como se dijo con anterioridad, el prototipo fue desarrollado de forma incremental, y para abordar las funcionalidades más avanzadas es necesario estudiar y comprender (y testear) las funcionalidades que las precedieron en sus versiones anteriores.

Lo que sigue es una descripción de cada uno de los ítems analizados para cada funcionalidad de cada versión del prototipo.

Funcionalidad:	Nombre de la Funcionalidad Testeada (ícono)
Descripción:	Descripción de la funcionalidad testeada.
Parámetros:	Todos los datos requeridos por el usuario para poder hacer andar la funcionalidad.
Precondición:	Que cosas se deben hacer antes para poder luego ejecutar la funcionalidad.
Testeo Realizado:	Que tests se hizo sobre la funcionalidad. Cada uno estará numerado, y la misma numeración será utilizada para los Resultados Esperados y los Resultados Obtenidos.
Resultado Esperado:	Que resultado es de esperarse para cada una de las pruebas descritas en el punto anterior.
Resultado Obtenido:	Que resultado efectivamente mostró el prototipo cuando se ejecutó el test correspondiente descrito en Testeo Realizado.
Errores:	Que errores se descubrieron al momento de efectuar los tests.
Conclusión Final:	Las conclusiones surgidas de contrastar los Resultados Esperados con los Resultados Obtenidos, y a la vista de los Errores que aparecieron.
Observaciones:	Observaciones interesantes o comentarios generales que no tenían cabida específica en los puntos anteriores o que los complicaban demasiado.


Para todos los testeos, la plataforma de software/hardware utilizada fue la siguiente:

Sistema Operativo:	Microsoft Windows 98 (release 2)
Procesador:	INTEL Pentium III 500 Mhz
Memoria:	256 MB
Monitor:	Color 14"



Solo se ha utilizado esta plataforma ya que en ningún momento apareció la necesidad o requerimiento de que el prototipo fuese portable, y por los motivos expuestos en el Capítulo 6, la plataforma más adecuada es la elegida.

7.2 Testeo de la Versión 1


7.2.1 Cargar Capa

Funcionalidad:	Cargar Capa 
Descripción:	Esta funcionalidad permite cargar una capa (también llamada Shapefile o Layer). Cabe recordar que las tres capas con las cuales trabaja el Prototipo son: <i>Ejes</i> (conteniendo las calles), <i>Direcciones</i> (conteniendo los números de puerta sobre determinadas calles) y <i>Comercios</i> (conteniendo los comercios, su ubicación y la rama o rubro al cual pertenece).
Parámetros:	Nombre y ubicación del archivo que contiene la capa a cargar. Este parámetro es ingresado por el usuario a través de un cuadro de diálogo de Windows que permite en forma gráfica, navegar por las unidades del equipo hasta seleccionar con el mouse el archivo correspondiente (con extensión .shp –de shapefile).
Precondición:	Ninguna.
Testeo Realizado:	Se cargaron varias capas, no solo las tres mencionadas (referidas a Montevideo), sino también capas provenientes de los Samples (ejemplos) del MapObjects. Además, se modificó el orden en el cual las mismas eran cargadas. Obviamente, se deben cargar capas que estén referidas a una misma geografía, no pudiendo cargar capas de Montevideo junto con capas de EEUU.
Resultado Esperado:	Se espera que la capa cargada se coloque encima de las demás capas ya cargadas en el mapa, mostrándose únicamente las formas contenidas en la capa (ya sean puntos, líneas o polígonos), pero no así los datos de la capa (ver funcionalidad Etiquetar Capa).
Resultado Obtenido:	Efectivamente se obtiene, luego de aplicada la funcionalidad, la capa seleccionada por encima de las ya cargadas y sin desplegar datos fuera de las formas que ésta contiene.
Errores:	<ul style="list-style-type: none"> Se permite cargar nuevamente una capa ya cargada. Esto no debería suceder. El programa debería corroborar, antes de cargar una nueva capa, si ya no está cargada. Se considera éste un error secundario.
Conclusión Final:	La funcionalidad cubre las expectativas, se desempeña correctamente y se considera despreciable el error encontrado.
Observaciones:	<ul style="list-style-type: none"> No es permitido “descargar” una capa. Es decir, que una vez cargadas, las mismas permanecen así hasta que se cierre el programa. Cuando se carga una capa, por defecto se la muestra toda. Es decir, que se le efectúa un Full Extent (ver funcionalidad Full Extent). Cada capa cargada se muestra en la Lista de Capas Cargadas. Por defecto, cada nueva capa cargada no aparece etiquetada (ver funcionalidad Etiquetar Capa / Quitar Etiquetas).


7.2.2 Zoom In / Zoom Out

Funcionalidad:	Zoom In (acercar)  / Zoom Out (alejarse) 
Descripción:	Realiza un zoom sobre una sección del mapa. Debe quedar claro que el zoom es sobre todas las capas del mapa, y no sobre una en particular. Esto se aplica tanto al zoom para acercar (in) como para alejar (out).
Parámetros:	Ninguno.
Precondición:	Haber cargado alguna capa.
Testeo Realizado:	Se dibujaron diferentes rectángulos con el mouse esperando que se efectuara un zoom sobre el área demarcada. Para el zoom out basta con hacer un clic sin la necesidad de trazar un rectángulo.
Resultado Esperado:	Se espera que el área marcada en el rectángulo sea sobre la cual se efectúa el zoom in, y para el caso del zoom out, se espera que se efectúa un “alejamiento” del mapa tomando como centro el punto donde se hizo el clic.
Resultado Obtenido:	Efectivamente se obtiene, luego de seleccionado un rectángulo (zoom in) o un punto (zoom out) el acercamiento o alejamiento deseado.
Errores:	Ninguno observado.
Conclusión Final:	La funcionalidad cubre las expectativas, y se desempeña correctamente.
Observaciones:	Ninguna.



7.2.3 Pan

Funcionalidad:	Pan (desplazamiento) 
Descripción:	Esta funcionalidad le permite al usuario desplazarse dentro del mapa. Es decir, “mover” el mapa dentro del área asignada para mostrarlo. Esta funcionalidad es de especial interés cuando se efectúan zooms de acercamiento sobre el mapa, ya que el área destinada al mapa no será suficiente para mostrar todo el contenido del mapa ahora que es “mas grande”.
Parámetros:	Ninguno.
Precondición:	Haber cargado alguna capa.
Testeo Realizado:	Luego de seleccionado el botón de desplazamiento, y contando con cierto zoom sobre el mapa, se arrastró con el mouse el mismo.
Resultado Esperado:	Se esperaba que su contenido fuera desplazado al ritmo del movimiento de arrastre del ratón.
Resultado Obtenido:	Efectivamente se obtuvo el desplazamiento del mismo.
Errores:	Ninguno observado.
Conclusión Final:	La funcionalidad cubre las expectativas, y se desempeña correctamente.
Observaciones:	Ninguna.

7.2.4 Full Extent

Funcionalidad:	Full Extent (extensión total) 
Descripción:	Esta funcionalidad permite mostrar el mapa en su totalidad dentro del área destinada.
Parámetros:	Ninguno.
Precondición:	Haber cargado alguna capa.
Testeo Realizado:	Se oprimió el botón de Full Extent (o extensión total).
Resultado Esperado:	Se esperaba ver la totalidad del mapa, es decir, de todas las capas que lo componen.
Resultado Obtenido:	Efectivamente se observó todo el mapa al efectuar el testeo.
Errores:	Ninguno observado.
Conclusión Final:	La funcionalidad cubre las expectativas, y se desempeña correctamente.
Observaciones:	<ul style="list-style-type: none"> • Cuando se hace un full extent, todas las capas pierden sus etiquetas.

7.2.5 Etiquetar Capa / Quitar Etiquetas

Funcionalidad:	Etiquetar Capa  / Quitar Etiquetas 
Descripción:	Esta funcionalidad permite etiquetar la capa seleccionada de la lista de capas cargadas con la etiqueta seleccionada de la lista de etiquetas disponibles para la capa seleccionada. El concepto de “etiquetar” implica mostrar ciertos datos sobre una capa en particular. Por ejemplo, la capa <i>Ejes</i> permite etiquetarla de dos maneras: mostrando los nombres de las calles, o mostrando el código de cada calle. Cada dato es una etiqueta, en el entendido que cuando se etiqueta una capa, cada elemento de la misma muestra una etiqueta de texto indicando el valor del dato para ese elemento (cada línea que representa una calle muestra una etiqueta de texto mostrando el nombre de cada una). Además, la funcionalidad análoga de Quitar Etiquetas simplemente quita la etiqueta seleccionada de la capa elegida, como es de esperar.
Parámetros:	La capa a etiquetar (elegida de la lista de capas cargadas) y la etiqueta (dato) a colocar (elegido de la lista de etiquetas para la capa seleccionada).
Precondición:	Haber cargado alguna capa.
Testeo Realizado:	Se seleccionó una capa y luego una etiqueta.
Resultado Esperado:	Se esperaba ver en el mapa un conjunto de etiquetas mostrando el dato elegido para la capa seleccionada.
Resultado Obtenido:	Efectivamente se observó, luego de elegir la capa y la etiqueta, todos los valores para esa capa.
Errores:	Ninguno observado.
Conclusión Final:	La funcionalidad cubre las expectativas, y se desempeña correctamente.
Observaciones:	<ul style="list-style-type: none"> • Sólo puede haber una etiqueta por capa a la vez. • Se permite etiquetar la cantidad de capas que el usuario desee.

7.2.6 Búsqueda por Intersección

Funcionalidad:	Búsqueda por Intersección Buscar por Intersección
Descripción:	Esta funcionalidad le permite al usuario ingresar los nombres de dos calles, en forma independiente (léase en forma estructurada), y poder visualizar en el mapa la intersección de ambas (si así ocurre). Además, se efectuará un zoom hacia el punto de intersección, y un desplazamiento (pan) tomando como centro tal punto. El objetivo de esto es poder “enfocar” correctamente el punto en cuestión de manera de ubicarlo visualmente, sin mostrar todas las calles, lo cual crea confusión y obliga al usuario a buscar detenidamente el punto de corte.
Parámetros:	Nombre de la primer y segunda calle. Estos parámetros son ingresados por el usuario a través de dos cuadros de diálogo independientes.
Precondición:	Haber cargado la capa <i>Ejes</i> primero.
Testeo Realizado:	<ol style="list-style-type: none"> 1. Se seleccionó el botón “Buscar por Intersección”, y luego, en forma separada y una después de la otra, los nombres de las dos calles de las cuales se pretende ubicar su intersección. 2. Se ingresaron nombres de calles que no estaban en las capas, así como nombres de calles mal escritos (con errores ortográficos). 3. Se ingresaron calles que no se cortan.
Resultado Esperado:	<ol style="list-style-type: none"> 1. Se espera ver en forma visual en el mapa, la intersección de ambas calles ingresadas (si así corresponde), 2. Un mensaje de error indicando que alguna (o ambas) no existe, o 3. Un mensaje de error indicando que ambas no se cortan. <p>A su vez, se espera ver en un color una de las calles (azul), y en otro la otra calle (rojo). La intersección de ambas (la cual es un punto en el mapa) se mostrará en color amarillo.</p>
Resultado Obtenido:	<ol style="list-style-type: none"> 1. Para el caso correcto, en el cual se ingresaron dos nombres de calles exactamente igual a como aparecen en el mapa, y que efectivamente se cortan, el resultado fue el esperado, mostrándose en el mapa tanto las dos calles en sus colores, como la intersección en amarillo. 2. Para el caso en que una (o ambas) calle no existe, es decir que no pertenece a la porción de Montevideo representada por el mapa, o que está mal escrita, se mostró un correcto mensaje de error. No se distingue entre estos dos casos. 3. Para el caso en que ambas calles existen, están bien escritas, pero no se cortan, también se mostró un mensaje de error indicando tal situación.
Errores:	<ul style="list-style-type: none"> • El zoom y pan efectuado cuando las calles efectivamente se cortan, no parece funcionar correctamente en la totalidad de los casos. En particular, cuando ya existe cierto zoom sobre el mapa, esta funcionalidad, en ocasiones, hace demasiado zoom, lo que requiere que el usuario manualmente haga zoom outs para poder apreciar el área. Se considera éste un error secundario, aunque en ocasiones resulte algo molesto.
Conclusión Final:	La funcionalidad cubre las expectativas, se desempeña correctamente y se considera despreciable el error encontrado.
Observaciones:	<ul style="list-style-type: none"> • Es de particular interés notar que los nombres de las calles deben ser escritos exactamente igual a como están ingresados dentro de los Shapefiles, o sea, a como se muestran en el mapa. De no ser así, el programa dirá que la calle no existe. Este comportamiento es correcto pues para la primer versión del prototipo no se consideró un método de corrección de errores en la entrada del usuario. La única diferencia que puede existir, es el uso o no de mayúsculas para la primer letra del nombre de la calle. • El resultado (la intersección) permanece visible hasta la próxima búsqueda, ya sea por Intersección o por Número (ver funcionalidad Búsqueda por Número). • Si no se encuentran capas cargadas, se muestra un mensaje de error correspondiente. No se entiende que esto sea un error, ya que evidentemente para poder utilizar la presente funcionalidad, se debe haber cargado la capa <i>Ejes</i>, como se establece en las Precondiciones.

7.2.7 Búsqueda por Número

Funcionalidad:	Búsqueda por Número Buscar por Número
Descripción:	Esta funcionalidad le permite al usuario ubicar un número de puerta específico dentro de una determinada calle perteneciente al mapa. Dado que los números de puerta pertenecen al Shapefile <i>Direcciones</i> , y los mismos se representan como puntos en el mapa, el resultado aparecerá como un punto en color amarillo, haciendo un zoom y pan tomando como centro tal punto encontrado.
Parámetros:	Nombre de la calle y número de puerta, ingresados en dos cuadros de diálogo separados.
Precondición:	Haber cargado la capa <i>Ejes</i> y la capa <i>Direcciones</i> primero.
Testeo Realizado:	<ol style="list-style-type: none"> 1. Se seleccionó el botón “Buscar por Número”, y luego, en forma separada, se ingresó el nombre de la calle y el número de puerta. 2. Se ingresó un nombre de calle que no estaba dentro del mapa, o que estaba escrito incorrectamente (error ortográfico), los cuales en esta versión no son corregidos. 3. Se ingresó un número de puerta que no existe sobre la calle ingresada. 4. Se ingresó una calle que si bien pertenece al mapa, no contiene información de números de puerta, ya que la información de éstos es sobre ciertas calles.
Resultado Esperado:	<ol style="list-style-type: none"> 1. Se espera ver, en forma visual en el mapa, el resultado de la búsqueda. Es decir, un punto de color amarillo centrado y con cierto zoom, que se corresponda con el número de puerta deseado sobre la calle deseada. 2. Se espera un mensaje de error indicando que la calle no existe. 3. Se espera un mensaje de error indicando que el número de puerta no existe. 4. Se espera un mensaje de error indicando que la calle ingresada no contiene información de números de puerta.
Resultado Obtenido:	<ol style="list-style-type: none"> 1. Para el caso en que la calle existe, y contiene información de números de puerta, y que el mismo ingresado pertenece a la calle, efectivamente se muestra el punto color amarillo en el mapa. 2. Para el caso en que la calle ingresada no existe, o está mal escrita, efectivamente se muestra un mensaje de error mostrando esta condición. 3. Para el caso en que el número de puerta no existe dentro de la calle ingresada, efectivamente se muestra un mensaje de error mostrando esta condición. 4. Para el caso en que la calle ingresada no contiene información de números de puerta, también se muestra un mensaje de error mostrando esta condición.
Errores:	<ul style="list-style-type: none"> • Ocurre el mismo error que para el caso de la Búsqueda por Intersección, que involucra al zoom y al pan efectuado sobre el resultado de la búsqueda. Cuando el mapa se encuentra en Full Extent, estos funcionan correctamente, pero cuando posee cierto zoom, no siempre lo hace.
Conclusión Final:	La funcionalidad cubre las expectativas, se desempeña correctamente y se considera despreciable el error encontrado.
Observaciones:	<ul style="list-style-type: none"> • La calle no es coloreada, solo el número de puerta buscado. • Se aplica la misma observación con respecto a los nombres de las calles, los cuales deben ser ingresados perfectamente bien, sin ningún error ortográfico, ya que no se realiza ninguna comprobación en este sentido. • El resultado (el número de puerta) permanece visible hasta la próxima búsqueda, ya sea por Intersección o por Número. • La necesidad de cargar ambas capas como precondición, se debe a que, como se mencionó en el Capítulo 6, solo ambas capas juntas son capaces de tener el nombre literal de la calle junto con los números de puerta contenidos.

7.3 Testeo de la Versión 2

7.3.1 Función de Localización

Funcionalidad:	Función de Localización Función de Localización
Descripción:	<p>Esta funcionalidad le permite al usuario escribir un texto conteniendo una dirección en formato no estructurado para luego ubicarla en el mapa. Con “formato no estructurado” nos referimos a:</p> <ul style="list-style-type: none"> • poder localizar tanto una intersección de calles como un número de puerta dentro de una calle (por ejemplo: “san salvador esquina juan paullier” o “san salvador 1234”); • poder ingresar los dos nombres de las calles (para el caso de buscar una intersección) o la calle y el número (para el caso de buscar un número de puerta) en el <u>mismo cuadro de diálogo</u>, y específicamente en el mismo cuadro de texto. Es decir, que el usuario escribe todo junto en un mismo lugar, sin demasiada estructura; • poder escribir una frase en idioma español, con cierta intención e incluir palabras que no serán tomadas por el programa (por ejemplo: “<u>quiero ir a san salvador</u> esquina juan paullier”). <p>A su vez, se cuenta con un método de corrección de los errores introducidos en los nombres de calles que el usuario ingresa. Este es el método Soundex descrito en detalle en el Capítulo 6. Por ejemplo, si el usuario escribe “quiero ir a juan paullie” el programa encontrará la calle <u>Juan Paullier</u>, aunque se haya ingresado incorrectamente.</p> <p>Dado que no correspondería repetir lo discutido antes, vale la pena remitirse al Capítulo 6 para ver que tipos de errores corrige el método y cuales no, ya que se basa fuertemente en la primer letra y en las tres consonantes que le siguen. Se dice esto ya que no sería posible considerar todos estos errores dentro del presente testeo.</p> <p>Se debe tener presente que se trata en esta ocasión, de la primer versión de esta funcionalidad, la cual permite el ingreso de texto por parte del usuario para ser parseado por el programa. La segunda versión de esta misma funcionalidad, permite el ingreso hablado de texto mediante el dictado de una frase al programa (ver Sección 7.4.1).</p>
Parámetros:	Ingreso de un texto escrito por parte del usuario en el único cuadro de diálogo disponible para esto.
Precondición:	Haber cargado la capa <i>Ejes</i> y la capa <i>Direcciones</i> primero.
Testeo Realizado:	<p>Se efectuaron los siguientes testeos con el fin de abarcar la totalidad de las posibilidades de error con las que se puede encontrar el programa. Para todas ellas, se seleccionó el botón “Función de Localización” y luego se ingresaron los mismos textos que aparecen entre comillas.</p> <ol style="list-style-type: none"> 1. ingresar una intersección de calles correcta (“quiero ir a mercedes esquina minas”); 2. ingresar una calle y un número de puerta dentro de esa calle correcto (“quiero ir a durazno 1402”); 3. ingresar una calle que no exista (“quiero ir a micallecita 1234”); 4. ingresar una intersección que no sea tal, es decir que las calles no se corten (“quiero ir a canelones esquina uruguay”); 5. ingresar cualquier texto que no incluya ni una intersección ni un número de puerta dentro de una calle (“quiero comer una milanesa en dos panes”); 6. ingresar una calle mal escrita, es decir, cometiendo un error ortográfico que: <ol style="list-style-type: none"> a. sea detectado por el método de corrección (“quiero ir a mersedez esquina minaz”), b. no sea detectado por el método de corrección (“quiero ir a percedes esquina minas”).

Resultado Esperado:	<ol style="list-style-type: none"> 1. Se espera ver en forma visual en el mapa, la intersección de ambas calles ingresadas; 2. Se espera ver, en forma visual en el mapa, el resultado de la búsqueda. Es decir, un punto de color amarillo centrado y con cierto zoom, que se corresponda con el número de puerta deseado sobre la calle deseada; 3. Un mensaje de error indicando que la calle no existe; 4. Un mensaje de error indicando que ambas no se cortan; 5. Un mensaje de error indicando que no se encontró ni una intersección ni un número de puerta dentro de una calle; 6. Se espera que: <ol style="list-style-type: none"> a. El error sea detectado y corregido por el método de corrección (que interprete mersedez como Mercedes), b. El error no sea detectado ya que no se incluyen formas de detectar tipos de errores fuera del método Soundex.
Resultado Obtenido:	<ol style="list-style-type: none"> 1. Para el caso correcto, en el cual se ingresaron dos nombres de calles exactamente igual a como aparecen en el mapa, y que efectivamente se cortan, el resultado fue el esperado, mostrándose en el mapa tanto las dos calles en sus colores, como la intersección en amarillo; 2. Para el caso en que la calle existe, y contiene información de números de puerta, y que el mismo ingresado pertenece a la calle, efectivamente se muestra el punto color amarillo en el mapa; 3. Para el caso en que la calle ingresada no existe, o está mal escrita, efectivamente se muestra un mensaje de error mostrando esta condición; 4. Para el caso en que ambas calles existen, están bien escritas (o fueron corregidas), pero no se cortan, también se mostró un mensaje de error indicando tal situación; 5. Para el caso en que no se provee de información útil, se muestra un mensaje que indica que no se encontró ninguna intersección ni tampoco un número de puerta; 6. Se observó: <ol style="list-style-type: none"> a. Que el error efectivamente era detectado y corregido, b. Que el error no era detectado ni corregido (como se sabía de antemano).
Errores:	<ul style="list-style-type: none"> • Cuando se ingresa una calle que internamente tiene el mismo código Soundex que otra (¡como es el caso de Cuareim-C650- y Soriano-S650-!) si los nombres ingresados son correctos, es decir sin faltas de ortografía (“quiero ir cuareim esquina colonia”) entonces no hay error. Pero, si se comete un error en el nombre de alguna calle que tenga el mismo código que otra (“quiero ir a zoriamo y colonia”, en este caso el error se cometio en la calle Soriano, que tenía el mismo código que Cuareim), entonces el programa, en lugar de decir que Soriano y Colonia no se cortan, muestra el corte de Cuareim con Colonia!, ya que entiende que a “zoriamo” como Cuareim y no como Soriano, porque tienen el mismo código. Esto solo ocurre cuando se escribe mal el nombre de una calle que tenga otra con el mismo código (como ocurre con Soriano). El motivo es que cuando se usan los nombres correctos, la primera vez que el programa busca la calle, lo hace con el mismo nombre que el usuario ingresa (es una mejora del Soundex), y recién cuando no lo encuentra tal cual, utiliza el método Soundex para buscarlo. Cuando lo usa, como los dos tienen el mismo código asociado, encuentra el de la otra calle primero pues es la primera con la que se topa el algoritmo. • Mismo error de zoom y pan descrito anteriormente.
Conclusión Final:	<p>Considerando los resultados obtenidos con los esperados, y tomando en cuenta el error detectado, se concluye que en términos generales la funcionalidad cumple correctamente su objetivo. No obstante, el error indicado no puede ser pasado por alto, ya que muestra una clara falla del método de corrección utilizado o de su implementación (su puesta en implementación).</p>
Observaciones:	<ul style="list-style-type: none"> • Se observó aquí una deficiencia importante del método de corrección, o bien de la forma de implementarlo. Si bien tal vez no son muchas las calles con el mismo código asociado, de ninguna manera pueda esperarse que tal problema persista si se tiene en mente la realización de un producto final.

7.3.2 Función de Cercanías

Funcionalidad:	Función de Cercanías Función de Cercanías
Descripción:	<p>Esta funcionalidad le permite al usuario obtener un conjunto de comercios de cierto rubro que éste especifique a una cierta distancia de él. El ingreso del rubro como de la distancia, se hacen de forma “estructurada”, es decir, que primero se ingresa el rubro, solo, en un cuadro de diálogo, y luego, en otro, se ingresa la distancia la cual se espera en metros.</p> <p>Por lo tanto, no se permite el ingreso de una frase del tipo “quiero las farmacias a 300m”, sino que tal búsqueda debe ser en dos etapas: primero ingresar el rubro (farmacia) y luego la distancia máxima en metros de la que se desea obtener comercios (300).</p> <p>Para poder entender el rubro que el usuario ingresa, es decir, para poder finalmente corresponderlo con un super-rubro almacenado en el Shapefile <i>Comercios</i>, como se discutió en el Capítulo 6, se efectúan dos etapas:</p> <ol style="list-style-type: none"> 1. se aplica el método de corrección de errores visto anteriormente sobre el rubro ingresado por el usuario intentando ver si se corresponde con algún rubro almacenado no en el Shapefile sino en el Diccionario de Rubros. Allí se almacenaba cada super-rubro junto con un grupo de rubros relacionados que apuntaban a éste, es decir, posibles formas de referirse al super-rubro (recordar que se desglosaba el super-rubro por extensión). Por esto es que primero se verifica a través del método Soundex si lo que el usuario ingresó se corresponde con alguno de esos rubros. 2. cuando efectivamente se confirma que se encontró en el diccionario un rubro que se corresponde con lo que el usuario ingresó, se busca el super-rubro asociado, ya que así será como aparece en el mapa y en el Shapefile <i>Comercios</i>. <p>El resultado es mostrado como un conjunto de puntos (ya que el Shapefile <i>Comercios</i> sólo contiene puntos) en color amarillo, quienes representan los comercios pertenecientes al rubro ingresado por el usuario dentro de la distancia ingresada.</p>
Parámetros:	<p>Ingresar el rubro de los comercios buscados, así como la distancia a la ubicación actual del usuario que tolera como máxima. Más allá de esa distancia no se buscarán comercios.</p>
Precondición:	<p>Haber efectuado una búsqueda, para que así el programa tome esa ubicación como la actual del usuario.</p>
Testeo Realizado:	<p>Los siguientes testeos fueron efectuados utilizando el botón “Función de Cercanías”, y luego de haber obtenido como resultado una ubicación, ya sea mediante el uso de la Función de Localización o mediante las búsquedas (por Intersección o por Número vistas en la Versión 1).</p> <ol style="list-style-type: none"> 1. se ingresó un rubro bien escrito (por ejemplo “bar”) junto con una distancia; 2. se ingresó un rubro que no existe (por ejemplo “casa de computación”) 3. se ingresó un rubro mal escrito, tal que: <ol style="list-style-type: none"> a. sea detectado por el método de corrección (por ejemplo “var”), b. no sea detectado por el método de corrección (por ejemplo “nar” suponiendo que se quiso decir “bar” pero se oprimió la tecla contigua). 4. se repitió el testeo 1. pero esta vez con otra distancia, para corroborar que la misma esté siendo tenida en cuenta.
Resultado Esperado:	<ol style="list-style-type: none"> 1. Se espera obtener como resultado un conjunto de puntos en color amarillo representando los comercios del rubro ingresado a una distancia igual o menor a la ingresada; 2. Se espera obtener un mensaje de error indicando que no existe tal rubro; 3. Se espera: <ol style="list-style-type: none"> a. que el error sea corregido, por lo tanto efectuándose la búsqueda correctamente y con normalidad, b. que el error no sea encontrado ya que no está incluido en los errores corregidos por el método de corrección seleccionado. 4. Se espera obtener el mismo conjunto del punto 1. pero con más o menos comercios, dependiendo si la distancia especificada es mayor o menor a la original. Es decir, que se pretende ver cambios en la cantidad de comercios arrojados como resultado.

Resultado Obtenido:	<ol style="list-style-type: none"> 1. Se obtuvo efectivamente un conjunto de puntos en color amarillo representando los comercios buscados; 2. Se obtuvo un mensaje de error indicando que no existía el rubro ingresado; 3. Se obtuvo: <ol style="list-style-type: none"> a. Un conjunto de puntos representando los comercios cuyo rubro es el corregido del ingresado por el usuario, b. Un mensaje de error indicando que el rubro ingresado no existe, ya que como se esperaba, el error no fue detectado, por lo que no fue corregido. 4. Se obtuvo un conjunto similar al del punto 1. pero con una cantidad menor o mayor, según las pruebas con diferentes distancias.
Errores:	<ul style="list-style-type: none"> • Mismo error de zoom y pan descrito anteriormente.
Conclusión Final:	Se concluye que la funcionalidad operó con éxito, arrojando el resultado esperado en la totalidad de los testeos efectuados.
Observaciones:	<ul style="list-style-type: none"> • El mayor error encontrado para el método de corrección era cuando existían dos calles con el mismo código, y eso no ocurre en esta funcionalidad, ya que para cuando se la utiliza, se tuvo que haber efectuado una búsqueda previamente. Por lo tanto, el error no se aplica a esta funcionalidad. • Vale la pena observar que los rubros que se muestran cuando se etiqueta el Shapefile <i>Comercios</i>, son en realidad los originales, los que denominamos super-rubros, y no los rubros extendidos que puede utilizar el usuario. El motivo es que las etiquetas de la capa se toman directamente del Shapefile, y no del diccionario conteniendo los rubros definidos individualmente.

7.4 Testeo de la Versión 3

7.4.1 Función de Localización

Funcionalidad:	Función de Localización	Función de Localización
Descripción:	<p>Esta función es la misma que la correspondiente de la Versión 2, con la diferencia que en esta tercer versión el usuario le dicta directamente al programa con la ayuda del reconocedor de voz utilizado.</p> <p>En particular, será interesante ver cómo la introducción del reconocedor de voz modifica (o no) el testeo efectuado sobre la <u>misma</u> función de localización, pero ahora con una entrada de voz.</p>	
Parámetros:	Se debe dictar al micrófono del equipo la frase conteniendo la dirección no estructurada que se desea ubicar en el mapa.	
Precondición:	Haber cargado la capa <i>Ejes</i> y la capa <i>Direcciones</i> primero.	
Testeo Realizado:	<p>Se efectuaron testeos muy similares a los efectuados para la función de localización de la Versión 2, con la diferencia de que en lugar de ingresar un error ortográfico venido tanto de la ignorancia de cómo escribir el nombre de la calle como de un error al escribirla; ahora se cuenta con la ignorancia de cómo decir el nombre de la calle, pero no de cómo escribirlo.</p> <p>Por ejemplo, como se comentó en el Capítulo 6, no es de esperarse que el usuario cometa errores de tipeo, ya que no escribirá nada! En lugar de escribir “nar” en lugar de “bar” porque la letra N es contigua a la letra B, podrá pronunciar “var” en lugar de “bar”. Este tipo de diferencias es a la que nos referimos como errores típicos de un dictado, ya no de un tipeo.</p> <p>A su vez, es de extrema importancia recordar lo discutido en el Capítulo 6 con respecto a la vigencia o validez del método Soundex cuando se introdujo el reconocedor de voz. Recordar que ahora es el reconocedor quien “corrige” los errores pronunciados, ya que nunca escribirá nada que no pertenezca a su diccionario. Por lo tanto, una calle como “mercedes” será interpretada como la calle Mercedes, pero no por el método de corrección, sino por el propio reconocedor.</p> <p>Los testeos efectuados fueron los siguientes (compararlos con los de la Sección 7.3.1):</p> <ol style="list-style-type: none"> 1. dictar una intersección de calles correcta (“quiero ir a mercedes esquina minas”); 2. dictar una calle y un número de puerta dentro de esa calle correcto (“quiero ir a durazno 1402”); 	

	<ol style="list-style-type: none"> 3. dictar una calle que no exista (“quiero ir a micallecita 1234”); 4. dictar una intersección que no sea tal, es decir que las calles no se corten (“quiero ir a canelones esquina uruguay”); 5. dictar cualquier texto que no incluya ni una intersección ni un número de puerta dentro de una calle (“quiero comer una milanesa en dos panes”); 6. dictar una calle mal pronunciada, es decir, cometiendo un error que: <ol style="list-style-type: none"> a. sea detectado por el reconocedor de voz (“quiero ir a mercedes esquina minaz”), b. no sea detectado por el reconocedor de voz (“quiero ir a percedes esquina minas”).
Resultado Esperado:	<ol style="list-style-type: none"> 1. Se espera ver en forma visual en el mapa, la intersección de ambas calles dictadas; 2. Se espera ver, en forma visual en el mapa, el resultado de la búsqueda. Es decir, un punto de color amarillo centrado y con cierto zoom, que se corresponda con el número de puerta deseado sobre la calle deseada; 3. Un mensaje de error indicando que la calle no existe; 4. Un mensaje de error indicando que ambas no se cortan; 5. Un mensaje de error indicando que no se encontró ni una intersección ni un número de puerta dentro de una calle; 6. Se espera que: <ol style="list-style-type: none"> c. El error sea detectado y corregido por el reconocedor (que interprete mercedes como Mercedes), d. El error no sea detectado por el reconocedor y que reconozca otra palabra válida del diccionario pero que no sea la deseada.
Resultado Obtenido:	<ol style="list-style-type: none"> 1. Para el caso correcto, en el cual se dictaron dos nombres de calles exactamente igual a como aparecen en el mapa, y que efectivamente se cortan, el resultado fue el esperado, mostrándose en el mapa tanto las dos calles en sus colores, como la intersección en amarillo; 2. Para el caso en que la calle existe, y contiene información de números de puerta, y que el mismo ingresado pertenece a la calle, efectivamente se muestra el punto color amarillo en el mapa; 3. Para el caso en que la calle dictada no existe, o está mal escrita, efectivamente se muestra un mensaje de error mostrando esta condición; 4. Para el caso en que ambas calles existen, están bien escritas, pero no se cortan, también se mostró un mensaje de error indicando tal situación; 5. Para el caso en que no se provee de información útil, se muestra un mensaje que indica que no se encontró ninguna intersección ni tampoco un número de puerta; 6. Se observó: <ol style="list-style-type: none"> a. Que el error efectivamente era detectado y corregido por el reconocedor, b. Que el error no era detectado ni corregido (como se sabía de antemano). Esto dará como resultado que se escriba una calle que no existe, lo cual debe dar un mensaje de error indicando que la calle no existe.
Errores:	<ul style="list-style-type: none"> • Mismo error de zoom y pan descrito anteriormente.
Conclusión Final:	Se concluye que la funcionalidad operó con éxito, arrojando el resultado esperado en la totalidad de los testeos efectuados.
Observaciones:	<ul style="list-style-type: none"> • La característica mas observable de la presente funcionalidad, es el hecho de que los errores, cuando son corregidos, lo son por el reconocedor de voz y no por el método de corrección. Entonces, cuando el usuario dicta el nombre de una calle y el reconocedor no es capaz de traducirlo correctamente, escribirá la palabra o frase que esté en su diccionario que sea más similar a lo que el usuario dijo. Por ejemplo, para el caso de la calle Mercedes mal dictada, tal vez el reconocedor escriba cosas tales como “me parece”, lo cual luego, aplicándosele el método Soundex no encontrará ninguna calle con ese nombre. Por lo tanto, arrojará un mensaje de calle no existente. • Una observación menos y más sutil, es la siguiente: imaginemos el ejemplo recién puesto con la calle Mercedes. Imaginemos que el usuario dicta esa calle al sistema, pero que el reconocedor escribe “me parece”, por sonar acústicamente similar, y que el usuario no se de cuenta de este error. Entonces, ahora imaginemos por último que el código asociado a “me parece” justo coincide con el código de una calle almacenada en el diccionario de calles! Esto daría como consecuencia que se identificara una calle que el usuario nunca dijo!, sino que por el contrario, fue producto del error de percepción del reconocedor de voz. Dada la dificultad de inventar tal caso, es que no se lo ha puesto como un error, pero bien podría ser considerado como tal.

7.4.2 Función de Cercanías

Funcionalidad:	Función de Cercanías Función de Cercanías
Descripción:	Esta funcionalidad es la misma que para la Versión 2, con la diferencia de que el usuario ingresa tanto el rubro como la distancia en forma hablada, dictándole al micrófono del equipo.
Parámetros:	Dictar el rubro de los comercios buscados, así como la distancia a la ubicación actual del usuario que tolera como máxima, de forma separada uno del otro.
Precondición:	Haber efectuado una búsqueda, para que así el programa tome esa ubicación como la actual del usuario.
Testeo Realizado:	<p>Se efectuaron testeos muy similares a los efectuados en la Sección 7.3.2, por tratarse de la misma funcionalidad, pero ahora de forma hablada.</p> <p>Recordar que, igual que para la funcionalidad anterior de Localización, la característica más notable aquí será la de la “corrección” de errores que el reconocedor de voz efectúa sobre todo en el rubro ingresado por el usuario.</p> <p>Cabe destacar que en este caso, los rubros que el usuario le podrá dictar al reconocedor, son palabras corrientes del lenguaje español, tales como “bar”, “farmacia”, “mercado”, etc, y no serán nombres de calles que, como el caso de la Función de Localización, hay que enseñárselos uno por uno ya que no pertenecen al lenguaje.</p> <p>Los testeos efectuados fueron los siguientes:</p> <ol style="list-style-type: none"> 1. se dictó un rubro junto con una distancia; 2. se dictó un rubro que no existe; 3. se dictó mal un rubro, tal que: <ol style="list-style-type: none"> a. sea detectado por el método de corrección (por ejemplo “var”), b. no sea detectado por el método de corrección (por ejemplo “nar”, sin buscarle un motivo por el cual alguien diría eso, solo por el bien del ejemplo!). 4. se repitió el testeo 1. pero esta vez con otra distancia, para corroborar que la misma esté siendo tenida en cuenta.
Resultado Esperado:	<ol style="list-style-type: none"> 1. Se espera obtener como resultado un conjunto de puntos en color amarillo representando los comercios del rubro ingresado a una distancia igual o menor a la ingresada; 2. Se espera obtener un mensaje de error indicando que no existe tal rubro; 3. Se espera: <ol style="list-style-type: none"> a. que el error sea corregido, por lo tanto efectuándose la búsqueda correctamente y con normalidad, b. que la palabra escrita sea otra del diccionario, como por ejemplo “dar”. 4. Se espera obtener el mismo conjunto del punto 1. pero con más o menos comercios, dependiendo si la distancia especificada es mayor o menor a la original. Es decir, que se pretende ver cambios en la cantidad de comercios arrojados como resultado.
Resultado Obtenido:	<ol style="list-style-type: none"> 1. Se obtuvo efectivamente un conjunto de puntos en color amarillo representando los comercios buscados; 2. Se obtuvo un mensaje de error indicando que no existía el rubro ingresado; 3. Se obtuvo: <ol style="list-style-type: none"> a. Un conjunto de puntos representando los comercios cuyo rubro es el corregido del ingresado por el usuario, b. Un mensaje de error indicando que el rubro ingresado no existe, ya que como se esperaba, el reconocedor escribió una palabra que no representaba un rubro válido. 4. Se obtuvo un conjunto similar al del punto 1. pero con una cantidad menor o mayor, según las pruebas con diferentes distancias.
Errores:	<ul style="list-style-type: none"> • Mismo error de zoom y pan descrito anteriormente.
Conclusión Final:	Se concluye que la funcionalidad operó con éxito, arrojando el resultado esperado en la totalidad de los testeos efectuados.
Observaciones:	<ul style="list-style-type: none"> • Se aplica aquí la misma observación hecha para el caso de la funcionalidad de localización: si justo el reconocedor escribiera por error una palabra que no fuese un rubro, pero que tuviese el mismo código Soundex que uno, el programa devolvería esos rubros, cuando el usuario nunca los pidió. • Recordar que las palabras válidas como rubros ya están en el diccionario del reconocedor, y que no hubo necesidad de enseñárselas, como es el caso de los nombres de las calles.

8. Conclusiones

Este Capítulo pretende mostrar las conclusiones a las cuales se arribó luego de la culminación del presente trabajo. Se ha dividido en Secciones que ilustran las conclusiones más importantes dentro de diferentes áreas.

Es de notar que las conclusiones a las cuales se llegó no hablan únicamente del sistema en general, sino por ejemplo, de la necesidad de los mercados de contar con un sistema de este tipo, y de las bondades que el lenguaje UML ha representado a lo largo del proyecto para modelar diferentes aspectos de éste.

Las conclusiones particulares que conciernen al Prototipo también serán abordadas al final del Capítulo.

8.1 Objetivos Cumplidos

La primera conclusión es que se cumplieron los objetivos planteados para el presente trabajo. Recordándolos de la Sección 2.2 junto con resultados esperados presentados en la Sección 2.3, estamos en condiciones de afirmar que los objetivos y resultados esperados han sido cumplidos.

Por un lado, se obtuvo un completo estudio, análisis y diseño de un sistema general que permite trasladar los servicios de localización a un usuario móvil cuya interacción con el sistema es a través de una interfaz en lenguaje natural hablado.

Por otro lado, se obtuvo un prototipo que muestra la viabilidad del sistema analizado. Tomando como punto de partida ciertos Casos de Uso y simplificándolos, se logró hacer funcionar un prototipo que cuenta con un desarrollo vertical. Esto es, que tomando como contexto el diagrama de módulos introducido en la Sección 2.4 (Módulos que componen el Sistema) y explicados en la Sección 2.5, el prototipo abarca una parte de cada capa, atravesando el diagrama en sentido vertical. Se comienza por almacenar los datos en bases de datos y archivos de texto (Capa Bases de Datos Geográfica), se cuenta con un servidor de mapas y diccionario geográfico (Capa de Contenidos), luego con un GeoCoder, GeoParser, consultas y servicios cercanos (Capa de Aplicaciones) e incluso una implementación de la capa de Reconocimiento de voz.

Por todo esto, es que se concluye que los objetivos y resultados esperados del proyecto se cumplieron. A continuación se entra en más detalle respecto a ciertas conclusiones particulares sobre diferentes aspectos del Sistema.

8.2 Factibilidad del Sistema

Esta Sección tiene el propósito de mostrar la factibilidad del sistema descrito a lo largo del presente documento. Cuando hablamos de factibilidad, nos referimos a, al menos, tres cosas diferentes:

- Que sea **Técnicamente Viable** incluyendo escenarios para la puesta en marcha de las diferentes alternativas tecnológicas,
- Que exista la **Infraestructura Necesaria** para llevar adelante las diferentes alternativas tecnológicas,
- Que el sistema sea **Vendible y Rentable** es decir necesario y útil para los potenciales clientes.

Estos temas serán los abordados aquí. Como es natural por no tratarse de un proyecto de ámbito económico o de marketing/ventas, el enfoque y la profundidad que se le dará al último punto (Vendible y Rentable) serán los suficientes como para concluir que el sistema en cuestión cumple éstos objetivos, al menos a los ojos de un estudiante de Ingeniería y algunos expertos de Movicom, sin adentrarse en mayores detalles.

Técnicamente Viable

Con el concepto de Viabilidad Técnica nos referimos a la posibilidad de contar con un sistema que de soporte a todos los Casos de Uso planteados en este documento, e incluso a los Requerimientos Alternativos. Dado que todos los Casos de Uso utilizan un mismo conjunto de tecnologías, bien vale la pena analizarlas una a una.

En primer lugar, está el problema de la tecnología utilizada en los dispositivos móviles. Para el caso analizado y diseñado en este trabajo, bastaría con un teléfono celular con capacidad GPS. Es decir, de alguna forma agregarle a un teléfono celular estándar la capacidad de transmitir su posición en todo momento a través de la utilización de un GPS. Si bien este tipo de tecnología no se encuentra (hoy día) a disposición, no es difícil de ver que tal integración no parece

un reto importante, sino que pareciera ser tan simple como dotar a un celular de un dispositivo GPS de un tamaño prudencial (lo cual de hecho ya se estaría casi en condiciones de hacerse si se tiene en cuenta que se disponen de dispositivos GPS para PDAs/Palms) y enviar sus coordenadas a la radio base más cercana de manera de que el sistema conozca con precisión su localización exacta en todo momento.

Por lo tanto, no se ven grandes dificultades para lograr disponer de un teléfono con capacidad GPS. Incluso, vale la pena recalcar, que si bien la tecnología GPS ha sido la elegida en este trabajo por su exactitud, desarrollo, disponibilidad y rapidez, existen otros métodos que incluso son aplicables a los teléfonos móviles actuales, que usando elementos como la potencia de la onda recibida en la radio base, o la diferencia de tiempo entre las recepciones de tres radio bases, o el ángulo de llegada a la antena de la radio base, pueden dar con mayor o menor precisión la ubicación del móvil. No hay que olvidar tampoco la tradicional ubicación por celda, en la cual el sistema conoce la celda en la cual se encuentra el usuario, aunque una celda puede tener tamaños desde unas decenas de metros hasta kilómetros.

El problema de ubicar al usuario viene naturalmente en segundo lugar. Es decir, conocer su posición de manera automática e inmediata. Cualquiera de estos dos calificativos es importante.

Tal ubicación debe ser automática, en el entendido de que no se puede pensar en un sistema de este porte siendo vendido a decenas de miles de usuarios de telefonía celular, y necesitar cientos de operadores que tengan como objetivo ayudar al sistema en ubicar a cada usuario. Por esto queda claro que tal localización tiene que ser automatizada, y en todo momento el sistema debe ser capaz de poder ubicar a cualquier usuario afiliado a éste.

Además, esto se debe hacer en forma inmediata. Es decir, no se puede esperar cinco minutos para que el sistema (o algún dispositivo especial) haga el cálculo de la ubicación exacta del usuario en ese momento. Pero, ¿por qué no? Porque en cinco minutos, un usuario móvil puede estar a kilómetros de distancia de donde efectuó la llamada originalmente. No es difícil imaginarse a un usuario conduciendo un vehículo y que desee averiguar cuál es la estación de servicio más cercana a su ubicación actual. Tampoco es difícil imaginarse el descontento de éste si la información devuelta por el sistema le muestra que ya se pasó de largo por la única estación abierta, por haberse demorado cinco minutos en calcular su ubicación.

Entonces, la pregunta natural es: ¿Existe en la actualidad (o en desarrollo) una tecnología que permita ubicar a una persona en forma automática e inmediata? La respuesta es que sí existe, y no es una sola, aunque aquí nos referiremos a una en particular, el GPS (Global Positioning System, ver **Anexo B**). Por interesarnos aquí únicamente en las conclusiones de este trabajo, es que no iremos en profundidad sobre los detalles del GPS o de los otros posibles sistemas de posicionamiento (Ver **Anexo A**). Lo que sí cabe señalar, es que estos dispositivos son pequeños (actualmente utilizados en computadoras de bolsillo o asistentes personales o PDAs), son económicos (tema del cuarto punto), son rápidos (en todo momento conocen la ubicación del dispositivo) y son “exactos”. Las comillas en el término “exacto” vienen de que la exactitud de una solución de este tipo depende fuertemente en los requerimientos del sistema que se involucra. Para nuestro sistema podemos aseverar con total seguridad que la tecnología GPS es exacta hasta de más.

En tercer lugar, luego de ubicar al usuario, está el problema de enviar los datos que el usuario habla al sistema, o al menos a la parte del sistema que se encargue de tomar los datos del usuario. Ya que nos encontramos en el contexto del trabajo con teléfonos celulares, la telefonía celular es quien hará de vínculo entre el usuario y el sistema propiamente dicho, es decir los equipos que contengan el software necesario para llevar adelante los Casos de Uso. Por lo tanto podemos confiar en que tanto los teléfonos celulares como la infraestructura de los operadores telefónicos es la adecuada (ver punto de Infraestructura Necesaria más adelante).

Como cuarto posible problema técnico, encontramos al reconocimiento de voz (ver **Anexo C**). En particular, el problema aquí reside en contar con un software capaz de reconocer un mensaje en lenguaje natural hablado²² y con posible ruido en la línea (ya que el ruido en las líneas de telefonía celular existe y no es menor). Si bien se cuenta en la actualidad con tecnologías de reconocimiento del habla muy avanzadas comparadas con las de los últimos años, lo que este sistema necesita es un software que, por un lado, entienda lo que el usuario dice a la velocidad natural con que habla; y por otro, que entienda a cualquier usuario, sin la necesidad de entrenar al sistema con la voz de cada usuario afiliado al mismo (lo que se conoce como independencia del locutor).

Estos dos problemas relativos al reconocimiento de voz no son menores, sino todo lo contrario, e incluso podría pensarse que son uno de los principales problemas técnicos que padece este sistema propuesto. Sin embargo, no hay que olvidar dos elementos. Primero, que las tecnologías disponibles actualmente permiten un alto grado de reconocimiento incluso en un diálogo casi a la velocidad normal con que una persona habla. Productos como el utilizado en el prototipo (IBM ViaVoice Millennium) por momentos sorprenden por su alta comprensión y su

²² Por esto es que, en principio, puede considerarse el estándar VoiceXML de utilidad ya que se enfoca a diálogos guiados. Por mayor información ver [TellMe, VoiceXML].

velocidad. Segundo, el avance en esta área hace prever casi con seguridad que este problema será solucionado en el corto o mediano plazo²³.

Tal vez, entonces, el problema restante, del reconocimiento independiente del locutor, sea de los más importantes, ya que no parece contar con el ímpetu de desarrollo con el que cuenta el primer problema de reconocimiento. Parece ser más importante para las empresas involucradas en estos desarrollos obtener un producto de reconocimiento en tiempo real y a velocidad normal, que uno lo suficientemente flexible para ser independiente del locutor. De todos modos, son de esperar avances tal vez en el mediano plazo en este tema, e incluso en la actualidad ya se cuenta con algunas versiones de productos de reconocimiento de voz independientes del locutor²⁴. Algo que excusa a las empresas desarrolladoras de sistemas de reconocimiento de voz es que generalmente se disponen de productos independientes del locutor cuando las gramáticas y diccionarios capaces de soportar son relativamente pequeños, y se cuenta con productos dependientes del locutor cuando se soportan gramáticas y diccionarios muy grandes.

Otro problema no menor es el de la síntesis de voz. Si bien en la actualidad se cuenta con una amplia gama de productos capaces de sintetizar la voz humana con gran precisión, lo que aún no logran es dotar de naturalidad y espontaneidad a tal síntesis.

También es justo decir, que últimamente se puede apreciar un cambio de rumbo en los productos ofrecidos, y en los futuros problemas a resolver por parte de las empresas desarrolladoras en cuanto a dotar a sus productos de una síntesis de voz que suene natural, así como de una independencia del locutor, aún soportando grandes diccionarios.

Según expertos del área de la telefonía celular²⁵, el problema de entrenar a los usuarios del sistema, sin dudas podría tirar abajo una iniciativa en este sentido, ya que es impensable para un operador telefónico disponer de los recursos y la logística necesaria para entrenar al sistema con decenas de miles de usuarios.

Como quinto problema técnico, podríamos mencionar la vulnerabilidad del sistema a los errores introducidos cuando el usuario habla. En particular, nos referimos a errores en los nombres de las calles, en los nombres de los rubros de comercios, etc. Por tanto, ciertas medidas se deben tomar. En particular, para el caso de los prototipos, se utilizó el método Soundex ya descrito. Como se comentó, no es el único, y lo importante es que al momento de desarrollar un sistema de este tipo ya se cuentan con soluciones que abordan esta problemática²⁶. Por lo tanto este problema no parece ser de extrema importancia.

Infraestructura Necesaria

Este punto tiene como objetivo analizar la posibilidad de problemas técnicos en la infraestructura necesaria para dar soporte a un sistema de este tipo. Por lo tanto, comencemos por identificar que elementos son necesarios, es decir, que entendemos por infraestructura para este caso.

Primero, se necesita una infraestructura de telefonía celular, es decir, una red de antenas o radio bases capaces de cubrir el área a la que se pretenda dar servicio. Debido a que la información transmitida desde el dispositivo móvil (teléfono celular con capacidad GPS) hacia estas antenas es voz, alcanzaría la infraestructura existente de empresas como ANCEL o MOVICOM, si es que éstas también soportasen la recepción de la ubicación del usuario dada por el GPS contenido en el dispositivo, por supuesto, lo cual merecería un estudio a parte.

Vale la pena recordar los comentarios hechos en la Sección 3.6 (Requerimientos Alternativos) en donde se señalaba que la eventual utilización de otro tipo de dispositivos tipo PDA para transmitir datos, podría requerir de modificaciones en la infraestructura estándar de telefonía celular. También es justo decir que tales modificaciones ya se están poniendo en práctica. Por ejemplo, la empresa MOVICOM a través de la utilización de WAP (Wireless Application Protocol) para acceder a Sitios Web programados en WML (Wireless Markup Language), o la empresa ANCEL con su protocolo CDPD que permite a cualquier dispositivo con un radio MODEM CDPD comunicar datos a través de la reciente infraestructura instalada (año 2001).

En particular, las alternativas manejadas en la Sección 3.6 deberían hacer uso de la tecnología WAP o CDPD en el corto plazo, ya que son tecnologías que están disponibles en nuestro mercado actualmente. Tal utilización de tecnologías sería necesaria ya que los dos modelos alternativos incluyen el envío de datos desde y hacia el usuario. El problema con CDPD por ejemplo, es que solo permite la transmisión de datos, y no de voz, mientras que WAP trabaja sobre CDMA. Claramente existen también en el mundo otras tecnologías que podría dar soporte a estos modelos alternativos (como es el caso de GSM), pero en el Uruguay y en la actualidad, solo se dispone de las mencionadas, y

²³ Es interesante ver como en [IVLM] se indica que tal reconocedor de habla continua es parte de los trabajos futuros de la empresa Telefónica España para su browser IVLM.

²⁴ Ver [SpeechWorks].

²⁵ Ver [Entrevista].

²⁶ Ver [Onomastix] y su línea de productos, en particular el MetaMatch™.

tomando en cuenta la muy reciente migración a teléfonos celulares digitales y la casi inutilización de CDPD, puede esperarse que las cosas sigan como están unos cuantos años más (reflexión personal del autor).

En resumen, la infraestructura de telefonía celular no parece ser un impedimento para el completo desarrollo del sistema en cuestión, e incluyendo algunas de las variantes introducidas por los Requerimientos Alternativos.

Como segunda cosa, se necesitan equipos por parte de los operadores de telefonía celular que acepten las llamadas efectuadas al sistema y que “contengan” al sistema en sí, es decir, a los componentes de software y hardware necesarios para que éste funcione correctamente. Estamos hablando aquí no solo de computadoras, sino de equipamiento específico para atender llamados telefónicos originados en teléfonos celulares.

En concreto, el sistema utilizado en la actualidad por la empresa MOVICOM consta de un equipamiento Motorola® denominado IVR (Interactive Voice Response), el cual atiende las llamadas de los celulares por ejemplo al servicio Movimemo® de MOVICOM. Luego de entrar la llamada, el usuario es preguntado para oprimir el número de opción de menú que desea, o decirlo. Por tanto, este sistema ya cuenta con un primitivo reconocimiento de voz, que en este caso se reduce a números dentro de una opción de menú.

Podría pensarse en extender dicho sistema, permitiéndole aceptar llamados al sistema en cuestión y atenderlos de manera de poder entablar un diálogo fluido en idioma español, más allá de los números²⁷. Aquí el problema de los costos es el más significativo, y se lo comentará más adelante.

A modo de resumen, el problema de la infraestructura es complejo, aunque el sistema que se analizó podría hacer uso de la infraestructura de antenas y radio bases existente, y solo requeriría extender las capacidades de los equipamientos que atienden los llamados telefónicos permitiéndoles correr aplicaciones de reconocimiento de voz avanzadas.

Sistema Vendible y Rentable

Este punto pretende mostrar una característica fundamental en todo emprendimiento, la posibilidad de venta. Por más que el sistema sea técnicamente viable y cuente con una infraestructura que lo soporta, si no es atractivo para las empresas operadoras de telefonía celular así como para los potenciales clientes, nos encontraríamos frente a un trabajo que no sería capaz de salir de la órbita académica.

Obviamente tal característica dista de ser un objetivo del presente trabajo, pero sin duda introduce un aspecto al menos interesante para comentar.

Por un lado, no existe en el mercado actual ningún sistema ni remotamente similar al aquí expuesto. Incluso no existe un sistema que acepte llamadas telefónicas estándares a través de un operador que brinde los servicios analizados, mucho menos uno automático y orientado a clientes móviles.

Por otro lado, además de que no es difícil de imaginar la aceptación que un sistema de este tipo traería por parte de los usuarios, según los expertos consultados, es un producto que va a salir al mercado, siendo las preguntas más importantes ¿Quién lo sacará al mercado?, ¿Cuándo? y ¿En qué condiciones técnicas y a que precios?

En cuanto a la rentabilidad de un sistema de este tipo, es claro que hay varios factores a analizar: la inversión inicial por parte de las empresas operadoras, el costo mensual de éstas, y el monto a cobrar a los usuarios por tales servicios. Sobre este tema, los consultados expresaron el enorme costo que insumiría la inversión inicial, debido a que montar la infraestructura necesaria (solamente los equipos) sería un problema muy grande. Los motivos son: los equipos no son estándar, sino que son utilizados exclusivamente por empresas de telefonía celular; las empresas que los proveen, instalan y configuran dichos equipos son extranjeras, aumentando aún más los costos; y por último, el desarrollo de aplicaciones (por ejemplo el software de reconocimiento de voz que iría en los equipos) conlleva elevadísimos costos.

Esto quiere decir, que tal vez no sea posible la integración de los equipos IVR con un software de reconocimiento ya existente como es el caso del producto SpeechWorks Recognizer, ya que éstos equipos no son computadoras estándar. Obviamente esto requiere de una investigación más profunda que dictamine si es posible o no tal integración, y en caso de no ser posible, que convendría: o utilizar los equipos IVR existentes y desarrollar un reconocedor de voz; o utilizar un producto como el de [SpeechWorks] a costa de cambiar los equipos IVR.

Debido a la dificultad de utilizar los actuales (y antiguos) sistemas propietarios IVR, es que surge la posibilidad de utilizar nuevas tecnologías como WAP o VoiceXML, siendo ésta última la más atractiva para adaptarla al uso de un reconocedor de voz y síntesis de voz.

²⁷ Ver [SpeechWorks] para una descripción de un reconocedor de voz posible de ser utilizado en conjunto con un IVR.

Por último, los intereses de las empresas de telefonía celular son dos: generar más minutos de aire y dar un contenido diferenciado al de la competencia. Teniendo en cuenta que no existe actualmente un producto de estas características, y que naturalmente generaría minutos de aire, todo indica que estamos frente a un producto vendible.

Con respecto a los costos mensuales o anuales involucrados en brindar soporte telefónico a través de operadores “vivos” (personas que atienden el teléfono en la empresa) vs. sistemas de reconocimiento de voz capaces de hacer un trabajo similar, vale la pena ver un análisis hecho en este sentido presentado en [BusinessCase] por la empresa Nuance, quien vende soluciones basadas en sistemas de reconocimiento de voz para empresas de manera de poder reemplazar (y complementar) a los operadores telefónicos.

Por referencias sobre la respuesta obtenida por los usuarios, es interesante ver también el documento [ScoreBoard] donde se muestra las preferencias y respuestas de los usuarios de los sistemas de V-Commerce (Voice Commerce).

8.3 Planteo de Escenarios

Esta Sección sirve de resumen de la anterior, e introduce dos posibles escenarios (de tiempo) para llevar a la implementación un sistema completo como el descrito en este documento. Tales escenarios han sido planteados entre el autor y los expertos consultados²⁸, llegando a la siguiente conclusión:

Escenario a Corto Plazo:

En un escenario a corto plazo, podemos pensar en implementar un sistema meramente semejante al aquí descrito, pero con una gran diferencia: no contar con la interfaz hablada. Obviamente más que una modificación, se trata de otro sistema completamente diferente, pero por el bien de identificar un escenario a corto plazo es bueno hacer el ejercicio. Tal sistema se implementaría a través de WAP o CDPD. Es decir, que sistema y usuario intercambiarían datos entre sí, y no la voz.

Para el caso de utilizar WAP²⁹, se pensaría en una solución brindada por la empresa operadora que actualmente trabaja con WAP, que es MOVICOM. No se necesitarían complejos equipos dentro de la empresa operadora, ya que con servidores estándar (por ejemplo con sistemas operativos Windows 2000 o NT) y aplicaciones existentes se puede desarrollar un sistema que reciba las peticiones de servicio en forma de texto, las procese, y las devuelva en forma de texto también, utilizando WML como lenguaje para generar los sitios Web a los cuales los usuarios se conectarían. Tal vez incluso, se podría pensar de devolver al usuario un mapa sumamente primitivo y de muy baja calidad y poco tamaño (debido a las serias limitantes gráficas de los teléfonos celulares actuales).

Para el caso de utilizar CDPD³⁰, se pensaría en una solución brindada por la empresa operadora que actualmente trabaja con CDPD, que es ANCEL. De modo similar al anterior, no se necesitarían complejos equipos del lado del operador, pero si se requerirían dispositivos que contengan un radio MODEM CDPD³¹, que no son ni funcionan como teléfonos celulares. Aquí también la respuesta podría hacerse en forma de mapa devuelto al usuario, dependiendo de las capacidades gráficas del dispositivo utilizado.

Como otra alternativa de corto plazo (aunque tal vez sea más justo decir Mediano Plazo debido a lo nuevo de la tecnología), y que utilizara las bondades de la interfaz de voz, puede pensarse en disponer de un sitio Web desarrollado utilizando VoiceXML, al cual le lleguen peticiones de diálogo en forma vocal desde un teléfono o dispositivos móviles. En la actualidad, los sitios que utilizan éste lenguaje se dedican casi exclusivamente a pequeños diálogos muy guiados, en los cuales se espera que el usuario diga alguna opción o frase pequeña y muy puntual. Es de esperar que este lenguaje se adapte también hasta poder procesar frases enteras habladas en lenguaje natural, así como gramáticas dinámicas de muy gran tamaño.

Esta alternativa de VoiceXML también le aqueja el problema de la interacción con los dispositivos GPS para recibir la ubicación actual del usuario.

²⁸ Ver [Entrevista].

²⁹ Ver [Movicom, OMA y WAP].

³⁰ Ver [Ance].

³¹ Tales dispositivos ya se encuentran disponibles, como los radio MODEMs CDPD del PDA *iPAQ* de Compaq, que permite al usuario navegar por Internet usando el servicio ANCEL Integra (ver [Ance]), o los dispositivos móviles m-POS de la empresa Akyman® Uruguay, ver [Akyman].

Escenario a Largo Plazo:

En un escenario a largo plazo, podemos pensar en implementar el sistema en su totalidad. Es decir, que se cuente con teléfonos celulares con capacidad GPS, con los equipos necesarios del lado del operador, pero por sobre todas las cosas, con una tecnología de reconocimiento de voz capaz de ser independiente del locutor y que pueda funcionar en forma conjunta con los equipamientos de los operadores de telefonía celular³².

Podría pensarse aquí en servidores VoiceXML capaces de recibir la ubicación actual del usuario venida del GPS, y entablar un diálogo más flexible que los actuales que utilizan este lenguaje.

Con respecto a los terminales, no sería de ningún asombro si incluso en el corto o mediano plazo se contara con teléfonos celulares capaces de enviar su ubicación actual a la radio base que lo esté escuchando.

Incluso podría pensarse de la utilización de tecnologías como GSM, que permiten la transmisión de voz y datos a alta velocidad, lo que permitiría al usuario dirigirse al sistema en forma hablada, y éste a responderle en forma de un mapa interactivo detallado incluyendo la respuesta deseada, en el cual el usuario podría efectuar zoom, desplazamientos, nuevas búsquedas dentro de los resultados obtenidos, etc.

Como resulta evidente, es necesario efectuar una investigación concienzuda para poder evaluar las diferentes alternativas tecnológicas a corto, mediano y largo plazo. El interés aquí fue el de presentar ideas basadas en las tecnologías disponibles en la actualidad (WAP, CDPD) y en las emergentes (VoiceXML).

8.4 Flexibilidad de la Solución Propuesta

Aunque este documento incluya un Capítulo entero dedicado al análisis y diseño de una solución para un sistema en particular, debe quedar claro que dicha solución propuesta es sumamente flexible. Esto significa que la solución planteada es tentativa e inicial, y que puede ser modificada dependiendo del modelo de negocios de quien lo implemente, de la realidad tecnológica actual al momento de implementarlo, del conocimiento y experiencia del equipo de desarrollo, etc.

Algunos ejemplos de esta flexibilidad son:

- No se impone ninguna metodología de implementación del sistema. El modelado del mismo ha sido hecho en UML, pero la codificación del software puede hacerse con cualquier lenguaje de programación, preferentemente uno orientado a objetos dado que se utilizó UML.³³
- El uso de Interfaces en el modelado (ISpeech, IGeoCoder, IGeoParser, ILocalización), permite al desarrollador del software implementar a su modo cada componente, ya que lo único fijo deben ser las operaciones exportadas por cada interfaz, pero la forma que tiene cada componente de efectuar esa operación es reservada, y depende de cada desarrollador o equipo.
- Los Casos de Uso analizados son generales, pudiéndose (y debiéndose) adaptarlos a las necesidades al momento de implementar dicho sistema. A su vez, éstos seguramente no son los únicos Casos de Uso posibles de ser implementados. Pero incluso en caso de decidir utilizar sólo estos Casos de Uso presentados aquí, son lo suficientemente generales como para ser adaptados a diferentes necesidades. Recordar que van desde ubicación de direcciones y lugares, hasta publicidad, pasando por el cálculo de rutas.
- Se presentan posibles (y totalmente válidas) alternativas de requerimientos, por lo que se puede pensar en implementar este sistema u otro similar con algunas restricciones basándose en los diferentes escenarios de tiempo planteados, así como los modelos de negocios planteados. A su vez, se alienta a buscar otros requerimientos alternativos además de los aquí expuestos.
- No se especifica la tecnología a utilizar en los dispositivos móviles. Solo se requiere que tengan capacidad de voz y GPS, o incluso alguna tecnología similar a GPS en cuanto a exactitud y velocidad.³⁴

³² Como pueden ser los productos de la empresa SpeechWorks, la cual ya se mencionó (ver [SpeechWorks]).

³³ Ver **Anexo C** por los diferentes Servicios de Voz que el Estado del Arte permite.

³⁴ Ver **Anexo A**.

- No se especifica la tecnología/estándares a utilizar para el reconocimiento de voz y síntesis de voz, dejando este tema para abordar al momento de la implementación efectiva del sistema en su totalidad, dados los continuos avances en la materia.
- Se alienta a implementar otros mecanismos de corrección de los errores introducidos por el usuario, además del Soundex ya explicado. Un ejemplo de esto son los utilizados por la empresa Onomastix.
- No se limita al Uruguay exclusivamente. Tanto el diseño global del sistema como la implementación en particular del Prototipo permite la adaptación a cualquier mapa de cualquier país. Indudablemente, el mayor problema técnico aquí es el de la voz, ya que tanto los reconocedores como los sintetizadores de voz son absolutamente dependientes del país o región, lo cual no ocurre con el procesamiento de las consultas geográficas, que se aplican de igual manera para cualquier mapa.

8.5 Potencial de UML

Cabe destacar como una conclusión final del presente trabajo, el potencial de modelado del lenguaje UML.

En particular, este modelado se utilizó para efectuar un diseño del sistema en cuestión. Tal diseño, como se puede apreciar en el Capítulo 5 y recientes conclusiones, es general y no particular. Es decir, que trata de ser abarcativo y flexible, y de no imponer restricciones al diseño.

Tomando en cuenta esto, ha sido notable la adaptabilidad del lenguaje UML para modelar dicho sistema. Esto muestra que es un lenguaje útil no sólo para modelar sistemas concretos, específicos, sino también para modelar sistemas generales y flexibles como el presente.

8.6 Conclusiones del Prototipo

Con respecto al prototipo, se puede concluir que éste cumple con el objetivo planteado que era el de demostrar la viabilidad del sistema por medio de la implementación de un prototipo “vertical” (término explicado antes).

Se entiende que con un prototipo de esta naturaleza, no se esta demostrando retunda y completamente que el sistema es efectivamente implementable. Pero, sirve de mucho para mostrar que se trata de un sistema con un gran tinte de realidad, que tiene fundamentos tecnológicos y no se funda en supuestos de tecnologías futuras.

A su vez, vale la pena recalcar que el prototipo sigue la metodología iterativa incremental presentada en la Sección 6.1.2, pero no está basado directamente sobre el análisis y diseño efectuado para el sistema todo.

Los motivos de esto son varios: el prototipo representa una versión reducida y definitivamente diferente a cualquier Caso de Uso del sistema; los tiempos que insumirían efectuar un análisis, diseño e implementación del prototipo teniendo en cuenta que se trataba de un alumno solo para hacerlo, eran demasiados; el interés principal estaba en estudiar, analizar y diseñar una solución general para un sistema de gran porte, y grandes esfuerzos se dedicaron a ello, pero no surgía como interés principal el análisis y diseño del prototipo, sólo importaba que el prototipo representara al sistema en general y mostrara que éste era posible.

Se concluye también que el Prototipo, en su uso y funcionamiento, se comporta correctamente³⁵, presenta una interfaz amigable (tanto gráfica como vocal), muestra tiempos de respuesta adecuados y se desempeña bien en todas sus funcionalidades.

³⁵ Ver Capítulo 7 por los resultados de los testeos efectuados sobre el Prototipo.

9. Trabajos Futuros

Este Capítulo pretende mostrar los trabajos futuros relacionados con el presente proyecto. En particular, pretende mostrar posibles áreas de trabajo para seguir en la línea de investigación actual y particularmente acerca de la expansión de las capacidades del Prototipo.

Por tratarse de un proyecto con muchas puntas, muchas áreas, muchas tecnologías y que no está basado en ningún otro proyecto existente, no resulta difícil sugerir posibles áreas de trabajo que extiendan, profundicen y utilicen como base el presente trabajo.

Los trabajos futuros que se ven como posibles son los siguientes, separados tomando en cuenta el objetivo, ya sea general (que involucre estudiar un conjunto de temas) o puntual (que involucre estudiar o implementar cierto tema en particular). La división es *grosso modo*, y no está exenta de comentarios.

Trabajos Futuros de **Objetivo General**:

- Implementación de un Prototipo on-line. O sea, poder realizar las pruebas con el prototipo desde un teléfono celular que permita identificar la ubicación actual del usuario, y, además, que le devuelva a éste el resultado desde el sistema a su teléfono. Aquí podría pensarse en utilizar la infraestructura ya existente de los operadores de telefonía celular. En lo posible, sería bueno también poder utilizar la voz para la comunicación Usuario-Sistema.
- Estudio profundo de las arquitecturas y escenarios tecnológicos que darían soporte al sistema aquí analizado en su totalidad, incluyendo el uso de la voz y de la ubicación a través de un dispositivo GPS. Este estudio comenzaría con el estudio de las tecnologías WAP, CDPD y VoiceXML.
- Incorporación al Prototipo de una respuesta en forma de mapa. Esto implicaría, al menos, un fuerte estudio de las capacidades gráficas que las diferentes tecnologías de dispositivos móviles permiten. Esto impactaría directamente en el tipo de dispositivo móvil a utilizar, ya que, en principio, no parecería viable utilizar un teléfono celular para desplegar mapas en él. Esto traería a la mesa otros dispositivos, de amplio crecimiento, como son los PDAs. Por lo tanto, parece una buena oportunidad incluso migrar el prototipo actual para que permita el trabajo con dispositivos del tipo PDA aprovechando las grandes capacidades gráficas que éstos poseen.
- Estudiar más Casos de Uso aplicables al Sistema. Simplemente, estudiar qué otros Casos de Uso sería interesante agregar al sistema, incluso a costa de eliminar o modificar alguno existente en este documento. También hay que notar que la incorporación de un Caso de Uso, impacta fuertemente en todo el proyecto.
- Hacer andar el Prototipo (o similar) en el corto plazo a través de la utilización de la tecnología WAP, sólo con interfaces de texto y gráficas, aún a costo de no poder utilizar la voz (ver Sección 3.6 y Capítulo 8 para más detalles).

Trabajos Futuros de **Objetivo Puntual**:

- Estudio e incorporación de un Reconocedor de Voz en “Forma Directa”. Esto refiere a la interesante posibilidad de disponer de un software con estas características para efectuar pruebas de testeado con el prototipo, y verificar bajo qué condiciones es mejor utilizar un reconocedor estándar, y bajo qué condiciones es mejor uno en forma directa. En base a esos resultados, se podría pensar de incorporar al prototipo los dos tipos de reconocedores y elegir cual utilizar en forma dinámica, utilizándose cada tipo cuando las condiciones así lo indiquen.
- Mejoras al GeoParser en lo referente a la intención del usuario. Esto es poder entender el deseo original del usuario sin la necesidad de que éste seleccione opciones de menú, es decir, que el sistema se vuelva lo menos orientado al diálogo que se pueda para favorecer la utilización de las tecnologías de reconocimiento de habla continua. Además, poder reconocer

otro tipo de objetos geográficos más allá de las direcciones y calles, hacer más flexible el entendimiento de direcciones no estructuradas, etc. Cualquier otra mejora que se le de al GeoParser redundará directamente en un sistema más amigable para el usuario, así como más flexible ya que será capaz de entender un mayor espectro de alternativas.

- Estudio e incorporación de otro método de corrección de los errores introducidos por el usuario. La idea es incorporar y testear el prototipo utilizando otros mecanismos de corrección de errores más robustos que el Soundex. O, al menos, intentar aplicar todas las posibles mejoras comentadas del método Soundex al prototipo de manera de obtener una mayor robustez. En caso de pensar en utilizar otra interfaz que no sea la hablada, es decir que el usuario, por ejemplo, escribiera por medio de un teclado, es de interés estudiar otras alternativas de corrección de errores, aparte del método Soundex el cual está orientado a resolver problemas en el habla y no en lo escrito. Por lo tanto, se necesitaría encontrar otro método de corrección que sea pensado específicamente para corregir errores de tipeo.
- Incorporar datos completos de Montevideo e incluir datos de otras ciudades, de manera de aumentar la capacidad de respuesta del Prototipo como también para trabajar con la información contextual. Con respecto al contexto, es de interés resolver los problemas surgidos de la repetición de nombres de elementos geográficos en las distintas ciudades o departamentos, y como éstos se resuelven a partir del contexto.
- Conjugar la respuesta del sistema con un módulo de ruteo, de manera de poder decir al usuario cómo llegar a la ubicación ingresada desde su ubicación actual. En particular, estudiar e incorporar librerías de ruteo ya implementadas en otros lenguajes. En principio, esta posibilidad aparece como una de las más probables de realizar en el corto plazo, ya que tales librerías se encuentran disponibles en la actualidad, y solo es cuestión de adaptarlas al prototipo desarrollado. Esto también se aplica a cualquier otro módulo interesante.
- Mejoras en el reconocedor de voz con respecto al entrenamiento necesario para reconocer efectivamente la voz del locutor. En particular, intentar utilizar un reconocedor de voz independiente del locutor, lo cual, dada la complejidad, se intentaría obtener uno ya desarrollado en el mercado o en proceso de desarrollo en algún ámbito académico. Recordar que este punto es fundamental ya que juega un importante papel en la viabilidad de un sistema de este tipo, como surgió de la entrevista.
- Incorporación de un GPS al Prototipo y trabajo de campo con éste. Continuando con uno de los puntos ya comentados, incorporar el sistema de posicionamiento global al dispositivo utilizado de manera de poder realizar pruebas en campo con éste. También resultaría interesante probar alguno de los otros métodos de posicionamiento expuestos en el **Anexo A**, aunque en la actualidad tales tecnologías parecen totalmente fuera de alcance.
- Probar con otros software de reconocimiento de voz. En particular con algún producto de tipo SDK que permita una buena interacción con el lenguaje de programación utilizado (sea cual fuere éste). Un caso es el Microsoft Speech SDK, tanto en la versión actual 5.1 como en la emergente para la infraestructura .NET.
- Devolver el resultado en forma de voz sintetizada, estudiando para esto varios productos (software comerciales) que lo hagan. Esto es de importancia debido al interés mostrado por los usuarios de servicios telefónicos a la hora de evaluar la interacción con éstos sistemas. Los usuarios dan mucha importancia a la naturalidad de la síntesis de la voz, tanto o más que la capacidad del sistema para reconocer sus voces. De aquí que la importancia en investigar productos de síntesis de voz que permitan comunicarse de manera natural y espontánea con el usuario sea de interés.

10. Referencias

A continuación se presentan las referencias utilizadas a lo largo de todo el documento, en orden alfabético:

- [Adaptación] Paper “Language Model Adaptation for Conversational Speech Recognition Using Automatically Tagged Pseudo-morphological Classes”, de C. Crespo, D. Tapias, G. Escalada y J. Alvarez, de: Proceedings of the International Conference on Acoustic, Speech and Signal Processing (ICASSP), Munich, 1997.
- [Akyman] Sitio Web de la empresa Akyman®: <http://www.akyman.com.br> o <http://www.akyman.com.au>, cuya subsidiaria en Uruguay dispone de dispositivos móviles mPOS con capacidad CDPD.
- [Ancel] El operador estatal de telefonía celular Ancel cuenta en la actualidad con la única línea de comunicación CDPD existente en el país, la cual se comercializa a través del nombre Ancel Integra. Tal servicio permite a usuarios móviles navegar por Internet así como crear redes IP privadas.
- [Brown] Paper “Toward Speech as a Knowledge Resource”, de EW Brown et al, obtenido de IBM Systems Journal, Vol. 40, No. 4, año 2001.
- [BusinessCase] Paper “The Business Case For Speech Recognition”, año 2000, obtenido de [Nuance].
- [Dragon] Sitio Web del producto Dragon Dictate, de la empresa Dragon Speech Recognition: <http://www.dragonsys.com/naturallyspeaking/> o el Sitio Web de ScanSoft: <http://www.scansoft.com>
- [Entrevista] Entrevista realizada personalmente al Sr. Humberto Moccio (División Ventas) y al Ing. Gabriel Pereira (División Datos) del operador de telefonía celular MOVICOM, Junio 2002, en oficinas de la empresa, sita en el Edificio “Torre El Gaucho”.
- [ESRI] Paper “What Are Location Services? The GIS Perspective”, Diciembre 2000, obtenido del sitio Web de la empresa ESRI (Environmental Systems Research Institute): <http://www.esri.com/industries/locationservices/>
- [EstadoVoz] Paper “Estado del arte en tecnologías de voz”, de Miguel Ángel Rodríguez Crespo et al, obtenido de [TID] en el Número 20 de Marzo del 2001.
- [GIS] Sitio Web de GIS Development: <http://www.gisdevelopment.net>
- [GPS] Sitio Web de MedicionesGPS: <http://www.medicionesgps.com> y Sitio Web de MundoGPS: <http://www.mundogps.com>
- [IVLM] Paper “Acceso vocal a contenidos de Internet: Plataforma IVLM”, de Juan Calero González et. al., obtenido de [TID] en el Número 20 de Marzo 2001.
- [Larman] Libro “UML y Patrones”, de Craig Larman, Prentice Hall, Pearson, ISBN 970-17-0261-1.
- [LAS] Sitio Web de Language Analysis Systems Inc.: <http://www.las-inc.com>. Grupo de investigación en el área de *Computational Linguistics* (Lingüística Computacional), cuya rama empresarial abarca la empresa Onomastix (ver [Onomastix]) para la venta de sus productos.
- [Lippmann] “Speech Recognition by Machines and Humans”, de R. P. Lippmann, Speech Communication, Cap.22: Págs. 1-15.
- [Location] Paper “Location, Location, Location”, de Micheal Dennis, Marzo 2001, obtenido de java Location Services: http://www.jlocationservices.com/company/Micheal_Dennis/LocationTechnologyArticle.html

- [Mainstream] Paper “Is Speech Recognition Becoming a Mainstream?”, de Savitha Srinivasan y Eric Brown, IBM Research Center, obtenido de <http://www.computer.org/computer/homepage/0402/GEI>, año 2002.
- [MapObjects] Documentación referente al producto MapObjects de la empresa ESRI obtenida de su Sitio Web así como de la materia electiva Sistemas de Información Geográfica, INCO, FING, UDELAR.
- [Maptuit] Paper “The Road Network – the Foundation of Location Based Services”, de Max Stevens-Guille, Enero 2001, obtenido de Maptuit Corporation: <http://www.maptuit.com>
- [Mobile] Paper “An Introduction to Mobile Positioning”, Diciembre 1999, de Mobile Lifestreams, obtenido se su sitio Web: <http://www.mobilePositioning.com>
- [Movicom] El operador privado de telefonía celular Movicom cuenta en la actualidad con el uso exclusivo de la tecnología WAP que le permite a un usuario navegar por Internet utilizando un teléfono celular WAP *Enabled* (habilitado para WAP), como es el caso del Motorola TimePort.
- [MSSpeech] Sitio Web de Microsoft Speech: <http://www.microsoft.com/speech/>
- [Nuance] Sitio Web de la empresa Nuance Communications: <http://www.nuance.com>.
- [OMA] Sitio Web de la Open Mobile Alliance, organización dedicada a dar soporte al protocolo WAP: <http://www.openmobilealliance.org>
- [Onomastix] Empresa dedicada a la comercialización de productos de tratamiento, almacenamiento, procesamiento y corrección de nombres (de personas), tomando los avances de LAS (ver [LAS]) como base. Su Sitio Web es el <http://www.onomastix.com>
- [Rec1] Paper “Últimos desarrollos en tecnologías de voz y del lenguaje”, de Ismael Cortázar Múgica et. al., Número 24, obtenido de [TID] en Enero de 2002.
- [Rec2] Paper “Tecnologías de Reconocimiento del Habla para Teléfonos Móviles”, de Stefan Dobler, obtenido del Ericsson Review Magazine No.3 año 2000.
- [Rec3] Paper “Reconocimiento de Voz en el entorno de las nuevas redes de comunicación UMTS e Internet”, de Luis Villarubia Grande y Ismael Cortazar Múgica, obtenido de [TID] en el Número 23 de Noviembre de 2001.
- [ScoreBoard] Paper “2000 Speech User Scoreboard”, obtenido de [Nuance], division Market Research.
- [Soundex1] Paper “Is Soundex Good Enough for You?”, de Frankie Patman y Leonard Shaefer, año 2001, obtenido de [LAS].
- [Soundex2] Artículo “Soundex – Can It Be Improved?”, de Peter Christian, año 1998, publicado en Computers in Genealogy Vol.6 N°.5 de Marzo de 1998.
- [SistLoc] Paper “Sistema de Localización en Redes Móviles: el servicio de emergencia 112”, de Ernesto Aranda Almansa et al., obtenido de [TID] en el Número 21 de Junio de 2001.
- [SpeechWorks] Sitio Web de la empresa SpeechWorks: <http://www.speechworks.com>; dedicada exclusivamente a productos de reconocimiento y síntesis de voz con soluciones tanto para equipamientos IVR (SpeechWorks OpenSpeech Recognizer v6.5 Second Edition) como para dispositivos móviles (SpeechWorks Speech2Go).
- [TellMe] Paper “VoiceXML in the Enterprise: Facts and Fiction”, de la empresa Tellme Networks: <http://www.tellme.com>
- [TID] Sitio Web de la empresa Telefónica España, Investigación y Desarrollo (I+D): <http://www.tid.es> Sección Publicaciones. Allí aparecen todos los números de las publicaciones de Telefónica I+D.
- [ViaVoice] Sitio Web del producto IBM ViaVoice y similares: <http://www-3.ibm.com/software/speech/es/>

- [VoiceWeb1] Paper “The Voice Web”, obtenido de [Nuance].
- [VoiceWeb2] Paper “Business Opportunities on the Voice Web”, obtenido de [Nuance].
- [VoiceXML] Sitios Web de: W3C: <http://w3.org.voice>; de VoiceXML Forum: <http://www.voicexml.org> y de VoiceXML Review: <http://www.voicexmlreview.org>
- [Voyager] Nuance Voyager Voice Browser, ver [Nuance].
- [VozUMTS] Paper “Servicios de voz en redes UMTS”, de Juan Calero González y Carlos Baena Parrado, obtenido de [TID] en el Número 21 de Junio de 2001.
- [VB6] Libro “Descubre Visual Basic 6.0”, de Bob Reselman et. al., Prentice Hall, publicado por QUE, ISBN 84-8322-078-4.
- [WAP] Sitio Web de la organización WAP Forum, creadora y primera promulgadora de esta tecnología que en la actualidad la comparte con la OMA (ver [OMA]): <http://www.wapforum.org>
- [Whose] Paper “Whose Name Is It: Names, Ownership and Databases”, de Kerry Dematteis et al., año 2001, obtenido de [LAS].

ANEXO A: Tecnologías de Posicionamiento

Este Anexo pretende dar una breve descripción de las diferentes tecnologías de posicionamiento que existen en la actualidad para la ubicación de un dispositivo móvil.

Introducción

El interés de mostrar la existencia de tales tecnologías se hace evidente al reconocerlas como esenciales para poder brindar una solución al problema planteado en este proyecto. Luego, en el siguiente Anexo, se explicará más en detalle una de estas tecnologías en particular.

Las tecnologías de posicionamiento disponibles son normalmente divididas en dos grupos: las basadas en terminales y las basadas en la red de la empresa operadora de telefonía celular. De estos dos grupos, algunas de las tecnologías más conocidas son las siguientes³⁶:

Basadas en Terminales

GPS - (Global Positioning System)

Usa un juego de satélites para localizar la posición de un usuario (ver **Anexo B**). Este se ha usado en los sistemas de navegación de vehículos así como en los dispositivos portátiles especializados durante algún tiempo, y ahora se está haciendo su camino en la Internet Móvil.

E-OTD - (Enhanced Observed Time Difference)

Estos sistemas operan colocando receptores de señal o faros de referencia en la red móvil como unidades de medida en varios sitios geográficamente distribuidos. Cada uno de estos receptores tiene un reloj de precisión que cuando una señal es recibida por al menos tres receptores, las diferencias de tiempo en la llegada de la señal son calculadas. Luego, las diferencias de tiempos son combinadas para producir líneas hiperbólicas desde las cuales la ubicación es estimada.

Basadas en Red

COO - (Cell Of Origin)

En este esquema, el área cubierta por la célula de la estación base de la red de telefonía es usada para ubicar la localización del llamador. Por lo tanto, la precisión generalmente depende del tamaño de la célula, pero precisiones de hasta 150 metros pueden ser logradas.

TOA - (Time Of Arrival)

En forma similar al E-OTD, la diferencia observada en el tiempo de llegada de la señal a tres receptores base es la forma de localización.

AOA - (Angle Of Arrival)

La versión más común de esta técnica es conocida como Búsqueda de la Menor Dirección de Apertura, que requiere una compleja red de antenas en muchas de las células. Estas antenas trabajan juntos para determinar el ángulo de llegada de la señal. De cada antena receptora del ángulo de la señal, se puede determinar la ubicación de donde se originó la llamada.

³⁶ Por mayor información ver [Mobile, SistLoc].

ANEXO B: GPS

Introducción

Este Anexo pretende dar una descripción general de los sistemas de posicionamiento global, o GPS de sus siglas en inglés. Se comenzará definiéndolo para luego pasar a describir las técnicas utilizadas comúnmente, junto con las precisiones obtenidas de tales mediciones. Luego se darán las ventajas del GPS frente a las demás tecnologías de posicionamiento existentes. Finalmente se mostrará un diagrama comparativo de las diferentes opciones tecnológicas dentro de los GPS, comparando su técnica, prestaciones, precisiones y hasta sus costos.³⁷

El motivo para incluir un apartado exclusivo de GPS es que a lo largo del presente proyecto, se ha hecho hincapié en como esta tecnología podía ser utilizada junto con la solución propuesta, y cómo ésta aparecía como la candidata ideal para los propósitos aquí descritos.

¿Qué es el Sistema GPS?

El sistema de posicionamiento global (GPS) es una tecnología que le permite a un usuario, obtener su posición las 24 hs. del día en cualquier punto de la Tierra. El mismo fue desarrollado por el Departamento de Defensa de los Estados Unidos, pero su uso se ha extendido también al ámbito civil. El rango de precisión de una posición va de los 30 mts a unos pocos mm, dependiendo del equipamiento y las técnicas utilizadas.

¿Cómo Funciona?

El sistema GPS está formado por una constelación de 24 satélites, que orbitan la Tierra a una altura de 20200 kilómetros, emitiendo constantemente señales de radio. El receptor GPS calcula su posición, efectuando mediciones de distancia a cuatro (4) o más satélites. La distancia individual a un satélite es determinada en función del tiempo que tarda en viajar la señal desde el satélite al receptor y su velocidad de propagación. La posición del satélite es conocida para el receptor. Usando luego una geometría relativamente simple, este último determina las coordenadas del punto relevado.

Técnicas y Precisiones

Posicionamiento Autónomo

El posicionamiento autónomo (con un solo receptor) tiene una precisión que oscila entre los 10 y los 30 m. Los receptores autónomos son de bajo costo y muy utilizados en la navegación deportiva.

Corrección Diferencial

Técnica que permite obtener las coordenadas de un punto por debajo de los 3 metros en planimetría y menor precisión en altimetría, mejorando ambas de acuerdo al tipo de receptor utilizado. El método se basa en la corrección de todas las posiciones tomadas (calculadas con un receptor fijo en un punto conocido), que luego son aplicadas a un receptor itinerante.

Receptores Geodésicos

Los receptores geodésicos son equipos de alta complejidad, que permiten obtener precisiones que van del rango de los 3 cm a unos pocos mm tanto en planimetría como en altimetría. La distancia a los satélites, a diferencia de los demás métodos, es calculada en función de la fase de la onda portadora que envían los satélites.

Diferencia con los Métodos Tradicionales

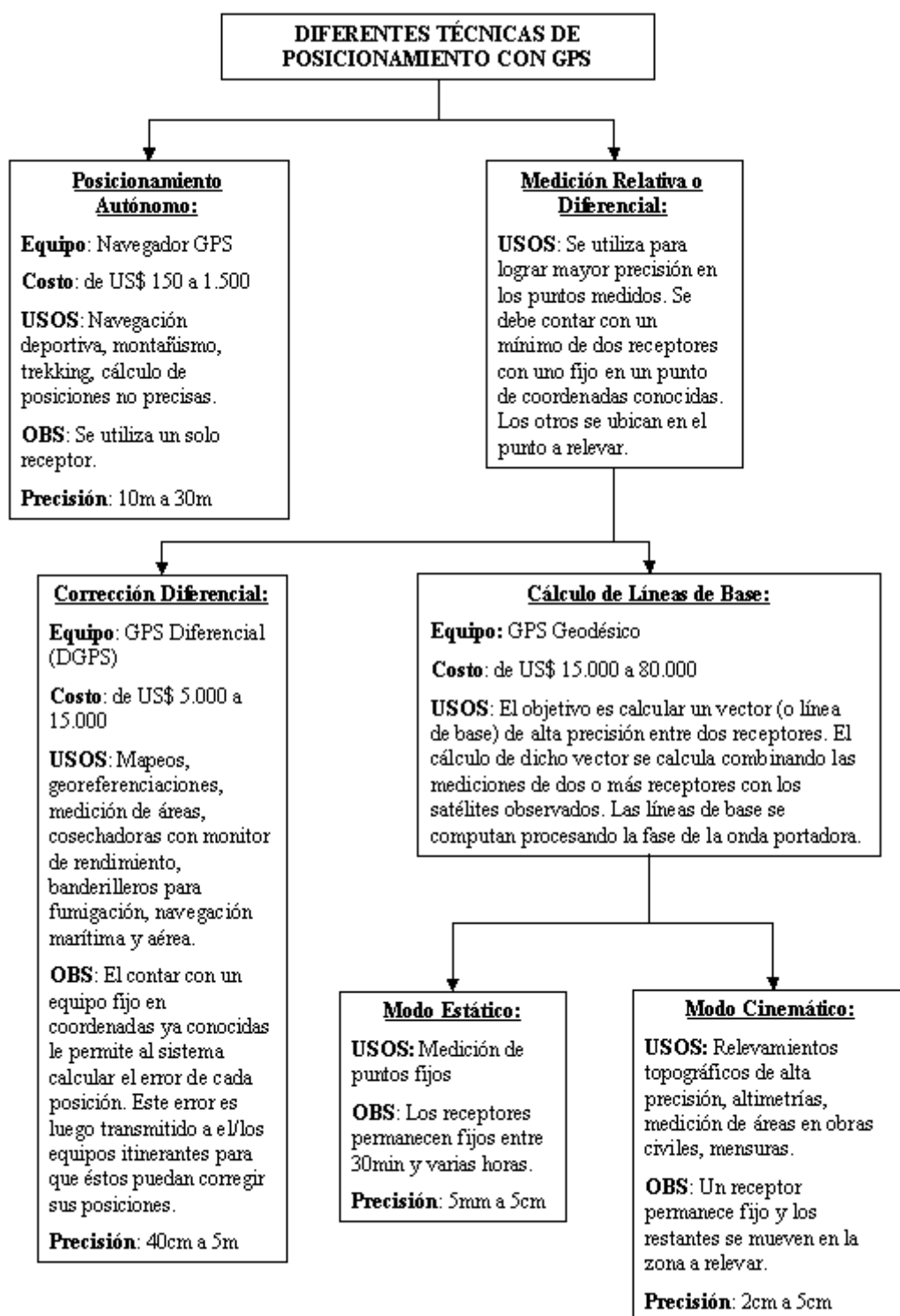
Ciertamente la medición con GPS tiene algunas ventajas sobre otras técnicas:

³⁷ Por mayor información ver [GPS].

- No tiene requerimientos de visual entre la estación base y el receptor itinerante. Hasta el advenimiento del GPS, la intervisibilidad era un gran factor limitante en cualquier práctica de medición.
- Permite realizar mediciones dinámicas (por Ej. con un vehículo en movimiento).
- Cada punto relevado es una medición independiente, por lo tanto no existe arrastre de errores.
- El GPS puede utilizarse prácticamente bajo cualquier condición climática.

Todo esto produce un dramático impacto en la productividad, eficiencia y precisión. El GPS es en este momento la forma más veloz, económica y precisa, que existe de medir.

Diferentes Técnicas Utilizando la Tecnología GPS



ANEXO C: Estado del Arte en Reconocimiento del Habla

Este Anexo pretende dar al lector una idea general del estado del arte en tecnologías de reconocimiento del habla, con particular énfasis en el reconocimiento de voz, ya que es el área de mayor interés en el presente proyecto, y es en donde están los mayores esfuerzos y donde se suscitan las mayores dificultades hoy en día³⁸.

Introducción

El concepto de utilizar la voz para interactuar con diferentes elementos del mundo, no es en lo absoluto nuevo para los seres humanos. Es más, el habla ha sido un elemento fundamental para el desarrollo humano. Basta con leer lo que autores como Maturana³⁹ han dicho: “El homo sapiens no se volvió sapiens por el desarrollo de su intelecto, sino por el desarrollo de su lenguaje. Es el lenguaje y su progresiva sofisticación lo que produjo el desarrollo intelectual, y no al revés”.

Sin ánimo de entrar en discusiones filosóficas, lo que importa destacar aquí es que el lenguaje, y en particular el lenguaje hablado, es de extrema naturalidad para los seres humanos, y que efectivamente es la “interfaz por excelencia”. Esto justifica utilizar sistemas de reconocimiento del habla que den soporte a variados servicios actualmente realizados por operadores o no realizados, ya que el costo de contar con operadores humanos sería demasiado alto (imaginemos la diferencia en costos de contar con un sistema automático de soporte de llamadas telefónicas que funciona las 24 horas del día, los 365 días del año, versus hacer frente a los costos que conlleva implementar el mismo sistema, con el mismo nivel de soporte, pero utilizando operadores humanos).

Últimos Avances

A continuación se presentan los últimos avances efectuados en las diferentes áreas de interés de las tecnologías de reconocimiento del habla, las que pueden separarse en: reconocimiento de voz natural; conversión texto-voz; y verificación del locutor⁴⁰:

Reconocimiento de Voz Natural

Esta área es la que sin dudas presenta el mayor avance en los últimos años dentro de lo que se llaman las áreas del reconocimiento del habla. El concepto de reconocimiento de voz **natural**, implica que el usuario sea capaz de hablar en forma continua y fluida con el reconocedor como lo haría con otra persona, y que éste le entienda. Junto con el avance que representa haber logrado un sistema con tales características, además se le suman las siguientes:

- *Técnicas de reducción del efecto de ruidos*, muy relevante en ambientes con altos niveles de ruido, típicos de la telefonía;
- *Seguimiento del locutor*, lo que le permite al reconocedor centrarse en el locutor que le está hablando aislándolo del resto de los sonidos; y
- *Soporte multicodeificación*, lo que permite utilizar casi cualquier tipo de teléfono y red por parte del usuario.

Conversión Texto-Voz

A los conversores multiidioma de alta calidad existentes en la actualidad, se les ha incorporado una serie de características entre las que se incluyen las siguientes:

- *Corrección automática de texto del correo electrónico*,
- *Deletreo y lectura adaptada de términos de Internet*,
- *Detección automática del idioma de un texto*, y
- *Lectura de documentos ofimáticos (Word, Excel, PowerPoint, etc.)*

³⁸ Por mayor información ver [EstadoVoz].

³⁹ Biólogo Chileno ...

⁴⁰ Por mayor información ver [Rec1].

Verificación del Locutor

Estos sistemas son capaces de decidir con *total certeza* si una persona es quien dice ser, con tan sólo escuchar un “login” o una “password”, o cualquier palabra pronunciada por el usuario.

Cómo Funciona un Reconocedor de Voz

Esta sección pretende dar una idea básica y general de cómo funciona un software típico de reconocimiento de voz⁴¹. En especial se tomó como base para la explicación, el reconocedor de voz desarrollado por Telefónica España (ver referencias citadas).

Antes que nada, cabe recordar que todos los software de reconocimiento de voz vistos en la etapa de investigación de tales herramientas, mostraron que absolutamente todos trabajan con el mismo sistema. El sistema es que el software solo devuelve palabras que estén en su diccionario. Es decir, que solamente devolverá un texto que se aproxime a lo que el usuario dictó, pero compuesto de palabras pertenecientes al diccionario de dicho software. Esto es así ya que para cada palabra, el programa devuelve la palabra de su diccionario que tiene la mayor probabilidad de ser la correcta, es decir, la que suene fonéticamente más similar a la dicha por el usuario.

Por lo tanto, como no se asigna probabilidad alguna a palabras que no estén en el diccionario, las respuestas del programa están limitadas al vocabulario que este conoce. Si bien este vocabulario no es estático, sino que se le pueden agregar palabras (“enseñarle” palabras), las nuevas palabras deben ser agregadas mediante una funcionalidad específicamente hecha para esto, y no es capaz de aprenderlas mientras el usuario las dicta de manera de inmediatamente devolverlas como palabras válidas de su diccionario. Al menos en este proyecto no se encontraron referencias de productos o desarrollos que lo hicieran posible.

Volviendo al análisis del reconocedor típico y genérico (basado en el de Telefónica I+D), primero se analizará la información que utiliza el reconocedor para obtener la secuencia de palabras que con más probabilidad corresponde a la pronunciación del locutor. Esta información es:

- **El Diccionario:** es el conjunto de palabras que admite el reconocedor de voz. Fuera de este diccionario no se reconoce palabra alguna, como ya se mencionó.
- **El Modelo Acústico:** que permite calcular para cada palabra del diccionario, la probabilidad de que se esté pronunciando en un determinado momento de la frase. Dado que es imposible generar modelos para cada palabra del idioma español, se construyen modelos más elementales que representan los trifenemas (definidos más adelante). Así, los modelos de las palabras del diccionario se construyen por concatenación de estos trifenemas.
- **El Modelo de Lenguaje:** que asigna a cada secuencia de palabras una determinada probabilidad de aparición dentro de una frase dictada.

Un *trifenema* es una unidad acústica que representa a un fonema en función de cual es su predecesor y su sucesor. Por ejemplo, la palabra “casa” tendrá como trifenemas: c(\$,a), a(c,s), s(a,a), a(s,\$) donde las letras fuera del paréntesis representan el fonema central, y las de dentro el fonema anterior y posterior, respectivamente. El símbolo \$ indica que puede ser cualquier otro fonema que preceda o siga a esta palabra. Es decir, que cada trifenema indica qué sonido se asocia a un determinado carácter, dependiendo de su carácter anterior y posterior (su contexto local).

Esta granularidad se hace necesaria, como se mencionó, dada la imposibilidad de almacenar los modelos acústicos de cada palabra del idioma español. Por esto, se mantienen modelos de trifenemas, los cuales combinados entre sí (concatenados) dan lugar a la fonética de las palabras.

Es a partir de toda esta información que el reconocedor realiza un complejo proceso de búsqueda con el fin de obtener la secuencia más probable de palabras pronunciadas. La forma en que los reconocedores analizan lo dictado por el usuario es a través de pequeñas frases, y no de palabras aisladas. Es decir, que el reconocedor no intenta traducir palabra por palabra (imaginemos un algoritmo tipo “greedy” -ávido-), sino que por el contrario, espera aproximadamente hasta que el usuario haga la pausa necesaria para tomar aliento.

⁴¹ Por mayor información acerca del funcionamiento de un reconocedor de voz ver [Rec1, Rec2, Rec3, Brown].

Recoger una buena cantidad de palabras es importante para poder utilizar el Modelo de Lenguaje, ya que de esta manera se puede asignar una mejor probabilidad a cada palabra basándose en la probabilidad de aparición dentro de una determinada secuencia de palabras, cosa imposible de hacer si se tradujera palabra a palabra.

Ahora, luego de saber qué información maneja un reconocedor, y antes de pasar a su complejo funcionamiento, se mostrará cuales son las estructuras mediante las que almacena esa información:

- **El Diccionario:** es una estructura de árbol de trifenemas, en lugar de tratarse de una simple lista de palabras. El motivo es de eficiencia, ya que palabras que comienzan igual, comparten los mismos trifenemas.
- **El Modelo Acústico:** es común utilizar modelos acústicos de *trifenemas* (3) solamente, como ya se mencionó.
- **El Modelo de Lenguaje:** dado que es imposible estimar las probabilidades de aparición de secuencias de más de tres palabras juntas, se utiliza la probabilidad de la combinación de tres palabras, o en caso de no disponerse, de dos palabras (trigramas y bigramas respectivamente). Como ejemplo, para un diccionario de 30.000 palabras hay 27 billones de secuencias de tres palabras.

A continuación se explica el funcionamiento típico de un reconocedor. Se basará la explicación en el siguiente diagrama mostrando la arquitectura de un sistema de diálogo:

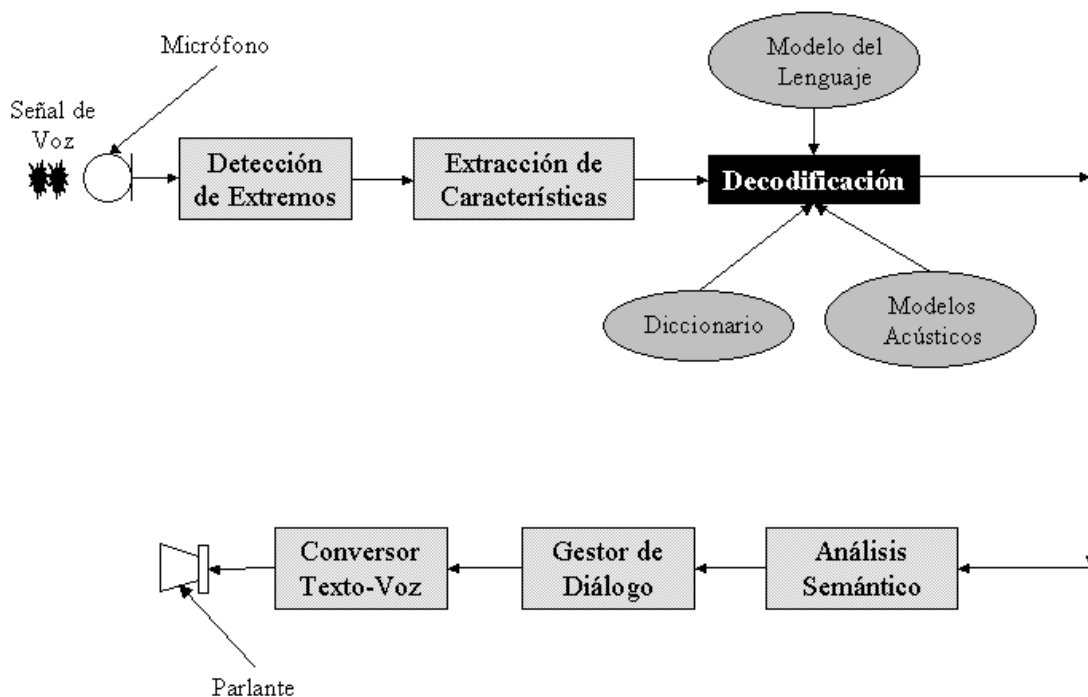


Diagrama de bloques de un sistema de diálogo

Actividades:

1. La señal de voz que entra por el micrófono es convertida en una señal eléctrica analógica es luego convertida en secuencias de bits (es decir, digitalizada).
2. El Detector de Extremos es quien detecta la presencia de voz y la pasa al siguiente bloque.
3. El Extractor de Características calcula una serie de parámetros de la señal de voz que tienen información relevante para el proceso de reconocimiento.
4. Estos parámetros se pasan al Decodificador, el cual se apoya en los modelos acústicos, del lenguaje y el diccionario (explicados antes) para generar la frase reconocida.
5. Posteriormente, el Analizador Semántico extrae el significado de la frase, que será utilizado por el Gestor de Diálogo para, en función del estado de la conversación, tomar la decisión más adecuada y hacer una predicción sobre la siguiente interacción con el usuario.
6. Finalmente, el Conversor Texto-Voz convierte en sonido la respuesta del sistema.

El módulo de Gestor de Diálogo, que no fue introducido antes por tratarse de un módulo que no siempre está presente en todo reconocedor de voz, es el encargado de, en lo posible, predecir sobre el contenido de la siguiente frase que pronunciará el locutor.

Diferentes Tipos de Reconocedores de Voz

Dado que el problema general del reconocimiento de voz aún no está totalmente resuelto, existen muchos tipos de reconocedores de voz diferentes. Además de esto, la clasificación que sigue sirve para comparar efectivamente dos reconocedores, cuando realmente corresponda tal comparación, es decir cuando pertenezcan a la misma clasificación, y cuando las pruebas se realicen bajo las mismas condiciones (ya sean de laboratorio o de campo).

Según el Número de Locutores que Reconozcan

Éstos pueden ser:

- *Dependientes del locutor.* Sólo reconocen la voz de la persona para la que han sido entrenados.
- *Multilocutor.* Reconocen la voz de un conjunto pequeño de personas.
- *Independiente del locutor.* Reconocen la voz de cualquier persona.

Según el Tamaño del Vocabulario que Reconozcan

Éstos pueden ser:

- Reconocedores de vocabularios *pequeños*: hasta 40 palabras.
- Reconocedores de vocabularios *medios*: hasta 400 palabras.
- Reconocedores de vocabularios *grandes*: hasta 4.000 palabras.
- Reconocedores de vocabularios *muy grandes*: hasta 40.000 palabras.
- Reconocedores de vocabularios *ilimitados*: más de 40.000 palabras.

Según el Canal

Éstos pueden ser:

- Reconocedores a través de *micrófono*.
- Reconocedores para la *red telefónica* (fija, móvil analógica o móvil digital).

Según el Tiempo de Respuesta

Éstos pueden ser:

- Reconocedores en *tiempo real*. Son reconocedores que dan una respuesta lo suficientemente rápida como para que un usuario pueda interactuar con él
- Otros reconocedores. El tiempo de respuesta no es un factor importante (por Ej. sistemas de reconocimiento para la transcripción de informes)

Características de un Reconocedor de Habla Natural

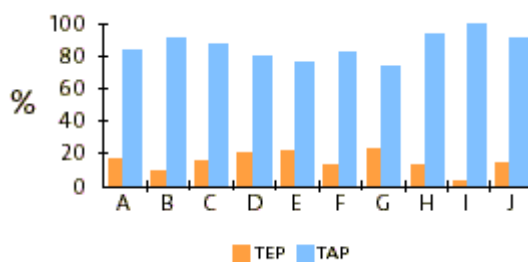
Esta Sección tiene como objetivo la presentación de las características que afectan el funcionamiento de los reconocedores de voz, y en particular, de los reconocedores de habla natural.

Independencia del Locutor

En el transcurso de un diálogo entre seres humanos, existen una infinidad de factores que se revelan al momento de hablar, que van más allá del propio mensaje que se está diciendo. Cosas como el acento de quien habla, el ímpetu puesto en ciertas palabras o frases, si se trata de un hombre o una mujer, de un adulto o de un niño, etc.

Este tipo de elementos son de gran aporte al momento de entender el diálogo para un ser humano, pero sin duda no ocurre lo mismo cuando es un sistema automático el que debe interpretar este diálogo. Para éste, más que una ayuda, son una complicación. Esto quiere decir, que las diferencias entre las distintas voces tiene un efecto negativo en la tasa de aciertos.

La siguiente figura muestra justamente esto. Para diez locutores diferentes, se muestran los porcentajes de acierto (TAP) y de error (TEP), los cuales sumados para cada locutor forman el 100%⁴².



Las diferencias en las tasas de error se deben a que los modelos acústicos que maneja un reconocedor para poder realizar el proceso de reconocimiento se obtienen, en el proceso de entrenamiento, a partir de un conjunto finito de voces de muchas personas, de modo que:

- El sistema no funcionará bien con aquellas voces que sean muy diferentes a las empleadas en el entrenamiento,
- Los modelos acústicos modelan una voz “promedio”, resultado de procesar las voces del conjunto, por lo que tampoco voces parecidas a las de dicho conjunto de prueba serán reconocidas con facilidad.

Aparte del modelo acústico, el modelo de lenguaje y el diccionario también afectan las tasas de reconocimiento: si la persona que utiliza el sistema pronuncia frases que están bien modeladas por el modelo del lenguaje, y cuyas palabras estén incluidas en el diccionario, el reconocedor funcionará mejor que en el caso contrario. Por lo tanto, se necesita dotar al reconocedor de técnicas de adaptación al locutor que permitan que el sistema pueda modificar dinámicamente los modelos acústicos, el diccionario y el modelo del lenguaje.

Efectos del Canal y del Ruido

Está demostrado por experimentos⁴³ que el ruido de fondo afecta relativamente poco a las tasas de reconocimiento de los seres humanos. Los motivos de esto son los siguientes:

- *Las personas tenemos dos oídos*, lo que nos permite la identificación de las fuentes de sonido y su separación.
- *La capacidad de predicción* del cerebro, apoyándose en una serie de fuentes de conocimiento, como el propio conocimiento del lenguaje de las personas y el contexto y tema de la conversación.
- *La capacidad de adaptación* del cerebro al ruido.

Desde el punto de vista del reconocedor de voz, ruido es cualquier señal acústica o eléctrica que contamine la señal de voz que queremos reconocer. Sabiendo esto, podemos clasificar este ruido en tres grandes grupos:

- *Ruido estacionario*. Son todos aquellos ruidos cuyas propiedades se mantienen constantes a lo largo del tiempo. Su principal característica es que se puede identificar (porque ocurre por un período largo de tiempo) y se puede predecir su comportamiento futuro, pudiéndose en algunos casos cancelarse.
- *Ruido no estacionario*. Todos los ruidos que no están incluidos en el punto anterior, principalmente los ruidos impredecibles y variados. La mayoría de los ruidos del mundo real, desafortunadamente son de este tipo, los cuales son muy difíciles de detectar y mucho más de corregir.

⁴² Estas pruebas fueron efectuadas sobre el reconocedor de habla continua de Telefónica en su versión de 1996.

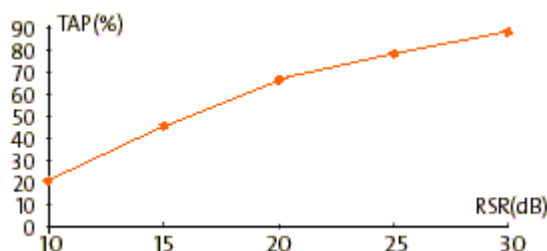
⁴³ Por mayor información ver [Lippmann].

- *Voces de fondo.* Estas voces degradan las tasas de reconocimiento y además ocurre que el reconocedor frecuentemente reconoce la voz de fondo en lugar de la del locutor actual.

Para el caso de reconocimiento por línea telefónica (nuestro caso), el problema del ruido se agrava por la influencia del canal telefónico. Por *canal* se entiende el conjunto de todos los elementos que hay entre el locutor y el reconocedor:

- el micrófono,
- la línea telefónica,
- diversos circuitos eléctricos y electrónicos por los que pasa la voz,
- la codificación que se emplea en la telefonía móvil como en la IP modifica las propiedades de la señal de voz.

La siguiente figura muestra el efecto del ruido estacionario en las tasas de reconocimiento de un reconocedor de palabras aisladas por línea telefónica cuyo vocabulario es de 500 palabras con un alto nivel de confusión (muy similares entre sí). En este experimento⁴⁴ no se efectuó ninguna técnica de compensación de ruido estacionario. El eje X muestra la relación señal/ruido (RSR, en dB), mientras que el eje Y muestra la tasa de acierto de palabras (TAP, en %).



Como se observa, la tasa de error aumenta (TAP disminuye) cuando la relación señal ruido disminuye (más ruido sobre la señal).

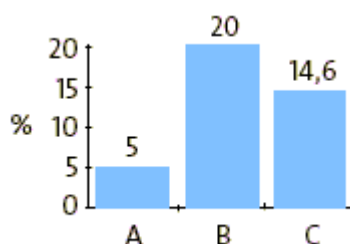
Independencia del Dominio Semántico

De todo lo visto hasta ahora, se desprende que el reconocimiento de voz a través de una línea telefónica, se enfrenta con una serie de dificultades que limitan seriamente las posibilidades de utilización de estas tecnologías en servicios reales, como el presentado en este trabajo. Por un lado, el problema de la independencia del locutor, que no está completamente resuelto (ver punto Independencia de Locutor), y por otro lado, el problema del canal y del ruido, que degrada las tasas de reconocimiento (ver punto Efectos del Canal y del Ruido).

Debido a que ninguno de estos problemas está totalmente resuelto en la actualidad, es aconsejable:

- Reducir el vocabulario que maneja el reconocedor de voz, a fin de disminuir la probabilidad de que confunda unas palabras con otras,
- Diseñar diálogos que guíen al usuario para que diga las cosas en la forma en que el reconocedor está esperando,
- Reestimando los modelos del lenguaje tan pronto como se disponga de diálogos provenientes de la interacción real con el locutor.

Con respecto al dominio semántico, o al dominio del problema, es interesante ver las tasas de error de un reconocedor de habla continua con un vocabulario, para la gráfica siguiente, de 5000 palabras, bajo tres experimentos diferentes: a) evaluación del reconocedor en la misma tarea para la que se lo había entrenado; b) evaluación realizada en una tarea distinta; y c) evaluación en la tarea distinta pero aplicando un método de adaptación del modelo del lenguaje⁴⁵:



⁴⁴ Experimento realizado por Telefónica I+D. Por mayor información ver [EstadoVoz].

⁴⁵ Adaptación efectuada por Telefónica I+D en su Reconocedor de Habla Continua. Por mayor información ver [EstadoVoz].

Como se puede apreciar, la técnica de adaptación reduce (en este caso) en un 27 por ciento la tasa de error, empleando tan solo unos pocos cientos de frases⁴⁶.

Características del Habla Espontánea

Si bien dejada para el final, es de las más importantes características que debe tener presente un reconocedor de habla continua. El habla espontánea o natural es el tipo de habla que empleamos los seres humanos al comunicarnos entre nosotros, y presenta las siguientes características que la hacen especial:

- Los sonidos aparecen pobremente articulados con relativa frecuencia,
- Se coarticulan muchos sonidos,
- La velocidad del habla es variable,
- Suelen aparecer correcciones, falsos comienzos y sonidos guturales,
- No sigue estrictamente las reglas del lenguaje,
- Un mismo idioma puede tener una multitud de acentos/dialectos.

Los efectos que estas características tienen sobre la señal de voz son:

- Muchos sonidos desaparecen o pierden parte de sus propiedades acústicas,
- Hay sonidos que no pertenecen al idioma “normativo”, pero sí a un dialecto del mismo,
- Los modelos del lenguaje funcionan peor que con el estilo de habla leído, ya que las correcciones, falsos comienzos y sonidos guturales reducen la capacidad de corrección del modelo del lenguaje.

Estos últimos tres problemas constituyen los principales retos que tiene la tecnología de reconocimiento de voz en nuestros días⁴⁷. Las soluciones de estas cuestiones, lejos de ser responsabilidad de una única disciplina de la ciencia, se alcanzarán como resultado de la sinergia de múltiples áreas de conocimiento.

Servicios de Voz

Esta Sección describirá familias de servicios que el presente estado del arte de la tecnología permite tratar como servicios automáticos de voz, y por tanto, permite ofrecerlos al usuario de la forma en que le resulta más cómoda: mediante una interfaz vocal.

Portales de Voz⁴⁸

La idea de un portal de voz es simple: comunicar a través de la voz los contenidos y servicios que están disponibles en Internet. Esto expande los conceptos de ISP, portal, buscador, navegador⁴⁹, etc. Este concepto de Portal de Voz permite relacionar el mundo de Internet con el mundo de la voz a través de un conjunto de servicios que amplíen, complementen y den valor agregado al acceso vocal a Internet. Por esto se incluyen servicios como lectura de emails por voz, realización de llamadas telefónicas y envío de emails a direcciones de una agenda, etc.

A través de los portales de voz se expande el concepto de acceso a Internet a los usuarios de teléfonos, ya sean fijos como móviles, fuera de los que ya accedían a estos: los PCs, y mas recientemente los usuarios de móviles con WAP.

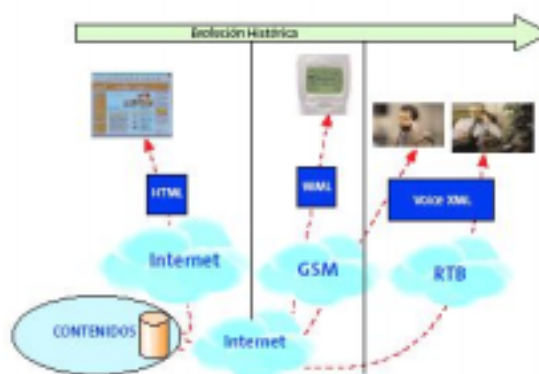
Por tratarse de un concepto que revoluciona el acceso a Internet, se presenta a continuación un breve esquema representando la evolución del acceso a Internet (HTML→WML→VoiceXML) en el tiempo (Evolución Histórica).

⁴⁶ Por mayor información ver [Adaptación].

⁴⁷ Comentario tomado de [EstadoVoz] con fecha de publicación Marzo 2001.

⁴⁸ Por mayor información ver [VoiceWeb1, VoiceWeb2, Mainstream].

⁴⁹ Aquí se refiere a Navegadores de Voz, o *Voice Browsers*. Por mayor información ver [VozUMTS, Voyager].



GSM aparece por ser la telefonía celular actual en España. Los portales de voz tienen como objetivos:

- La universalización del concepto de Internet,
- El incremento del tráfico telefónico,
- El incremento de la masa crítica de usuarios de acceso vocal a Internet por teléfono,
- La proliferación de nuevos negocios asociados: e-commerce, juegos interactivos, facturación, publicidad audible, etc.

Directorios Vocales

Este tipo de servicio apunta a buscar y extraer datos de una persona de una Base de Datos a partir de una descripción hablada de ésta. Este concepto es directamente aplicable a muchos servicios telefónicos tradicionales como lo son:

- El servicio de directorio telefónico (las Guías Telefónicas),
- El servicio de agenda corporativa de una empresa,
- El servicio de agenda personal.

Servicios de Información

La idea de estos servicios es la de ofrecer un acceso telefónico para proporcionar al cliente información de diversa índole. Es una de las ideas más explotadas por los servicios telefónicos tradicionales (información de noticias, tráfico, bursátil, servicios de empresas, etc.)⁵⁰

Tradicionalmente este tipo de servicio implicaba un gran costo, ya que por un lado, el servicio era ofrecido mediante operadores humanos, o por el otro, los contenidos vocales debían ser actualizados permanentemente mediante grabaciones de la voz humana.

Comercio Electrónico

Todos los análisis de mercado indican que el comercio electrónico será uno de los negocios que más dinero moverá en el futuro. Sin embargo, hasta la fecha son dos los motivos que no le han dejado cumplir sus expectativas, lo cual no quiere decir que no lo hará:

1. El número de personas que realmente tienen acceso a realizar este tipo de compras (actualmente, los que tienen acceso a Internet a través de un PC conectado a la red telefónica –o dispositivos portátiles),
2. La desconfianza en la seguridad de los sistemas.

Las tecnologías de voz pueden ayudar enormemente a vencer estas dos cuestiones:

- Realizando el pago mediante la voz: bastará con tener un terminal telefónico de cualquier tipo para poder realizar compras.
- Con la verificación del locutor: por medio del habla se puede generar en el cliente final una confianza total en la seguridad del sistema.

Asistentes Personales

Últimamente están proliferando los servicios que de una u otra forma ayudan al usuario a configurar, consultar, programar o administrar las distintas facilidades que le ofrecen los servicios (o incluso dispositivos) que utiliza.

La experiencia indica que el cliente final no suele hacer uso de este tipo de servicios, a menos que sus mecanismos de uso sean extremadamente sencillos. Que mejor entonces, que permitirle configurar sus servicios o dispositivos mediante el uso de su propia voz.

⁵⁰ Este tipo de servicio es en el que se enmarca el presente trabajo.