Universidad Austral de Chile
Facultad de Ciencias de la Ingeniería
Escuela de Ingeniería Civil en Informática

IMPLEMENTACIÓN DE UN SISTEMA DE RESPUESTA DE VOZ INTERACTIVA MULTIPLATAFORMA BASADOS EN UN ESTÁNDAR DE DESARROLLO DE ALTO NIVEL PROVISTO POR VOICEXML

TESIS DE GRADO PARA OPTAR AL
TITULO PROFESIONAL DE INGENIERO
CIVIL EN INFORMATICA

PROFESOR PATROCINANTE:
HERNAN LUIS VIDAL VIDAL

PROFESOR CO-PATROCINANTE:
WLADIMIR EDMUNDO RIOS MARTINEZ

RODOLFO FRANCISCO ROSAS ESCOBAR

VALDIVIA – CHILE
2003

Universidad Austral de Chile

Instituto de informática

Valdivia, 19 de mayo de 2003
Comunicación Interna N° 061/03

De : Luis Hernán Vidal Vidal.
A : Sra. Miguelina Vega R.
Directora de Escuela de Ingeniería Civil en Informática

Motivo: Informa Calificación Trabajo de Titulación.

Informar revisión y calificación del Proyecto de Título "Implementación de un Sistema de Respuesta de Voz Interactiva Multiplataforma basado en un estándar de desarrollo de alto nivel provisto por VoiceXML", presentado por el alumno Rodolfo Francisco Rosas Escobar., que refleja lo siguiente:

- Se logró el objetivo planteado de implementar un sistema de respuesta de voz interactiva basado en VoiceXML.

La revisión hecha sobre los estándares y tecnologías empleadas entregan una base referencial clara y de gran valor al momento de realizar tareas de integración de sistemas telefónicos y computacionales (CT1).

El trabajo de tesis presento una muy buena solución ante la gran diversidad de herramientas de desarrollo para aplicaciones (CTI), el desarrollo de una serie de prototipos entrega bastante información en cuanto a las capacidades de VoiceXML.

El desarrollo de una aplicación de cobro revertido, similar a una existente (en ejecución), permite apreciar los factores o condiciones para llevar VoiceXML a un ambiente de producción como sustituto eventual de sistemas tradicionales (propietarios).

Por todo lo anterior expuesto califico el trabajo de titulación del Sr. Rodolfo Rosas con nota 7,0 (siete como cero).

Sin otro particular, se despide atentamente

Prof. Ing. Luis Hernán Vidal Vidal.
Instituto de Informática.
Universidad Austral de Chile.

General Lagos 2086 · Campus Miraflores · Valdivia · Chile
Casilla 567 · Fono: 56 63 221427 · Fax: 56 63 293115 · instituto@inf.uach.cl · www.inf. uach.cl

Universidad Austral de Chile

Instituto de Informática

Ci No. 60/03

Fecha: 19/5/03

A: Miguelina Vega Rosales, Escuela Ingeniería Civil Informática

De: Wladimir RÍOS Martínez, Instituto de Informática

Asunto: Informe tesis alumno Rosas Escobar

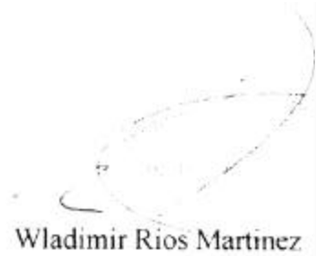
He revisado la tesis IMPLEMENTACION DE UN SISTEMA DE RESPUESTA DE VOZ INTERACTIVA MULTIPLATAFORMA BASADOS EN UN ESTANDAR DE DESARROLLO DE ALTO NIVEL PROVISTO POR VOICEXML, del alumno Rodolfo Francisco Rosas Escobar.

La tesis muestra un interesante componente de integración comunicación-informática que es el campo de acción de la llamada telemática, que ha dado lugar a nuevos perfiles de ingeniería en el Mundo y en Chile.

Hecho de menos un análisis mas profundo del problema de procesamiento de señales (DSP), que es imprescindible en este tipo de trabajo y que nuestra Facultad tiene los medios para haber trabajado en esos aspectos (Matlab y todas sus librerías asociadas a DSP). La incorporación en este tipo de trabajos de diseño de filtros es importante y hubiese contribuido a aclarar algunos aspectos de la página 34 que a pesar de la ejemplificación, queda difuso. Recomiendo una preparación en estos puntos para su examen de grado.

La evaluación de esta tesis es nota 6,5 (seis coma cinco)

Atentamente



Wladimir Rios Martinez
Dirección: General Lagos 2086, Campus Miraflores, Valdivia
Fono: (63)- 221427 – Fono-Fax : (63)- 293115
e-mail: wrios@inf.uach.cl

VALDIVIA, 19 DE MAYO DEL 2003

DE: GLADYS MANSILLA GOMEZ
A : DIRECTORA DE ESCUELA INGENIERIA CIVIL EN INFORMATICA

MOTIVO

INFORME TRABAJO DE TITULACION

Nombre Trabajo de Titulación: "IMPLEMENTACION DE UN SISTEMA DE RESPUESTA DE VOZ INTERACTIVA MULTIPLATAFORMA BASADO EN UN ESTANDAR DE DESARROLLO DE ALTO NIVEL PROVISTO POR VOICEXML

Nombre del alumno: RODOLFO FRANCISCO ROSAS ESCOBAR.

Nota: 7.0 (en números) siete (en palabras)

Fundamento de la nota:

-Este trabajo de tesis constituye una aplicación útil para quienes desean realizar quieren autenticación por voz.

-En la realización de este trabajo de titulación se alcanzan plenamente los objetivos planteados al inicio.

- La presentación y redacción del informe están bien elaboradas, abarcando tópicos que inciden directamente en esta tesis y expresado en un lenguaje formal apropiado.

- El alumno ha sabido introducirse en un tema nuevo, estudiarlo e implementar una aplicación.

- Es también valorable que como producto de este trabajo, el alumno haya presentado un trabajo en un congreso del área.

GLADYS MANSILLA GÓMEZ
DOCENTE INSTITUTO DE INFORMATICA

Porque el camino recorrido fue largo

Agradezco a mis padres Gustavo y Cristina, por el apoyo que me brindaron en todo momento. Ahora se ven los frutos de su confianza.

A mi hermana Alicia, por ser una fuerte inspiración de este logro.

A mis hermanos Gustavo y Jorge, por su apoyo incondicional y comprensión.

A mi amada Caroline, por ser un pilar fundamental en este triunfo, que no hubiera sido posible sin su apoyo, comprensión y amor.

INDICE

RESUMEN

Hoy en día los medios de transferencia de información van más allá del uso cotidiano del teléfono y la Internet. VoiceXML (Voice Extensible Markup Language) ofrece una ventana de posibilidades para la implementación a un alto nivel de sistemas de telefonía o servicios sobre Internet.

El desarrollo de aplicaciones, especialmente las relacionadas con servicios de respuesta de voz interactiva, en la actualidad no están normados; la mayoría de los fabricantes o proveedores de tecnología ofrecen suites de herramientas propietarias, de esta forma el desarrollo de nuevos servicios es determinado por la capacidad de las herramientas utilizadas, y en la mayoría de los casos la portabilidad de las aplicaciones o servicios no existe.

Debido a la necesidad de establecer estándares para el desarrollo de alto nivel hoy en día se esta potenciando VoiceXML, el cual permite desarrollar nuevos servicios para múltiples plataformas independientes del proveedor.

SUMMARY

Today the means of information transference go beyond of the daily use of the telephone and the Internet. VoiceXML (Voice Extensible Markup Language) offers a window of possibilities for the implementation to a high level of telephony systems or services on Internet.

The development of applications, specially the related ones to services of interactive voice response, at the present time are not standar; most of the manufacturers or suppliers of technology offers suites of propietary tools, of this form the development of new services is determined by the capacity of the used tools, and in most of the cases the portability of the applications or services it does not exist.

Due to the necessity to nowadays establish standards for the development of high level this harnessing VoiceXML, which allows to develop new services for multiple independent platforms of the supplier.

1 INTRODUCCIÓN

1.1 Antecedentes Generales

La introducción de las nuevas tecnologías en los equipos y sistemas de telecomunicaciones, basados cada vez más en equipos multimedia y/o interactivos, exigen la adaptación de la actividad a nivel un mayor profesional y de programación y/o adaptación de los programas que gestionan dichos equipos.

La integración de los sistemas informáticos y de telecomunicaciones y, por tanto de las tecnologías que los soportan, requieren de un profesional con competencias mas transversales desde el punto de vista tecnológico y de programación, donde se combinan elementos y sistemas electrónicos propios de las telecomunicaciones e informáticos, demandándole una visión sistemática y pluridisciplinar en constante evolución. La seguridad, fiabilidad y calidad exigida a estos sistemas adquiere cotas que solo mediante un nivel de alta cualificación y profesionalidad se deben afrontar.

Como implementar sistemas de respuesta de voz interactiva sobre arquitecturas telefónicas, web o híbridas basados en un modelo estándar independientes del hardware o software empleados, será el centro de investigación de este trabajo.

Se investigó el concepto de implementación de un sistema de respuesta de voz interactiva multiplataforma, basados en un estándar de desarrollo de alto nivel provisto por VoiceXML y se enfatizaron los conceptos de capacidades del lenguaje, soporte de la tecnología, documentación y diseño bajo dicho estándar.

El trabajo de tesis ha sido fortalecido, al ser contrastado con experiencias practicas, obtenidas al desarrollar y mantener sistemas de respuesta de voz interactiva (sobre plataformas propietarias), en la empresa Telefónica del Sur.

1.2 Objetivos a Lograr

1.2.1 Objetivo General

Implementar un sistema de respuesta de voz interactiva, sobre Web y sistemas telefónicos basados en VoiceXML como estándar para desarrollo de alto nivel, todo lo anterior independiente de las plataformas de hardware o software empleados.

1.2.2 Objetivos Específicos

Describir la arquitectura empleada por VoiceXML.

Describir conceptos básicos para la implementación de sistemas de respuesta de voz interactiva que utilicen VoiceXML basados en los estándares existentes.

Determinar casos de uso relevantes sobre las arquitecturas, en las cuales emplear VoiceXML: web y telefonía.

Construir un prototipo de servicio de respuesta de voz interactiva empleando VoiceXML, basado en los casos de estudio más relevantes.

Desarrollo de pautas que permitan la implementación de sistemas web, telefonía o híbridos basados en VoiceXML.

1.2.3 Estructura de esta Tesis.

El trabajo de tesis se ha estructurado de la siguiente forma:

Capítulo 1. Introducción.

Capítulos 2 y 3. Se discute el estado del arte, primero en cuanto a la plataforma de comunicación telefónica sobre la cual se sustenta la tecnología y segundo se entrega las bases de la integración de sistemas de telefonía y computadores, basado en estándares utilizados.

Capítulos 4 y 5. En estos capítulos se abarcan dos áreas importantes de esta tesis, primero se entregan las bases del lenguaje, evolución de este y sus alcances, y la nomenclatura de programación definidas para VoiceXML, segundo se muestran distintos tipos de aplicaciones desarrolladas bajo este estándar para demostrar algunas de las potencialidades del lenguaje dados los recursos disponibles para su desarrollo.

Capítulo 6. En este capitulo se entregan las directrices básicas necesarias para la evaluación e implementación de una plataforma de VoiceXML como alternativa a la migración de una plataforma de respuesta de voz interactiva tradicional.

Capítulo 7. Conclusiones.

2 SISTEMAS TELEFONICOS.

2.1 Introducción.

La red telefónica constituye la principal red de telecomunicaciones, permite que usuarios en cualquier parte del mundo puedan hablar unos con otros de una manera muy sencilla, y además, se puede utilizar para la comunicación de datos. Constituida por centrales y medios de transmisión, evoluciona desde una infraestructura analógica a otra digital, lo que permite ofrecen nuevos servicios a un costo reducido.

El dispositivo que genero el mayor desarrollo en el sector de las telecomunicaciones es el teléfono, cuya invención, es obra del italiano Antonio Meucci, ver figura 2.1.1 (reconocido por la Cámara de Representantes de los Estados Unidos), aún cuando la patente esta a nombre de Alexander Graham Bell.

Fig. 2.1.1 Antonio Meucci. [Axxon, 2002]

2.2 Necesidades.

El mercado de las telecomunicaciones se ha extendido a todo el mundo y se ha transformado desde sus inicios; este crecimiento se ha producido en gran medida como respuesta a los nuevos requerimientos de servicios cada vez más avanzadas por parte de la población (usuarios), todo esto muy ligado al desarrollo de nuevas tecnologías (telecomunicación y datos).

Las necesidades del mercado son múltiples, se podrían dar algunos ejemplos como: conocer los movimientos de la bolsa, el clima o el saldo de una cuenta bancaria, y no termina ahí pues las capacidades de desarrollo y prestación de nuevos servicios son ilimitadas.

2.3 Aplicaciones.

Si pensamos en aplicaciones posibles de desarrollar, el rango es muy amplio y seria difícil denotarlas todas, por esto a continuación se señalan algunas aplicaciones de interés:

- Un servidor Web que contenga un sistema de voz que grabe las preferencias sobre los artículos que se desea obtener información para que posteriormente el usuario llame y reciba la información que el realmente desea conocer. Las aplicaciones de recuperación de información pueden proporcionar noticias, deportes, tráfico, tiempo, información de las acciones de la bolsa, así como información más especializada (por ejemplo, las noticias de la compañía basadas en la Intranet). El servicio puede ser utilizado por suscripción, anuncio, o por tiempo de conexión [Voicexmlforum, 2001].
- El comercio electrónico es otra área. Las aplicaciones de orden por catalogo funcionarían bien, debido a que la voz lleva mucho menos información que los gráficos, y esa información debe fabricarse en serie en un solo stream de audio. Esto funcionaria bien si el cliente esta trabajando sobre un catalogo impreso (pedido por correo, o conoce exactamente las características del producto (por ejemplo un libro en particular, CDs, videos, entradas a conciertos o deportes, etc.) o si los productos pueden describirse en una frase o dos y no hay muchos de ellos (Ej. productos de deportes). Las aplicaciones de servicio de Cliente (la ubicación física del paquete, el estado de cuenta, y centros de llamado) Las aplicaciones financieras, las citas bancarias, accionarías y de comercio [Voicexmlforum, 2001].

- Debido a los estándares de seguridad sobre la web integrando la voz sobre la web, las aplicaciones de Intranet pueden ser desarrolladas en Voice XML, éstas pueden ser control de inventario, ordenes de insumos, entregar servicios a recursos humanos, y portales corporativos [Voicexmlforum, 2001].
- Las aplicaciones de mensajeria unificada pueden unificar la voz. Los mensajes vía e-mail pueden ser leídos por teléfono, los e-mail salientes pueden ser grabados por teléfono, los sistemas de información orientados a la voz pueden ser sincronizados con los organizadores personales y sistemas de correo [Voicexmlforum, 2001].
- Existen muchas otras áreas en las cuales es útil implementar servicios de voz, como verificar el estado de ofertas en sitios de subasta electrónica, autorización de pago de cuentas, y otros que no podemos aun concebir.

2.4 Tecnologías empleadas en sistemas telefónicos.

Es necesario realizar un análisis de las principales tecnologías empleadas en los sistemas telefónicos actuales, esto debido a que la comprensión de esta nos llevara a un acabado conocimiento de lo que es posible desarrollar con las herramientas y capacidades disponibles.

2.4.1 Qué es Telefonía-Computador.

En términos simples es la técnica de coordinar acciones de sistemas de telefonía y computadores. Esta tecnología ha existido en forma comercial desde mediados de los ochentas, pero ha sido explotado solo en algunos nichos de mercado, particularmente en los grandes Call Center[1], donde los volúmenes de llamadas fácilmente justifican el costo de construir complejos sistemas personalizados.

[1] Call Center, centros de atención de llamado.

En los años noventas, muchos factores se han combinado significativamente, simplificando los sistemas de telefonía y computadores, incrementando los lugares de mercados interesados en dicha tecnología.

Los estándares internacionales para la interconexión de telefonía y computadores han sido definidos notablemente por el CSTA (Computer-Supported Telephony Aplication) [Dialogic, 1996], entidad que planea y norma protocolos estándares de ECMA (European Computer Manufacturers Association). En el mercado masivo las APIs (aplicaciones de programación de interfaces) con sus características técnicas han sido promovidas fuertemente por los grandes jugadores del mercado como lo son Microsoft y Novell, y está logrando una rápida aceptación dentro del mercado.

Las tecnologías de procesamiento de la voz han ganado terreno, logrando avances con alta densidad de puertos y bajo costo. Las redes publicas ofrecen mas y mas servicios que habilitan las aplicaciones telefonía-computador, como el ID de línea de llamadas, y aun más importante, el mundo de la economía y los negocios en telefonía a tenido un incremento impresionante, incitando a las organizaciones comerciales para buscar maneras de hacer este proceso más eficaz y barato [Dialogic, 1996].

2.4.2 Convergencia entre Computadores y Telefonía.

Los sistemas de telefonía pública y privada proporcionan las trayectorias en tiempo real de la información entre dos o más conexiones. Tradicionalmente estas trayectorias de información toman la forma de conexiones de voz, originalmente a través del trazado de circuitos análogos vía cable, pero también a través de una gama cada vez más amplia de tecnologías tales como transmisión de radio, codificación de la señal digital, y fibra. En cierto punto, estas trayectorias de transmisión fueron

explotadas por aplicaciones no de voz específicamente como son el fax y transmisión de datos.

Al comienzo, cada aplicación que no fuera de voz requería un conjunto distinto de equipamiento de terminales dedicados a ella, en términos de telefonía para cualquier usuario que tenga conectado un dispositivo a la red telefónica. Las maquinas de fax conversaban solo con otras maquinas de fax, y los dispositivos computacionales enviaban datos solo a otros dispositivos computacionales. Pero en los noventa, estos conjuntos de equipamientos han comenzado a traslaparse, y los propósitos generales de los computadores han comenzado a encontrar un punto de intersección.

Los computadores hoy en día pueden enviar y recibir todo tipo de información que viaje a través de una red telefónica: ellos pueden actuar como maquinas de fax; también pueden interactuar con humanos a través de sintetizadores y reconocimiento de voz; y por supuesto pueden enviar y recibir datos en muchos tipos de formatos. Es en esta intersección, donde los propósitos generales del computador sirven como punto de interfaz, el cual hace que la telefonía y computadores cautivando y potenciando el valor de mercado [Dialogic, 1996].

2.4.3 Control de Llamadas y Medios de Procesamiento.

Esto juega un papel crucial en el rol de la interfaz, lo sistemas de computadora deben interactuar con la red telefónicas en dos vías fundamentales.

- Primero, deben ser capaces de controlar como la llamada es establecida, reconfigurarla y “colgar”, este es un término usado en telefonía para concluir una llamada. A esto se le llamara “función de Control de Llamadas”.
- Segundo, deben ser capaces de enviar y recibir información a través de las interfaces finales, generando y recibiendo la información en los formatos

apropiados ya sea de fax, voz, tonos, o datos. A esto se le llamará “función de medios de procesamiento”.

Una aplicación de Telefonía-Computador usualmente requiere alguna combinación de ambas funciones. Estas funciones, control de llamadas y medios de procesamiento tienen su contraparte en el ser humano cuando este realiza el uso cotidiano del teléfono, por ejemplo:

- Levantar el auricular, presionar los dígitos de marcado, y escuchar por los tonos de señal exitoso cuando se concreta la conexión, la llamada del usuario representa las funciones de control de llamadas.
- Una vez que la comunicación es establecida, hablar y escuchar son la representación humana de las funciones de procesamiento de medios.

Las primeras aplicaciones de teléfono-computador se concentraban en el procesamiento de medios con solo limitadas funciones de control de llamadas. Por ejemplo, el primer sistema de correo de voz respondía llamadas entrantes, con un saludo, y posteriormente grababa la llamada como un mensaje. Como los sistemas consistían inicialmente en funciones de procesamiento de medios, con las funciones de control de llamadas limitadas a detectar el ring, la respuesta de la llamada, y el colgar después de que el mensaje ha sido tomado.

En comparación con los nuevos sistemas de correo de voz y aplicaciones de atención automática actuales han agregado funciones como la transferencia de llamadas. Aplicaciones como esta requieren mayor comprensión del control de llamadas. Debido a la baja en los costos en las tecnologías de procesamiento de señales, estas aplicaciones han agregado avanzadas funciones de procesamiento de medios como la síntesis de la voz, reconocimiento de voz, e interfaces de fax. Las aplicaciones para Call Center requieren incluso funciones mas sofisticadas de funciones de control de llamadas.

Estas aplicaciones implementan características como una bienvenida la contestar una llamada con un extenso rango de opciones de respuesta de voz para posteriormente colocar la llamada en espera una cola, finalmente se coordina la llegada simultanea de llamadas y los datos de ella son asociados a un representante del servicio en seleccionado. Las aplicaciones desarrolladas para Call Center utilizan típicamente las mas avanzadas funciones de control de llamadas y procesamiento de medios, incluyendo funciones de control de llamadas especiales para monitorear las llamadas mientras pasan a través de las colas en su camino hacia su destino final, y funciones procesamiento de medios que comprenden cuales usuarios pueden completar sus negocios sin haber realizado ninguna interacción con alguna persona [Dialogic, 1996].

2.4.4 Hardware Modular de Procesamiento de Medios.

El hardware de procesamiento de medios es relativamente simple, se considera que cada línea telefónica tiene un conjunto de recursos de hardware dedicado. Por ejemplo, una tabla típica de procesamiento de voz puede soportar cuatro líneas telefónicas análogas, con digitalización de la voz y circuitos fijos en cada canal.

El hardware de procesamiento de medios se vuelve considerablemente más complejo, sin embargo, cuando las aplicaciones necesitan estar disponibles para reconfigurar recursos on-the-fly (en funcionamiento). Los grandes sistemas también necesitan extensibilidad en incrementos modulares acomodados al crecimiento de la aplicación en cuestión; por ejemplo, una aplicación de media-escala puede requerir un pool de dos interfaces de circuitos T1 (provee un total de 48 canales de voz), 48 digitalizadoras de voz y unidades de lectura, ocho reconocedores de diálogos, ocho canales de procesamiento de fax, y veinticuatro interfaces análogas. Estos recursos deben ser configurables on-the-fly, esto significa que una llamada entrante que toma un canal T1 debe ser asignable a los digitadores, unidades de lectura, reconocedores, procesadores de fax e interfaces análogas en cualquier combinación.

Como la configuración no puede entrar en una única tabla de circuitos (y si se pudiera no seria fácil expandir la misma), por esto han sido propuestas varias arquitecturas en las cuales tales sistemas pueden ser montados. Las dos propuestas líderes, se conocen como MVIP y SCbus.

MVIP y SCbus especifican tipos de buses mediante los cuales interconectar tarjetas de telefonía, y entregan un mecanismo separado de comunicación para coordinar los subsistemas. El esfuerzo de MVIP es administrado por la organización GO-MVIP; el SCbus fue desarrollado por el grupo de trabajo de SCSA, incluido recientemente dentro del foro de telefonía de computadores de la empresa Dialogic, la cual promueve el estándar ECTF [Dialogic, 1996].

2.4.4.1 CTbus.

Es un tipo de bus de alta velocidad que permite le interoperabilidad de sistemas de buses mezclados, entre los productos CTbus y otros productos CT. Este bus tiene características especiales tales como:

- Cumple con las especificaciones del (ECTF).
- Termina con la confusión sobre interoperabilidad de buses.
- Los desarrolladores pueden soportar uno de los modos de compatibilidad.
- Los desarrolladores de sistemas pueden construir sistemas mixtos.
- Alta flexibilidad.

Existen dos tipos de buses CT, los que se describen en la tabla 2.4.4.1.1.

Tabla 2.4.4.1.1 Comparativa buses CT, H.100 vs H.110.

H.100	**H.110**
Provee entre 2560 (16 streams SCbus y 16 streams CTbus) y 4096 (Caso de CTbus solo) timeslots en 32 streams.	Provee entre 3072 y 4096 timeslots en 32 streams
Las placas que cumplen con el H.100 tienen solamente el factor de forma PCI.	PCI compacta, factor en forma de eurocard
Usa un nuevo cable plano.	Usa backplane (sin cable plano) para enlutar datos TDM entre placas

2.4.4.2 SCbus.

El SCbus es un bus de alta velocidad en tiempo real [Dialogic, 2000], es un estándar abierto del SCSA; es un bus ancho y tiene una capacidad de 16 TDM (Time Division Multiplexed). Si analizamos las características generales del SCbus podemos notar lo siguiente: es un cable plano con conectores, se utiliza para compartir recursos de hardware en telefonía-computador, interconecta componentes de procesamiento de voz, conecta cualquier dispositivo SCbus con otro, permite compartir recursos caros y de expansión, por ello es capaz de que los diferentes recursos se conecten entre si, y se compartan con todo el sistema, además provee sistemas de gran densidad y mayor flexibilidad. La arquitectura del SCbus tiene características especiales que se deben tomar en cuenta:

- El cable plano cuenta con 1024 timeslots y 2048 sobre el Backplane VME.

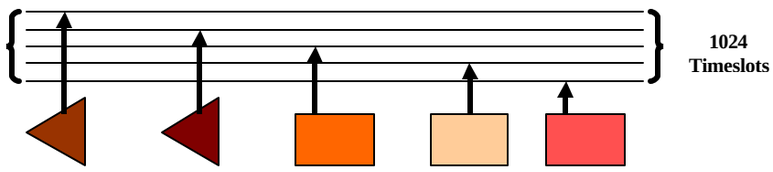


Fig. 2.4.4.2.1 Esquema de conexión SCbus.

- Utiliza un esquema de codificación Mu-law o A-law.
- No necesita el terminador.
- Todos los dispositivos son iguales entre si.
- Cada dispositivo transmite sobre un único timeslot.
- Los dispositivos siempre transmiten sobre un timeslot impidiendo duplicación.
- Los timeslots de transmisión están “clavados”.
- Los dispositivos pueden escuchar cualquier timeslot, la mayoría de ellos pueden escuchar solo uno a la vez (excepto las MSI, DCB).

- Múltiples dispositivos pueden escuchar al mismo timeslot simultáneamente (caso de música en espera, ver Fig. 2.4.4.2).

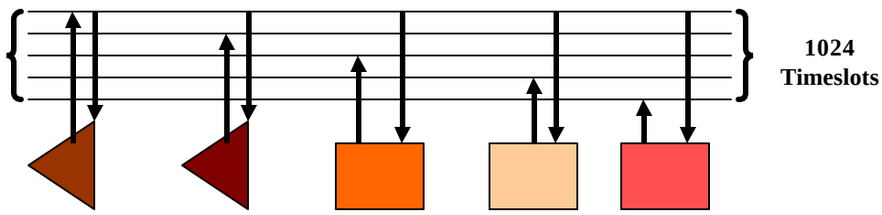


Fig. 2.4.4.2 Caso de música en espera.

En un escenario real ¿Cuántas llamadas simultáneas pueden soportar 1024 timeslots del SCbus?, si en el escenario se cuenta solo con recursos de voz e interfaces de red, el SCbus soporta 512 llamadas. Cada llamada utiliza 2 timeslots (uno para la interfaz de red y uno para el recurso de voz).

Dadas las características antes mencionadas sobre el SCbus queda claro que: tiene un ancho de banda muy alto (en sistemas PC y VME), permite una conmutación simple sobre el bus (puentea dos llamadas cualquiera mediante la conexión de dos interfaces de red analógicas o digitales), comparte recursos en forma eficiente a través de múltiples placas, en el futuro el canal de mensajes soportará comunicación en tiempo real entre placas.

2.4.5 Señalización. Control de conexión de llamadas.

La red telefónica es un sistema ampliamente distribuido de nodos inteligentes conmutados. Para estos nodos la cooperación exitosa para el establecimiento y corte de las llamadas, estas deben comunicarse con cada uno y con el equipamiento terminal de usuario. Este proceso es llamado “señalización”. Una señalización de conexión exacta y confiable entre el teléfono y los sistemas informáticos es esencial para los usos acertados de teléfono-computador, desde que la señalización es el

medio de control y constituye el único medio de comunicación entre sistemas inteligentes en dos dominios distintos.

La señalización puede tomar lugar dentro de la banda, esto es, a través de la ruta de habla sobre el canal telefónico, o fuera de banda, esto es, a través de algún canal de comunicación de la ruta de habla. Hoy en día las redes telefónicas, y los equipos y terminales de señalización generalmente están dentro de la banda (excepto por los dispositivos ISDN[2]), mientras la señalización entre la telefonía conmutada es obtenida fuera de la banda por razones de eficiencia y seguridad.

El equipamiento original de terminales de señalización fue por supuesto, la voz humana, como subscriptor que habla al operador. El primer equipo automático de terminales de señalización con tiempos que realizan quiebres a través de pulsos en una línea telefónica análoga y genera tonos conmutados especiales de alerta al subscriptor como el ring y ocupado. En muchos sistemas telefónicos, la señalización por tonos es ahora usada para equipos terminales de señalización fuera de banda en ambas direcciones. El esquema mas conocido para equipos terminales de señalización de redes es el DTMF[3], bajo el cual genera pares simultáneos de tonos que representan cada digito marcado. Desafortunadamente la señalización desde la red telefónica regresa al equipo terminal y esto ha sido estandarizado, una situación totalmente familiar es que un subscriptor realice una llamada internacional. La señalización por tonos que retornan desde el final de una llamada internacional pueden no asemejarse a la señalización por tonos local, y el subscriptor puede no ser capaz de reconocer la diferencia entre ocupado, o marcando de otro país.

Es necesario decir que, es un reto significativo el diseñar equipos terminales teléfono-computador que sean capaces de interpretar la amplia variedad de tonos y

---

[2] ISDN, Integrated Services Digital Network, se define como una red conmutada de canales digitales que proporciona una serie de servicios integrados, siguiendo las recomendaciones serie I del CCIT.

[3] Dual Tone Multi Frecuency, corresponde a los tonos generados por un aparato telefónico estándar al presionar alguna de sus teclas, la frecuencia de cada tono indica a que digito corresponde (0…9, *, #).

otras señales generadas fuera del la banda por muchos elementos de la red mundial de telefonía. De hecho, la realización de señalización exacta y segura entre interfaces telefónicas basadas en computadores y equipos telefónicos tradicionales es una de las dificultades más grandes en la construcción de aplicaciones teléfono-computador.

Esta dificultad puede ser aliviada de alguna manera cambiando de lugar los esquemas de señalización fuera de banda, que confían generalmente en la mensajería digital no ambigua. Por ejemplo en, la señalización digital orientada a mensajería en un dispositivo terminal ISDN básico es mucho más confiable que una señalización análoga en la banda. Enlaces de integración de telefonía y computadores (CTI), son ofrecidos actualmente por la mayoría de las modernas PBXs, entregando mecanismos de señalización, a través del cual un sistema computacional puede recibir una señalización consolidada de grupos de extensiones telefónicas.

Múltiples métodos de señalización se encuentran disponibles en un único sistema telefónico. Una PBX puede soportar simultáneamente un enlace CTI, circuitos ISDN, y un conjunto propietario de señalización digital. Cualquiera de estas puede proveer mayor exactitud de la información de señalización para aplicaciones de telefonía-computador y están disponibles a través del dispositivo terminal de señalización análoga dentro de la banda en los mismos conmutadores [Dialogic, 1996].

2.4.5.1 Protocolo MFCR2.

La señalización MFCR2 es un sistema de señalización internacional que transmite información numérica y otra relacionada con la llamada y las líneas de llamada del abonado. Las características fundamentales del sistema de señalización R2 son las siguientes [C.C.I.T.T, 1969]:

- Explotación automática y semiautomática.
- Elevada confiabilidad en la transmisión de la información necesaria para el abastecimiento de una comunicación.

- Gran rapidez de establecimiento de la comunicación, es decir, reducción del periodo de espera entre el momento en que el abonado que llama ha terminado de marcar y el momento en que percibe la señal de llamada.
- Explotación unidireccional y bidireccional.
- Posibilidad de aprovechar las ventajas de los sistemas modernos de conmutación mediante la previsión del suficiente numero de señales en los dos sentidos, lo que permite transmitir las informaciones numéricas y otras relativas a la línea del abonado solicitante y del abonado.
- Posibilidad de adoptar un método de señalización entre registradores de extremo a extremo, lo que permite simplificar los registradores en los centros de transito y reducir su tiempo de ocupación.

2.4.5.2 Protocolo ISDN.

ISDN o RDSI (Integrated Services Digital Network) es una red digital pública de voz y datos integrados que puede enviar información 50 veces más rápido que un módem a 2.400 bits por segundo. Lo hace de forma digital, creando dos canales de transmisión de 64 kilobits por segundo (Kbps) que pueden transmitir voz o datos, y un canal de transmisión de 16 Kpbs que transmite paquetes de datos de información o de control.

La interfaz básica ISDN/RDSI es el estándar para conectar a los subscriptores a ISDN/RDSI. La interfaz utiliza el par de cables telefónicos existentes para transmitir sólo información digital. Tres canales o líneas digitales existen en este cable. Los canales son multiplexados dándole a cada uno una fracción de tiempo en el cable.

Los tipos circuito o canales ISDN/RDSI son:

Tabla 2.4.5.2.1. Topología de canales ISDN/RSDI.

Canal ISDN	Descripción
A	Canal analógico telefónico de 4Khz.
B	Canal digital PCM de 64Kbps para voz y datos.
C	Canal digital de 8 o 16Kbps.
D	Canal digital de 16kbps para señalización fuera de banda.
E	Canal digital de 64Kbps para señalización ISDN interna.
H	Canal digital de 384, 1536 o 1920kbps.

Una de las fortalezas de ISDN es que especifica pocas interfaces físicas para poder conectar a la misma red una gran variedad de equipos (datos, voz, video, etc.). Las características principales de ISDN:

- Conexiones digitales punto a punto. (No más líneas analógicas).
- Señalización por canal común, esta es una de las propuestas más fuertes de ISDN, aquí se propone enviar la señalización de circuiteria por un canal diferente al que se utiliza para enviar la señal de la voz.
- Interfaz de multipropósito al usuario, para transmitir diferentes servicios como voz, datos y video por un mismo estándar de conexión.

Las combinaciones mas usadas actualmente son:

- Básica BRI: 2B+1D

Fig. 2.4.5.2.1. Esquema de combinación ISDN BRI. [Etsi,1990]

- Primaria PRI: 23B+1D (T1:1536 Mbps, usado en USA, Japon)

  30B+1D (CCITT: 2046 Mbps, usado en Europa, Chile.)

Fig. 2.4.5.2.1. Esquema de combinación ISDN PRI. [Etsi,1990]

Un servicio integrado puede proporcionar un amplio surtido de información en un sencillo y bien organizado paquete. Esta información puede estar en forma de voz, datos o vídeo. Inicialmente, los servicios disponibles en ISDN/RDSI no estarán integrados. Voz y datos, a pesar de poder ser accedidos juntos en una terminal integrada, tienen poco que ver unos con otros. Las llamadas de voz sólo contendrán voz, y las de datos sólo contendrán datos.

2.4.5.3 Protocolo SS7.

Para entender el protocolo de señalización 7 (SS7), es útil entender el concepto de capas de protocolo. En 1983 algunas de las mayores compañías de telecomunicaciones comenzaron a presentar numerosos problemas cuando realizaban desarrollos, por causa de los computadores de muchos tipos, cuando intentaban comunicarse con otro a través de las redes de conexión. Entonces se decidió crear alguna interfaz específica que pudiera ser usada por todos.

En el proceso, se dieron cuenta que al crear una interfaz especifica se volvería lento el desarrollo e implementación de futuros estándares y futuras tecnologías computacionales. Por esto, se insto a crear interfaces especificas, para esto se decidió crear un modelo de arquitectura de capas, el cual pudiera ser usado en el desarrollo de futuras redes. Como resultado se creo la interconexión de sistemas abiertos (OSI)

modelo adoptado posteriormente por la organización internacional de estándares (ISO).

Las capas de protocolo consisten en programas modulares, cada uno diseñado para desempeñar cierto grupo de funciones, y cada diseño debe ser capaz de ofrecer funcionalidad a otros módulos. Por ello cada modulo se convierte en una “parte” de la arquitectura de protocolo entera, y cada parte que ofrece un servicio funcional a otra parte se le llama “parte usuario”.

Es difícil encontrar un protocolo que incorpore las siete capas de funcionalidades agrupadas en exactamente como se presentan en el modelo OSI. La SS7 no es la excepción. Sin embargo para quienes tienen familiaridad con el modelo OSI, puede ser útil verlo en paralelo con un protocolo no familiar a modo de referencia.

La comparación de OSI con SS7 se muestra con este propósito en la figura 2.4.5.3.1; cabe notar que mientras mas se desciende en capas se incorporan funcionalidades de OSI directamente, algunas de las funcionalidades de capas superiores se mezclan con las inferiores.

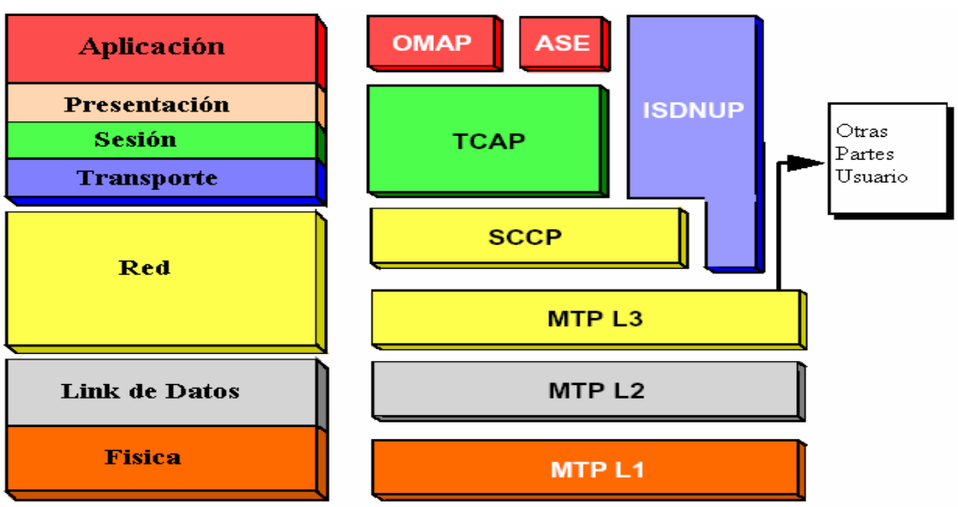


Fig. 2.4.5.3.1 Modelo de Capas OSI (izquierda), Modelo de Capas SS7 (derecha).

A continuación se explican en la tabla 2.4.5.3.1 los conceptos generales del protocolo por capas.

Tabla 2.4.5.3.1 Especificación Modelo de capas SS7.

ISDNUP	(Integrated Services Digital Network Users Part)
MTP L1	(Message Transfer Part) nivel 1 representa la capa física. Esto quiere decir que dicha capa trata con la conexión al computador el cual utiliza funcionalidades programadas dentro de una red, a través de la cual desea comunicarse. El MTP nivel 1 considera los enlaces de control de reloj, y todas las consideraciones físicas de envío de mensajes sobre alambres, esta capa es netamente Hardware.
MTP L2	La capa MTP 2 desempeña el último manejo inteligente de mensajes antes de ser enviado fuera de los enlaces. Asimismo realiza la primera dirección inteligente de los mensajes recibidos de la red. Por ello el nivel 2 esta ciertamente asociado con los enlaces, una de las principales tareas asignadas es la de simplemente monitorear y reportar, no controlar.
MTP L3	La funcionalidad de MTP nivel 3 se divide en dos grupos. Uno de estos grupos trata de cuando MTP envía mensajes y los recepciona, esto se refiere a SMH (Signalling Message Handling). El segundo grupo es cuando MTP nivel 3 maneja el trafico, los enlaces y las rutas disponibles para ello. Este grupo esta referido a SNM (Signalling Network Management).
OMAP	(Operations, Maintenance and Administration Part) Esta es una parte importante debido a que revela en la capa de aplicación la necesidad de administración y manejo operacional del protocolo.
ASE	(Application Service Element) Este es un servicio asociado a la capa de aplicación.
TCAP	(Transaction Capabilities Application Part), desde el mismo nombre se puede notar un gran cambio. Esta ya no es una referencia a la parte usuario, esta ahora esta parte es aplicación, este es el servicio primario ofrecido de aplicación el cual es el paquete de datos.
SCCP	(Signalling Connection Control Part) entrega la dirección del subsistema numérico necesario, manejando las entregas en secuencia cuando es necesario (gracias a la capa 3 MTP), entregando servicios de segmentación cuando los mensajes obtenidos son muy extensos, se manejan consideraciones de tipo global.

3 TECNOLOGIA CTI.

3.1 Definición de CTI.

Las siglas CTI son un acrónimo anglosajón que significa “Computer Telephony Integration” (Integración de Sistemas de Telefonía y Computador).

La integración de telefonía y computadores, tecnología conocida como CTI, ha permitido el desarrollo de un gran número de nuevos y avanzados servicios, los cuales han generado un gran interés en el panorama económico y empresarial, debido al valor agregado que la combinación de los beneficios provistos por los sistemas computacionales y los sistemas de telecomunicaciones entrega.

Los componentes principales de la tecnología CTI son:

- PBXs.
- Modems.
- Tarjetas de Telefonía.
- Computadores.

Dichos dispositivos cuentan con características particulares las que se detallaran a continuación.

3.1.1 PBX.

PBX (Private Branch Exchange, Central Privada de Conmutación), suelen ser el dispositivo principal en los sistemas CTI. Las PBX permiten todo tipo de conexiones al exterior: líneas analógicas, accesos básicos y primarios RDSI, líneas GSM, conexiones empleando VoIP (Voz sobre IP) [Martin, 2002].

Para que una PBX adquiera funcionalidad CTI, es necesario contar con el hardware y firmware necesario en la central, provisto por el fabricante, también es necesario contar con un conector a un computador que permita controlar algunas funcionalidades de la central (para entregar servicios CTI), este conector es conocido como Link CTI [Edgard, 1997].

Fig.3.1.1.1 PBX Nortel

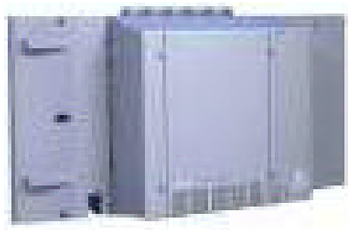


Fig.3.1.1.2 PBX AVAYA.

3.1.2 Modem.

Estos dispositivos pueden proveer servicios CTI básicos, los que van desde el envió de fax desde el PC, hasta pequeños sistemas de respuesta de voz interactiva. Con la adecuada programación pueden ofrecer IVRs de baja densidad de tráfico, con características similares a sistemas que emplean tarjetas de telefonías propietarias, lo cual presenta una potencialidad para desarrollo a bajo costo y de estudio en centros de investigación pequeños.

Fig.3.1.2.1 Modem Interno asus 56 kbps.

Fig.3.1.2.2 Modems Externo USR robotics 56k.

3.1.3 Tarjetas de Telefonía.

Las tarjetas de telefonía son hardware propietario, construido para aplicaciones CTI, estas se conectan a un computador generalmente en forma directa sobre el bus ISA o PCI, entre las más empleadas se destacan las tarjetas Dialogic, y Aculab. Las tarjetas de telefonía pueden ser programadas en diferentes lenguajes y entornos de desarrollo, además proveen una gran gama de alternativas de interconexión y recursos para el desarrollo de aplicaciones CTI. Existen tarjetas analógicas, tarjetas digitales (soportan una gran variedad de protocolos como ISDN, MCF R2, SS7, etc.), las cuales incorporan una amplia variedad de recursos como [Vidal, 2000]:

- recursos de señalización,
- recursos de media (reproducción de mensajes).
- recursos de conferencia.
- recursos de línea (permiten conectar directamente un equipo telefónico al computador).
- soporte para comunicaciones de VoIP (Voz sobre IP).
- recursos de fax.

La mayoría de las tarjetas de telefonía soportan todas las funcionalidades de voz de un modem (DTMF, detección de silencio, grabación de llamadas de voz, identificación de llamada y mas), y la todas o la mayoría de las funciones de telefonía especificas, incluyendo reconocimiento de tonos, extensiones de líneas digitales, desconexión remota y mas.

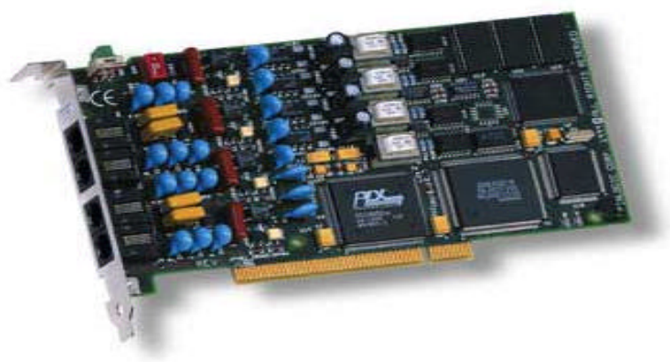


Fig. 3.1.3.1 Dialogic D4/Pci [Ivrsoft, 2002].

Fig. 3.1.3.2 QX 2000 [Nmscommunications, 2002].

3.2 MODELOS DE IMPLEMENTACION CTI.

3.2.1 Modelo “Third Party CTI”[4].

A este modelo se le suele llamar en castellano “acceso en tercera persona”, también podría ser llamarlo “modelo cliente servidor”. Como todos estos nombres indican, el objetivo es que un solo computador (el servidor de telefonía) se comunica con los dispositivos telefónicos. Cuando un computador cualquiera quiere hacer uso de las facilidades CTI tiene que pedírselo al servidor a través de la red (por tanto, este modelo necesita que los computadores estén conectados en red). Aunque los primeros sistemas de este tipo funcionaban sobre protocolo IPX (red Novell), actualmente se ha evolucionado hacia el protocolo IP. Concretamente son servicios de red que utilizan el transporte simplificado UDP (User Datagram Protocol) por lo que no son fiables fuera de las redes locales o, como mucho, metropolitanas [Tanembaum, 1991]. El ejemplo más importante de acceso “Third Party” es el estándar TSAPI [Novell, 1998], (Telephony Services Application Programs Interface) que fue desarrollado conjuntamente por Novell y ATT.

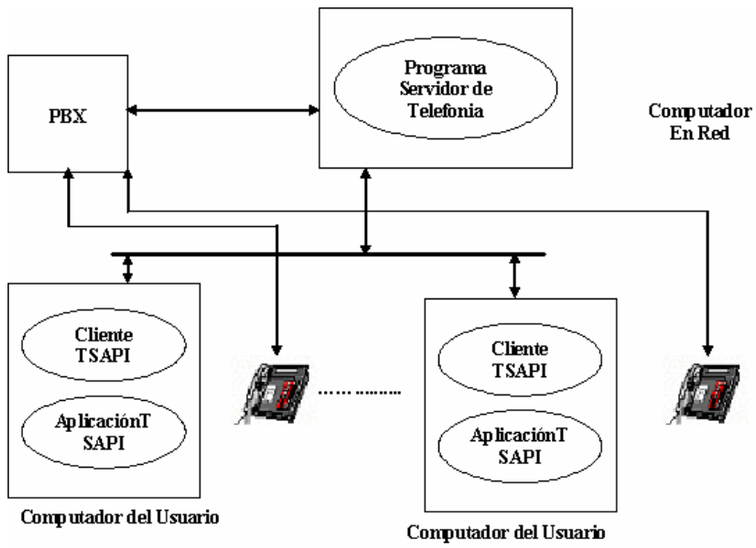


Figura 3.2.1.1. Recursos Telefónicos Compartidos (Third Party CTI) [Martin, 2002].

[4] Third Party o “acceso en tercera persona”, en la mayoría de la documentación el término se emplea en inglés.

3.2.2 Modelo “First Party CTI”[5].

En castellano, “acceso en primera persona”. Se trata de que cada computador que realizar funciones CTI esté conectado a uno o varios dispositivos telefónicos. En un modelo “First Party” puro, un computador sólo puede acceder a los dispositivos que están conectados a él (en “Third Party” cualquier computador puede acceder a cualquier dispositivo, sujeto a la política de permisos que se establezca en el servidor). Actualmente, existen modelos híbridos, esto es: sistemas “First Party” que tienen una “extensión remota”. La “extensión remota” es un módulo cliente-servidor que permite que un computador acceda a los dispositivos conectados a otro, como si fuera él el que los tiene conectados físicamente. Sin embargo, esta es una solución bastante pobre: da problemas a la hora de que dos computadores accedan a la vez a un mismo dispositivo y, desde un punto de vista de diseño, la “extensión remota” no es más que un remiendo sobre el sistema inicial. El ejemplo más importante de acceso “First Party” es el estándar TAPI [Microsoft, 1996] (Telephony Application Programs Interface) que fue desarrollado por Microsoft, las versiones iniciales eran “First Party” puro, a partir de la versión 2.1 se incluyó el RTAPI (Remote TAPI).

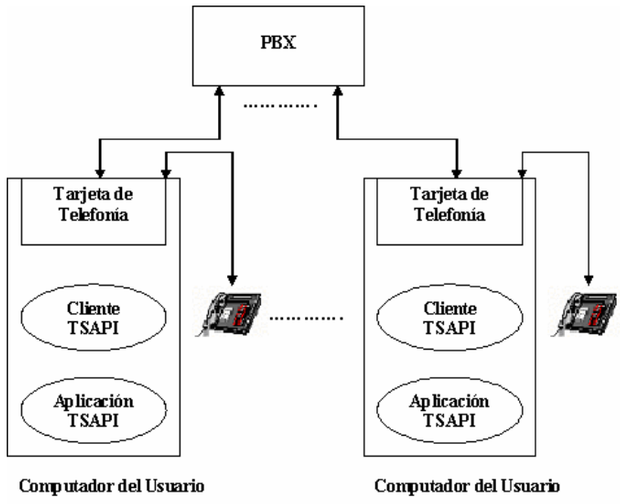


Figura 3.2.2.1. Recursos Telefónicos Dedicados (First Party CTI) [Martin, 2002].

[5] First Party o “acceso en primera persona”, en la mayoría de la documentación el término se emplea en inglés.

3.3 Interconexiones CTI.

Las interconexiones en las arquitecturas CTI se pueden clasificar en [Vidal, 2002]:

Interconexiones de telefonía:

- Digitales, empleando señalización: ISDN PRI, ISDN BRI, MCF R2, SS7, etc.
- Analógicas.
- O provistas mediante VoIP.

Interconexiones de datos: permiten el intercambio de información con sistemas como bases de datos, envió o recepción de mensajes sobre la red de datos., etc.

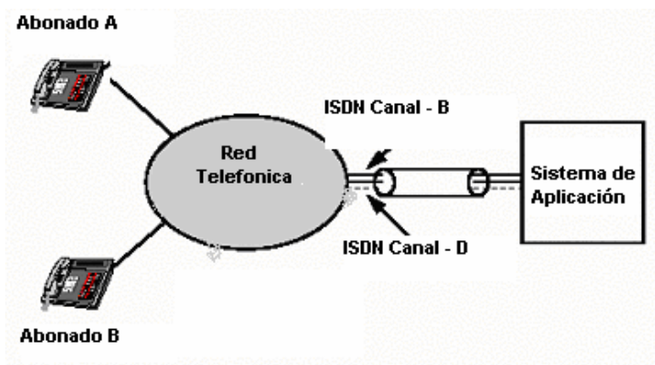


Figura 3.3.1 Aplicación CTI Utilizando Señalización ISDN [Elliot, 2001]

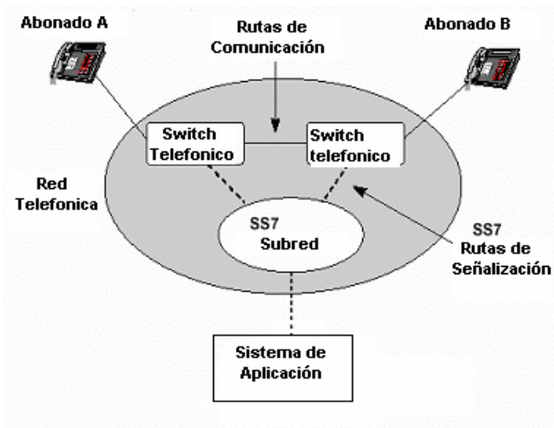


Figura 3.3.2 Aplicación CTI Utilizando Señalización 7 [Elliot, 2001]

3.4 Procesamiento de la Voz.

Para desarrollar cualquier sistema basado en voz, se debe comenzar analizando la señal de voz humana. La siguiente figura muestra un modelo de análisis de procesamiento de voz.

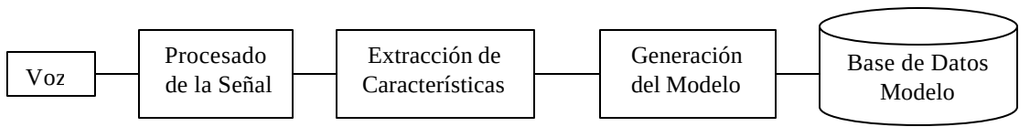


Figura 3.4.1 Fase de análisis y modelamiento de la voz.

Lo que se utiliza para realizar el procesamiento de voz, es el análisis de los sonidos, este análisis se basa en:

- Análisis de sonidos según el mecanismo de producción de estos, en el tracto humano. Proceso seguido para transformar la onda acústica digitalizada en los vectores de características acústicas.
- Extracción de la información más interesante para la posterior generación de modelos. El objetivo será encontrar el significado de una señal de voz como se muestra en la figura 3.4.2

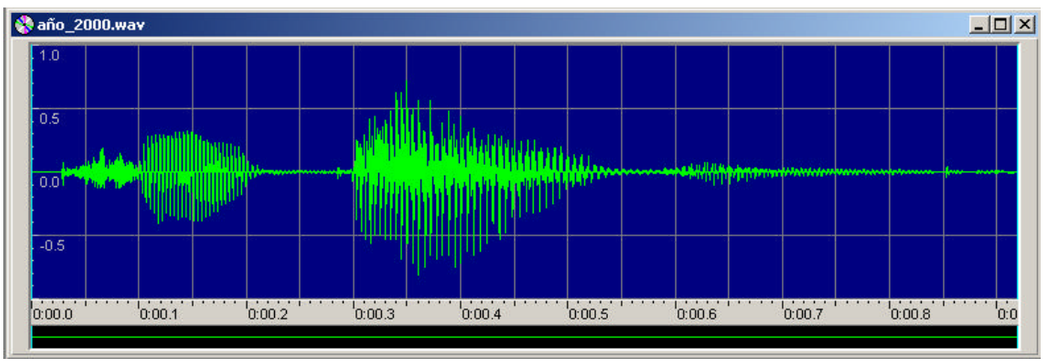


Fig. 3.4.2 Onda de voz representada por una señal digitalizada. Palabra: "*año dos mil*".

La figura anterior es b que se denomina una onda acústica digitalizada, es decir, una representación del sonido en un intervalo de tiempo.

Para realizar un análisis, se debe asumir que la señal de voz es estacionaria en periodos muy cortos de tiempo (se extraen valores característicos analizando la señal en tramas de 20 a 32 milisegundos de longitud).

Antes de realizar cualquier método de análisis se debe considerar la problemática asociada al campo del habla. Dicha problemática esta englobada en cuatro problemas fundamentales:

a) Variabilidad: la voz humana presenta un factor importante de variabilidad, debido principalmente a tres causas que son: variabilidad Intra-locutor, variabilidad Inter-locutor y canal de transmisión y entorno. Dichos factores hacen que una misma palabra nunca se perciba con las mismas características acústicas. La primera causa, variabilidad intra-locutor, se debe a variaciones no lineales en las duraciones de los sonidos elementales. Dichas variaciones se producen principalmente en la modificación del ritmo de pronunciación. Además, pueden darse cambios en la frecuencia e intensidad, debido fundamentalmente a cambios en el estado físico/psíquico del locutor o al modo de pronunciación (susurrar, tararear, gritar, etc.).

La segunda fuente de variabilidad es el inter-locutor, y se debe a variaciones físicas en el tracto vocal de los locutores, así como el sexo, edad y localización geográfica (pronunciación y acento típico de cada región). Los cambios que afectan a la señal cambiaran en la escala de frecuencias. El último motivo de variabilidades debe al canal de transmisión de la señal y al entorno más o menos ruidoso que rodee la transmisión.

Los elementos que pueden causar distorsión son los micrófonos utilizados, el ancho de banda de la línea de transmisión (si el reconocimiento se realiza a través de la línea telefonía), ruidos de fondo, interferencias. Dadas todas las posibles fuentes de variabilidad, se debe realizar una extracción minuciosa de los elementos esenciales de la voz con independencia del contexto y el entorno.

b) Continuidad: cuando hablamos, normalmente pronunciamos una frase en la que los silencios son casi imperceptible. Esto supone un problema importante a la hora del reconocimiento, ya que el único modo de analizar los diferentes elementos acústicos es poder separarlos. De igual modo los sonidos elementales, como los fonemas, aparecen concatenados sin ningún tipo de separación y se modifican por el contexto inmediato (fonemas adyacentes). Este factor se reconoce con el nombre de "articulación" y se debe al hecho de que la pronunciación de un fonema requiere que el tracto vocal humano adquiera la configuración adecuada, ya que dependerá tanto del fonema pronunciado como de los fonemas adyacentes a cada situación, los cuales variarán las características del primero.

c) Redundancia: se puede demostrar fácilmente que la señal de voz es redundante, ya que el mensaje implícito en una comunicación normal puede transmitirse a una velocidad de 50 bits/seg., la transmisión adecuada de las señales de voz requiere del orden de los 100 bits/seg., por ejemplo para 8Khrz (extracción de 800 muestras por segundo) y 12 bits de muestra. La redundancia se debe a que la señal de voz transporta información adicional que identifica al locutor, su estado de ánimo y físico, acento, entonación, etc. Esta redundancia hace que la elección de parámetros de extracción de información este estrechamente ligada al tipo de información que se quiere extraer de la señal de voz.

d) Multi-interactividad: existen diferentes niveles en el proceso de percepción y comprensión del habla fuertemente relacionados, y que aporta información útil al reconocimiento del habla. En la tabla 3.4.1, se describen los cuatro niveles principales envueltos en la multi-interactividad.

Tabla 3.4.1. Niveles de Multi-interactividad.

Análisis Acústico	Corresponde al análisis de las características acústicas de la señal de la voz.
Análisis Fonético	Este nivel extrae los elementos acústicos fundamentales como fonemas, silabas, etc.
Análisis Morfológico	Identifica los elementos constructivos aplicándoles reglas gramaticales para la detección de construcciones correctas del lenguaje.
Análisis Semántico	Se encarga de la comprensión del lenguaje transmitido, eliminando interpretaciones incorrectas sobre la base del conocimiento de la realidad y del contexto del mensaje recibido.

Todos los niveles están estrechamente relacionados, aportando cada uno de ellos información fundamental en el proceso del reconocimiento. Desgraciadamente, a pesar de que se conoce la relación entre todos, no existe aun ningún mecanismo capaz de integrar toda esta información. Es por ello que el proceso de reconocimiento del lenguaje natural resulta tan complejo.

3.4.1 Naturaleza de la señal de voz

Las señales sonoras se caracterizan por tener un contenido frecuencial en el rango de los 300Hz a 4000Hz y una energía muy elevada. Estas señales se generan por intervención de las cuerdas vocales, y además presentan cierta periocidad.

Las señales no sonoras, también denominadas fricativas, se caracterizan por tener un contenido muy bajo de energía y una componente frecuencial uniforme. Presentan aleatoriedad en forma de ruido blanco (señal con valores aleatorios impredecibles).

De este modo, la señal dividida en dos partes, se modela como dos fuentes d e excitación diferentes que serian las que conformasen la arquitectura del tracto vocal. Se puede observar como seria fácil, en apariencia, modelar el tracto vocal como un filtro variable en el tiempo, cuyos parámetros varían en función de la acción consciente que se realiza cuando se pronuncia una palabra.

El filtro variable en el tiempo tiene dos posibles señales de entrada para simular señales sonoras y no sonoras. Para señales sonoras la excitación se simula con un tren de impulsos de frecuencia controlada, mientras que para las señales no sonoras la excitación será un ruido aleatorio (ruido blanco), esto se ejemplifica en la Fig.3.4.1.1

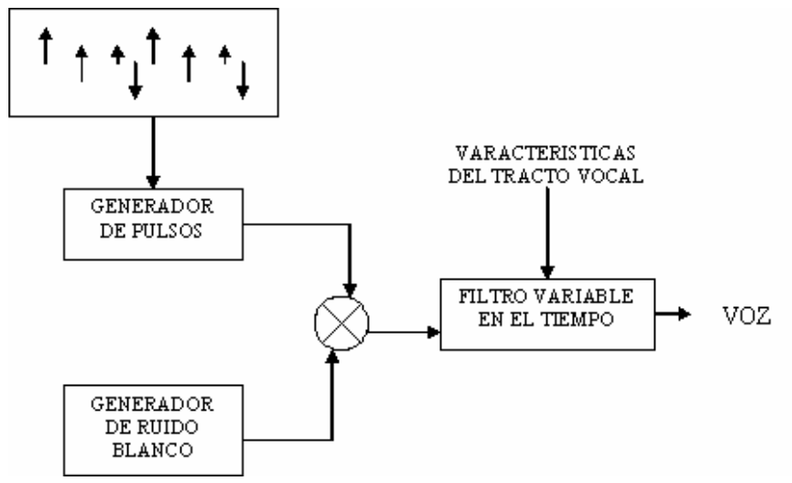


Fig. 3.4.1.1 Modelo de generación de sonidos en el tracto vocal (aproximado).

3.4.2 Reconocimiento Avanzado De Voz (ASR).

Durante los últimos diez años se han venido utilizando los sistemas de Respuesta de Voz Interactiva (IVR Interactive Voice Response) por los tonos del teléfono, para dar acceso a bases de datos, conectando a las personas con la información que necesitan desde cualquier lugar y en cualquier momento [Nuance, 2002]. Esto constituyó un primer paso en la integración de la telefonía y la

computadora. A pesar de permitir un acceso rápido y eficiente a la información, y de ofrecer una variada gama de transacciones, los sistemas IVR se veían limitados por la utilización del teclado del teléfono como herramienta de base, y por la dificultad para estructurar contenidos amplios e informaciones complejas de forma eficiente.

Los avances tecnológicos en algoritmos lingüísticos, procesamiento del lenguaje natural, gestión del vocabulario y reproducción del lenguaje, han hecho posible servirse de una herramienta mucho más fácil y flexible: la voz humana.

Las prestaciones de las actuales soluciones de Reconocimiento Avanzado del Lenguaje (ASR) van desde el reconocimiento de letras y números pronunciados, hasta estructuras de oraciones complejas.

Un gran logro han sido las actuales aplicaciones de Reconocimiento del Lenguaje Natural, que permiten la comprensión de oraciones complejas pronunciadas a la velocidad normal de comunicación. Esta tecnología permite conseguir transacciones más sencillas y rápidas sin necesidad de ayuda, y se aumenta la satisfacción del usuario.

Los sistemas de reconocimiento de voz desarrollan todo su potencial, combinándose con la utilización de bases de datos cada vez más avanzadas, diseñadas para aplicaciones de Internet y a la hora de proporcionar la información que el usuario desea, se emplea la técnica de Conversión Texto a Voz (TTS).

3.4.3 Tecnología TTS.

Usando conocimiento acústico, fonético, y lingüístico, un texto ordinario puede convertirse a voz. Este proceso es popularmente llamado conversión texto a voz (Text to Speech). La tarea de estos sistemas es convertir un texto entero, en un discurso que represente justamente el sentido que conlleva el texto. Existen actualmente

diversas técnicas para la generación de mensajes orales por computador, se intentará señalar las características esenciales de los sistemas de conversión de texto a voz.

Entre los varios esquemas de texto-a-voz que se han probado, hay algunos que concatenan palabras pregrabadas. Sistemas de conversión texto a voz que generan voz humana sintética por medio de concatenación de sub palabras segmentadas tales como sílabas, morfemas, fonemas, difonemas o trifonemas generalmente aceptan un texto no restringido, pero, si bien ellos podrían tener un ritmo sonoro y entonación natural, su calidad de interpretación varia ampliamente. El resultado puede ser altamente inteligible, especialmente después de ganar experiencia escuchando, pero ningún sistema text-to-speech, ha sido capaz de producir resultados comparables con sistemas de almacenamiento de palabras de moderada calidad.

La síntesis de voz el proceso de generar voz a partir de texto, es una emulación del proceso del habla producido por el humano a través de las cuerdas vocales y que tiene gran importancia porque nos apoya en el desempeño de un sin fin de tareas.

La síntesis de voz es realizada a través de los TTS, que son sistemas que transforman texto en sonidos que podemos reconocer como voz. La síntesis de voz es un proceso que se facilita más en algunos lenguajes porque existen reglas bien estructuradas o porque no existen tantas variaciones en sus fonemas (alófonos).

Existen diferentes tipos de enfoques para la producción de voz, a través de los sistemas de texto a voz, entre ellos se encuentran los que manejan la síntesis paramétrica, la síntesis articulatoria y la síntesis Concatenativa (ver anexo I).

3.5 Estándares para interacciones entre sistemas telefónicos y computadores.

3.5.1 CSTA.

CSTA es un acrónimo de Computer Supported Telecommunication Applications, y fue desarrollado por la ECMA, CSTA se suele asociar a TSAPI (de hecho, la dll cliente de TSAPI se suele llamar **csta32.dll**), sin embargo, dicha creencia es un error.

La verdad es que CSTA describe un protocolo de comunicación entre un ordenador y un dispositivo telefónico. Dicha descripción se divide en dos documentos:

- La descripción de los servicios que el sistema telefónico, ofrece al ordenador (Posibles acciones a realizar) y también de los eventos que se enviarán cuando ocurran determinados sucesos en el sistema telefónico.
- La descripción detallada de las tramas de datos intercambiadas(Application Protocol Data Units, APDUs) para cada servicio y para cada evento. Este tipo de descripciones se pueden hacer en ASN.1 [Tanembaum, 1991] y también en XML.

El protocolo CSTA se encuentra en la versión 3 (llamada por ECMA fase III), los documentos que lo definen son ECMA-269 (servicios) y ECMA-285 (protocolo).

Para la primera versión (fase I) los documentos eran ECMA-179 (servicios) y ECMA-180 (protocolo).

CSTA es, por tanto, un conjunto de acciones y eventos que se pueden Invocar/detectar desde un computador conectado a un dispositivo telefónico. Es un estándar libre que cualquier fabricante de dispositivos puede adoptar en sus productos. CSTA podría ser utilizable desde un programa que implementase la correspondiente comunicación siguiendo los estándares anteriores. Sin embargo, lo habitual es que los programadores utilicen programas preexistentes que ocultan los detalles de comunicación, empaquetamiento de APDU’s, estos programas ofrecen una API

(Application Programs Interface) consistente en una colección de funciones que es mucho más fácil de utilizar. Este tipo de software intermedio se suele llamar “middleware” y TSAPI es un ejemplo de middleware. TSAPI es una traducción casi directa de los servicios y eventos CSTA y por eso se les suele identificar, pero es posible que el middleware sea un driver TSP y estemos usando un dispositivo CSTA a través de TAPI. De hecho, según documenta la ECMA, el CSTA se ha usado, hasta ahora, con los siguientes middlewares: TAPI (Microsoft), TSAPI (Novell, ATT y otros), JTAPI (Sun y otros), JavaPhone (Sun) y CallPath (IBM).

3.5.2 TAPI (Telephony Application Programming Interface)

El estándar TAPI fue creado conjuntamente por Microsoft e Intel. En el ambiente TAPI, la conexión física es creada a nivel de escritorio. Esto significa que el teléfono en su escritorio esta conectado a un computador. Aunque la conexión se ubique en su escritorio, su computador puede estar independiente o accesando datos en un servidor de archivos.

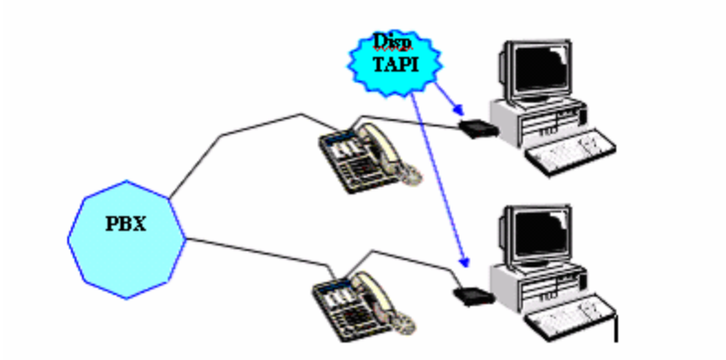


Fig. 3.5.2.1 Configuración TAPI.

Los vendedores de PBXs fabrican los dispositivos de hardware dependientes de TAPI para sus sistemas de teléfono. Los dispositivos pueden ser pequeñas placas de circuitos insertos en un teléfono, una tarjeta instalada en el computador, pero la mayoría son dispositivos externos. Un dispositivo externo debe tener inserto un cable en un puerto serial en la parte de atrás de su computador. El dispositivo debe tener

dos entradas telefónicas RJ-11, uno conectado al teléfono y otro conectado a la roseta de la pared. Estos dispositivos TAPI deben venir de un proveedor de PBXs y debe ser un dispositivo apropiado para un modelo de PBX particular que posee el cliente.

Para pequeñas compañías que no tienen un sistema telefónico de negocios, se le puede crear una conexión TAPI. Dispositivos TAPI como PATI 3000 están disponibles para una única línea telefónica análoga. Este dispositivo externo es similar al mencionado anteriormente, sin embargo, esta diseñado para uso con una línea análoga dedicada en lugar de un sistema telefónico digital. Una alternativa puede ser un modem que soporte el Unimodem TSP en Windows 95.

3.5.2.1 TSP (TAPI Service Provider).

El TSP es un driver software que esta escrito por un proveedor de PBXs para sus dispositivos TAPI. El TSP básicamente deja a otros programas conocer que conjunto de funciones de dispositivos TAPI utiliza. Un programa CTI middle-ware como PRIME's provee un link CTI que es necesario para utilizar una conexión CTI. El TSP, es parte vital de una instalación CTI exitosa.

La clave para establecer un ambiente de conexión CTI exitosa es a través del link CTI con el software PRIME's. Un link CTI de 32-bit tiene un costo bajo y un programa middle-ware telefónico-computador que soporte TAPI, TSAPI, Windows 95, Windows NT, Novell y bases de datos basadas en aplicaciones Windows. El link CTI utiliza OLE y DDE para mantener la integración de muchos administradores de contacto.

Los administradores de contacto y PIMs requieren un programa middle-ware como link CTI para comunicarse con un sistema telefónico que acepte TAPI o TSAPI.

3.5.3 TSAPI (Telephony Services Application Programming Interface).

El estándar TSAPI fue creado por AT&T (Lucent) y Novell. En un ambiente TSAPI la PBX esta físicamente conectada a un servidor de archivos en una red de computadores. En algunos casos la PBX puede estar conectada en un servidor telefónico separado en la red.

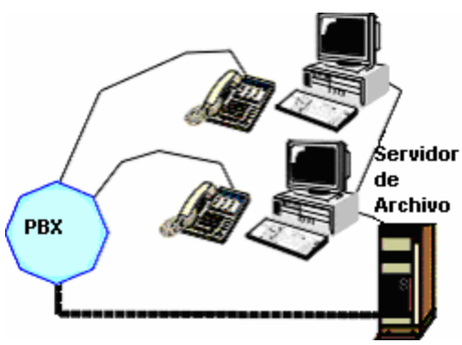


Fig. 3.5.3.1 Configuración TSAPI, la PBX es monitoreada por un servidor telefónico que esta directamente conectado a la PBX.

El hardware y el software requerido para la configuración de TSAPI es entregado por el proveedor de PBXs. Inicialmente el costo de TSAPI puede ser alto, pero para instalaciones de 25 a 30 usuarios CTI, TSAPI puede ser una solución mas barata por usuario. Cuando el software TSAPI es configurado, cada número de extensión telefónica del usuario CTI es asociada con su nodo en la red.

4. VOICEXML.

4.1 Introducción.

VoiceXML es un lenguaje de marcas para representar diálogos hombre computador, como el lenguaje HTML (Hyper Text Markup Language). Pero mientras que el HTML asume navegadores Web gráficos, con pantalla, teclado, y mouse, VoiceXML asume navegadores de voz con salida de audio (generado por computadora y/o pregrabado). VoiceXML ocupa el Internet para el desarrollo y la ejecución de aplicaciones de voz [W3C, 2001].

El lenguaje VoiceXML permite generar aplicaciones basadas en voz sin importar la plataforma tecnológica empleada, esto debido que esta en una capa superior de desarrollo, en la que se utiliza las capacidades de la tecnología, anexando dichas capacidades a su lenguaje de forma transparente. Además permite desarrollar aplicaciones tanto para Web como para telefonía tradicional, sin tener que modificar el código para una u otra tecnología o plataforma, lo cual conlleva a que la empresa disminuya sus costos considerablemente.

4.2 Raíces de VoiceXML

Las raíces de VoiceXML se remontan a varias reuniones informales en 1995 entre Dave Ladd, Chris Ramming, Ken Rehor y Curt Tuckey, todos ellos pertenecientes al laboratorio de investigación de AT&T. Ellos tenían una gran cantidad de ideas acerca de cómo la Internet podría afectar las aplicaciones de telefonía, cuando todas las piezas encajaran en un lugar en común: ¿por qué no tener un sistema de Gateway que provea un navegador que interprete un lenguaje marcado de diálogos de voz, que entregue el contenido Web y servicios a teléfonos ordinarios?. Esto comenzó en AT&T como un proyecto de teléfono Web. Cuando AT&T disolvió su alianza con Lucent el

proyecto de teléfono Web continuó por separado por ambas empresas, Chris se quedo en AT&T, Ken se traslado a Lucent, Dave y Curt los contrató Motorola.

A principios de 1999 AT&T y Lucent tenían dialectos incompatibles sobre el lenguaje marcado telefónico, Motorola tenía el nuevo VoxML, y otras compañías experimentaban con ideas similares, en particular IBM con SpeechML. Un lenguaje estándar debía ser diseñado para habilitar la voz sobre el Web. El original teléfono Web reunió a viejos amigos, ahora en distintas compañías, uniendo los esfuerzos individuales en un gran proyecto avalado por AT&T, Lucent y Motorola, comenzando lo que es hoy en día la organización del Forum de VoiceXML. IBM se unió como cofundador tiempo después. Desde marzo a agosto de 1999 un pequeño equipo de investigadores trabajaron en el forum para producir un nuevo lenguaje, VoiceXML 0.9, que combinaba las mejores características de los lenguajes anteriores y apuntaba a nuevas áreas, especialmente el soporte de DTMF y diálogos de iniciativa mixta. Después de que VoiceXML 0.9 fuera publicado, comenzó un periodo extensivo de comentarios, por la creciente comunidad del forum. Estos comentarios dieron como resultado mejoras enormes al lenguaje, incluyendo script del lado-cliente, propiedades y sub diálogos. Esto ocasionó que para marzo del 2000, saliera VoiceXML 1.0 y al menos 15 a 20 diferentes implementaciones disponibles.

El siguiente mes, el forum de VoiceXML sometió el lenguaje 1.0 al World Wide Web Consortium (W3C) para su consideración. En mayo, el W3C acepto VoiceXML, y este evento generó una gran cobertura de prensa, pero el merecido reconocimiento conlleva a una sumisión. El grupo de trabajo de navegadores de voz de la W3C tomó en sus manos la siguiente revisión del lenguaje. Este proceso tomó más tiempo de lo esperado por el forum, pero el énfasis y concienzudo interés de las muchas compañías participantes en un estándar robusto. El primer bosquejo público de trabajo de VoiceXML 2.0 fue publicado en octubre de 2001, luego en abril de 2002 una segunda revisión y en febrero de 2003 VoiceXML 2.0 es candidato a recomendación.

Los cambios de VoiceXML 1.0 a 2.0 fueron bastante conservadores, mucho del esfuerzo se avoco a clarificar los comportamientos esperados y corregir pequeños errores en su especificación [VoiceXMLforum, 2001].

A continuación se muestra un esquema global de la evolución de VoiceXML desde sus raíces.

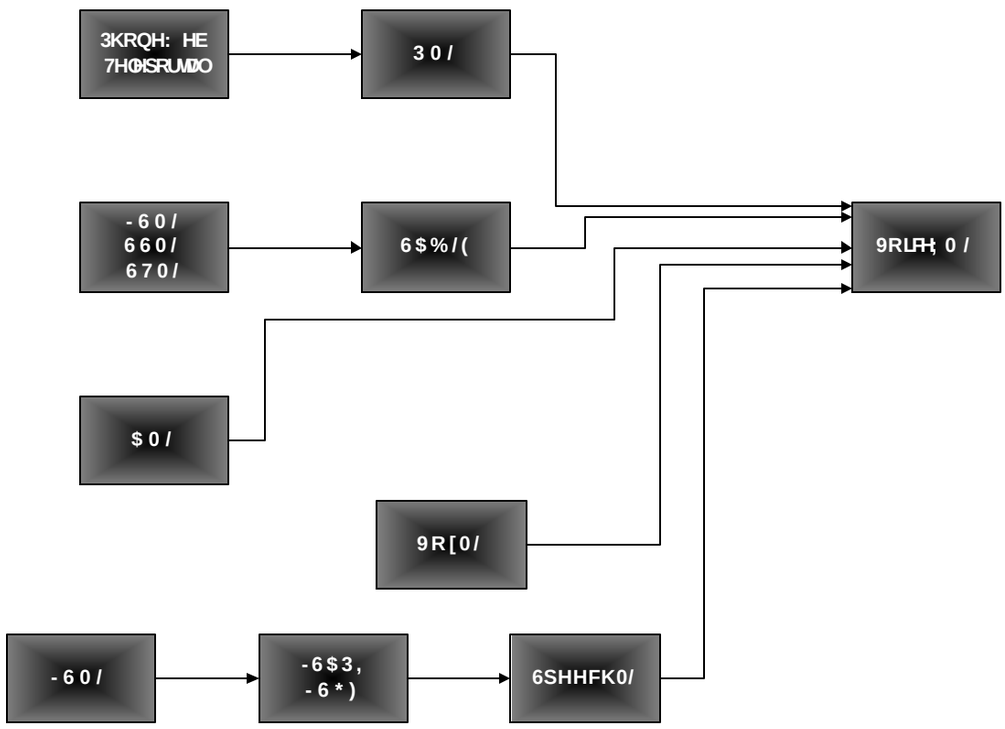


Fig. 4.2.1 Evolución de VoiceXML desde 1994 hasta 2002 [Sharma, 2002].

Tabla 4.2.1 Especificación Fig. 4.2.1 Evolución de VoiceXML.

Acrónimo	Especificación
JSML	Java Speech API Markup Language es un formato de texto usado por aplicaciones que escriben texto de entrada en sintetizadores de dialogo.
SSML	Speech Synthesis Markup Language es parte de una larga lista de especificaciones de marcas para navegadores de voz desarrollados a través de la W3C
STML	Spoken Text Markup Language es un lenguaje de características similares a SSML y JSML desarrollado por Bell-labs de Lucent company.
SABLE	La especificación envuelta por SABLE fue una iniciativa de unir y combinar criterios sobre tres lenguajes ya definidos: SSML, STML, JSML.
PML	Phone Markup Language es un lenguaje desarrollado por AT&T como un intento por simplificar el reconocimiento de diálogos.
AML	Agent Markup Language es un lenguaje desarrollado por Darpa como una especificación mas especifica de XML.
JSAPI	Java Speech API es una especificación que identifica una interfaz para programas java que utilizan TTS y ASR.
JSGF	Java Speech Grammar Format es una plataforma independiente de representación texto o gramáticas para ASR.
SpeechML	Speech Markup Language es una de las especificaciones más cercanas a VoiceXML desarrollada por IBM.
VoxML	Voice Markup Language es otra especificación cercana a VoiceXML desarrollada por Motorola.

4.3 Arquitectura de VoiceXML

El modelo arquitectónico asumido tiene los siguientes componentes (ver figura 4.3.1):

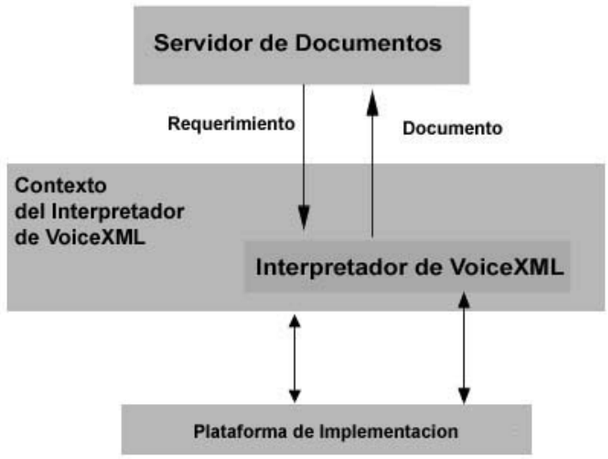


Fig. 4.3.1 Modelo Arquitectónico de VoiceXML.

Un servidor de documentos (por ejemplo un servidor Web) procesa los requerimientos de una aplicación cliente, el Interprete de VoiceXML, a través del Contexto del Intérprete de VoiceXML. El servidor produce documentos VoiceXML en respuesta, los cuales son interpretados por el Interpretador de VoiceXML. El Contexto del Intérprete de VoiceXML puede monitorear las entradas del usuario en paralelo con el Intérprete de VoiceXML.

La Plataforma de Implementación es controlada por el Contexto del Intérprete de VoiceXML y por el Intérprete de VoiceXML. Por lo tanto, en una aplicación de respuesta interactiva de voz, el Contexto del Intérprete de VoiceXML puede ser el responsable de una llamada entrante, recibiendo el documento VoiceXML inicial, y respondiendo la llamada, mientras el intérprete de VoiceXML conduce el diálogo después de la respuesta. La plataforma de implementación genera los eventos en respuesta a acciones del usuario (por ejemplo palabras o caracteres recibidos, desconectar) y eventos del sistema (por ejemplo tiempo de expiración). Algunos de estos eventos son realizados por el Intérprete de VoiceXML, especificado por el documento VoiceXML, mientras otros son realizados por el Contexto del Intérprete de VoiceXML.

4.4 Objetivos y Alcances.

El principal objetivo de VoiceXML es dar el mayor poder de desarrollo Web y entrega de contenido a las aplicaciones de respuesta de voz, y dejar libre a los programadores de esas aplicaciones de la programación de bajo nivel y manejo de recursos. Esto permite la integración de servicios de voz con servicios de datos usando el paradigma de cliente-servidor. Un servicio de voz es visto como una secuencia de diálogos de interacción entre el usuario y la Plataforma de Implementación. Los servidores de Documentos mantienen la lógica total del servicio, desarrollan las operaciones de bases de datos y producen los diálogos. Un documento VoiceXML

especifica cada diálogo de interacción para ser conducida por un Intérprete de VoiceXML. La entrada del usuario afecta la interpretación del diálogo y se recoge en las peticiones sometidas a un servidor del documento. El servidor de documento responde con otro documento VoiceXML y continúa la sesión del usuario con otros diálogos.

VoiceXML es un lenguaje de marcas que:

- Minimiza las interacciones Cliente-Servidor especificando múltiples interacciones por documento.
- Protege a los autores de aplicaciones de detalles de bajo nivel o específicos de la plataforma.
- Separa el código de interacción (en VoiceXML) del servicio lógico (script CGI)
- Promueve la portabilidad de servicios entre plataformas de implementación. VoiceXML es un lenguaje común para proveedores de contenido, proveedores de herramientas, y proveedores de plataformas.
- Es fácil de usar para interacciones simples, pero proporciona características de lenguaje para diálogos más complejos.

Además mientras que VoiceXML se esfuerza en acomodarse a los requisitos de la mayoría de los servicios de respuesta de voz, los servicios con requisitos complejos pueden ser mejor servidos por aplicaciones dedicadas que emplean un nivel de control más fino.

Dentro de los alcances del lenguaje describe la interacción hombre-máquina proveída por los sistemas de respuesta de voz, el cual incluye:

- Salida de habla sintetizada (text-to-speech).
- Salida a archivos de audio.
- Reconocimiento de ingreso hablado.
- Reconocimiento de ingreso DTMF.

- Grabación de ingreso hablado.
- Control de flujo del diálogo.
- Características Telefónicas como transferir llamada y desconectar.

El lenguaje proporciona los medios para recoger los caracteres y/o entrada hablada, asignando los resultados de las entradas a variables definidas en el documento, y haciendo las decisiones que afectan la interpretación de documentos escritos en el lenguaje. Un documento puede ser enlazado a otros documentos a través de URIs (Universal Resource Identifiers).

4.4.1 Características del lenguaje

VoiceXML es una aplicación XML, y tiene las siguientes características:

1. El lenguaje promueve la portabilidad de servicios, a través de la abstracción de los recursos de la plataforma.
2. El lenguaje se acomoda a la diversidad de plataformas en soportar formatos de archivos de audio, formato de gramáticas de lenguaje hablado, y esquemas URI. Mientras los productores de plataformas pueden soportar varios formatos de gramáticas el lenguaje requiere un formato de gramática común, llamada *XML* Form of the W3C Speech Recognition Grammar Specification, para facilitar la interoperatividad. Igualmente, mientras varios formatos de audio pueden ser soportados para grabación y emisión, los siguientes formatos de audio deben ser soportados:
    - Raw (headerless) 8kHz 8-bit mono mu-law [PCM] single channel. (G.711)
    - Raw (headerless) 8kHz 8-bit mono A-law [PCM] single channel. (G.711)
    - Wav (RIFF Header) 8kHz 8-bit mono mu-law [PCM] single channel.
    - Wav (RIFF Header) 8kHz 8-bit mono A-law [PCM] single channel.
3. El lenguaje soporta fácilmente la creación de tipos de interacciones comunes.

4. La lengua tiene semántica bien definida que preserve el intento del autor con respecto al comportamiento de interacciones con el usuario. La heurística del cliente no se requiere para determinar la interpretación del elemento del documento.
5. El lenguaje reconoce las interpretaciones semánticas desde gramáticas y hace esta información disponible a la aplicación.
6. El lenguaje tiene un mecanismo de control de flujo.
7. El lenguaje permite una separación de la lógica del servicio del comportamiento de la interacción.
8. No esta pensado para computación intensiva, operaciones de bases de datos, o sistemas antiguos. Estas se asumen son manejadas por recursos fuera del interpretador de documentos (por ejemplo el servidor de documentos).
9. La lógica general del servicio, manejo de estados, generación de dialogo, y secuenciamiento de diálogos se asume que reside fuera del interpretador de documentos.
10. El lenguaje provee formas de enlazar otros documentos utilizando URIs, y también de enviar datos a aplicaciones usando URIs.
11. VoiceXML provee formas de identificar exactamente que datos hay que enviar al servidor y por medio de cual método (POST o GET).
12. El lenguaje no requiere que los autores de documentos explícitamente la asignación o des asignación de recursos y el manejote la concurrencia, estos deben ser manejadas por la plataforma de implementación.

4.4.2 Características de diseño global para una aplicación básica.

Como ya se mencionó VoiceXML es un lenguaje de marcas extensible (XML), para la creación de reconocimiento automático de diálogos (ASR) y aplicaciones de respuesta de voz interactiva. Basado en el formato de XML, la sintaxis de VoiceXML encierra instrucciones dentro de una estructura de la siguiente manera:

< nombre _ elemento nombre_atributo="valor_atributo">

.....ítems contenidos.....

< /nombre _ elemento>

Una aplicación en VoiceXML consiste en uno o más archivos de texto llamados documentos. Estos documentos se identifican por la extensión “.vxml” y contienen varias instrucciones VoiceXML para la aplicación. Se recomienda que la primera instrucción en cualquier documento deba estar especificada para que el intérprete reconozca la versión de XML usada:

< ?xml version="1.0"?>

El resto de las instrucciones del documento debe incluir la etiqueta de vxml con la especificación de la versión de VoiceXML que es utilizada por el documento ("1.0" en el actual caso) como sigue:

< vxml version="1.0">

Dentro de las etiquetas de <vxml>, los documentos contienen elementos de quiebre de dialogo llamados *forms.*

Cada *form* tiene un nombre y es responsable de la ejecución de alguna parte de un dialogo en particular. Por ejemplo, se puede tener un *form* llamado “mainmenu” y los prompts utilizados, pueden ser seleccionados de una lista de opciones y entonces reconozca y ejecute la respuesta.

Un *form* se denota <form> y puede ser especificado por la inclusión de un atributo *“id”,* especificando el nombre del o los *forms.* Esto es útil si el form va a ser referenciado a algún otro punto de la aplicación o por otra aplicación. Por ejemplo:

<form id =”bienvenido”>

el cual le indica al documento VoiceXML el comienzo del form “bienvenido” como se muestra en la figura 4.2.2.1.

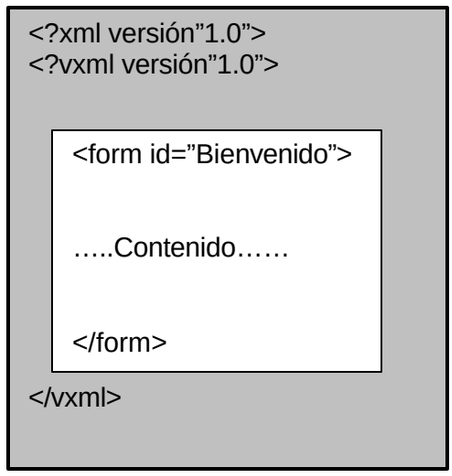


Fig. 4.2.2.1 Diagrama esquemático de un documento VoiceXML.

El form “bienvenido” puede contener varios elementos desempeñando las tareas que requiera el form. Entre los elementos posibles que se pueden contener en una forma están ésos señalados como “form ítems.”

A continuación se creara una aplicación simple en VoiceXML con el único propósito de escuchar un prompt al llamarlo. Para hacer esto, primero debemos tener un conocimiento básico de cómo usar *forms ítems.* Éstos son un grupo de elementos que se pueden incluir directamente debajo de la etiqueta < form>, para realizar las tareas requeridas por la aplicación.

Estos ítems están divididos dentro de dos grandes categorías: *field* ítems y *control* ítems. El ítem *field,* recopila la información de la llamada para llenar las variables (ítems de variables *field*).

Pueden contener los prompts dirigiendo lo que dice la llamada, que define la gramática y la interpretación de lo que dice y cualquier otro suceso relacionado. Los ítems de control, por otro lado, encierran las tareas basadas en no-reconocimiento.

Tabla 4.2.2.1 Lista de ítems *"form"* y *"control"* disponibles en la especificación 1.0 y 2.0

Ítems field	**Ítems control**
<**field**> - Las entradas son gavilladas vía reconocimiento de voz o DTMF que es definido por la gramática.	**<block>** - incluye una secuencia de las declaraciones para despliege de prompt.
<**record**> - Graba un clip de audio del usuario.	**<initial>** - Control de interacciones de iniciativa mixta con un form.
<transfer> - Transfiere al usuario hacia otro numero telefónico.	
<object> - Invoca un objeto de una plataforma especifica, que puede ser una entrada de usuario, retornando como resultado un objeto ECMAScript.	
<subdialog> - Realiza una llamada a otro *dialog* o documento (similar a la llamada a una función), retornando como resultado un objeto ECMAScript.	

Cada ítem form tiene asociado un ítem variable form que inicialmente es un conjunto valores indefinidos y no declarados ni asignados previamente (para los ítems form, esta es la misma variable ítem field). El nombre de la variable puede ser definido vía el atributo "nombre".

Existe una condición de resguardo para cada ítem que prueba si la variable tiene algún valor o no. Si no tuviera valor, la ejecución se salta el ítem en particular. De otra manera la ejecución procede normalmente.

A continuación se creó una aplicación de ejemplo, que usa la etiqueta <**block**> para englobar un simple prompt TTS llamado por el usuario a través de la aplicación. La aplicación se llamara prueba1.vxml y se muestra en la figura 4.2.2.3.

```
<?vxml versión =“1.0”?>
<vxml aplication=”prueba.vxml” versión=”1.0”>
        <form id=”test”>
                <block>
                <prompt> Hola Mundo</prompt>
                </block>
        </form>
</vxml>
```

Fig. 4.2.2.1 Ejemplo de una aplicación utilizando TTS.

5 APLICACIONES DISEÑADAS EN VOICEXML.

5.1 Introducción.

Se desarrollaron distintas aplicaciones, tratando de explotar las características más relevantes de VoiceXML.

VoiceXML es un lenguaje que puede ser modificado y mejorado de acuerdo a lo requerido por la empresa que diseña ambientes de desarrollo, para ofrecer al mercado, y ser mas competitivo, debido a esto los prototipos diseñados se validaron con tres ambientes de desarrollo provistos por tres empresas distintas:

1. Cambridge Voice Studio Versión 1.5[6].
2. VoiceGenie IDE 1.5[7].
3. Motorola Mobile ADK Voice Simulator 3.0 Beta[8].

Debido a que algunas de las funcionalidades requieren el uso de licencias de desarrollo, y en muchos casos es necesario el soporte de hardware (para realizar las pruebas), se empleo Cambridge Voice Studio para el desarrollo de sistemas de mayor complejidad, esto debido a que este entorno de desarrollo provee la mayoría de las funciones y sistemas de emulación para realizar las pruebas requeridas en los sistemas diseñados.

Se categorizaron las aplicaciones desarrolladas en base a su dificultad de programación: básica, medio, avanzado.

---

[6] Cambridge Voice Studio, Ambiente de desarrollo proporcionado por www.cambridgevoicetech.com
[7] VoiceGenie 1.5, es la versión de evaluacion disponible en para desarrollo en www.voicegenie.com
[8] Motorola mobile adk voice simulator 3.0 beta, esta versión esta disponible en www.motorola.com

5.2 Aplicaciones VOICEXML

Para realizar las pruebas del lenguaje VoiceXML, se programaron algunos servicios, como casos de ejemplo:

- Hola Mundo, es una aplicación que al momento de ser ejecutada despliega un mensaje predispuesto. (ver figuras: 5.2.1.2, 5.2.1.3, 5.2.1.4)

- Menú, esta aplicación ofrece opciones para que el usuario escoja una de las opciones entregándole la respuesta a la elección escogida. (ver figuras: 5.2.2.1, 5.2.2.2, 5.2.2.3, 5.2.2.4,5.2.2.5, 5.2.2.5, 5.2.2.6, 5.2.2.7, 5.2.2.8, 5.2.2.9, 5.2.2.10)

- Correo Automático, esta aplicación envía un email predeterminado a ciertas cuentas mediante algún evento. (ver figuras: 5.2.3.2, 5.2.3.3)

- Cuentamática, esta aplicación entrega el saldo de una cuenta bancaria y permite realizar depósitos. (ver figuras: 5.2.4.1, 5.2.4.2, 5,2,4,3, 5.2.4.4)

- Reservas, esta aplicación realiza la reserva de un salón se conferencias y también permite invitar a personas de una lista y además notificarles por email la fecha y lugar. (ver figuras: 5.2.5.1, 5.2.5.2, 5.2.5.3, 5.2.5.4, 5.2.5.5, 5.2.5.6, 5.2.5.7)

5.2.1 Hola Mundo.

La siguiente es la aplicación mas básica que se puede realizar, la cuál consiste en desplegar una respuesta al compilar, ya sea en modo TTS o ASR, en este caso se uso la simulación de ASR.

El código fuente de esta aplicación es:

```
<?xml version="1.0"?>
<vxml version="2.0">
        <form>
                <block>
                        <prompt>Hola Mundo! Esta es mi primera
                        aplicacion en VoiceXML.</prompt>
                </block>
        </form>
</vxml>
```

Fig. 5.2.1.1 Código fuente aplicación Hola Mundo.

A continuación se mostraran imágenes con la aplicación en VoiceXML ejecutadose en los distintos entornos de desarrollo empleados para las pruebas.

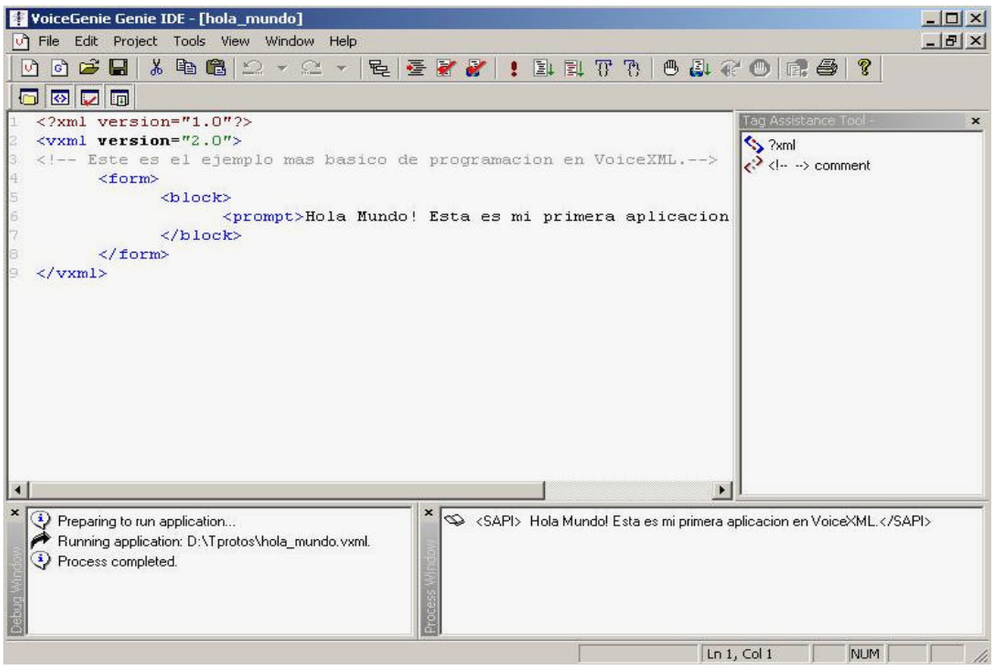


Fig. 5.2.1.2 Ambiente VoiceGenie.

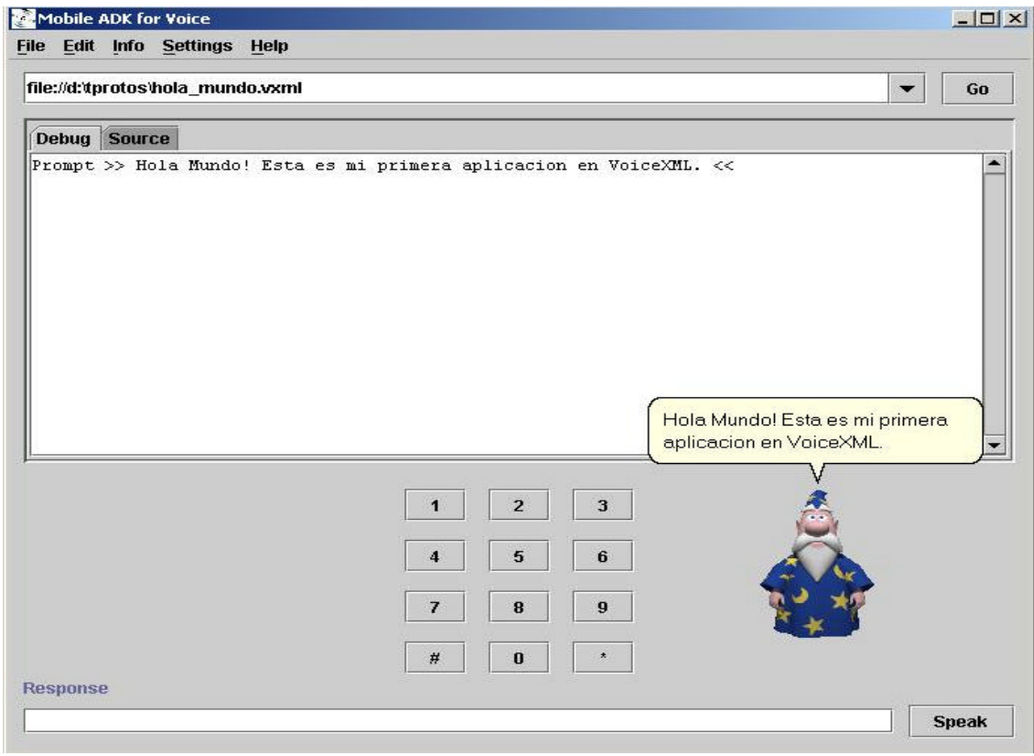


Fig. 5.2.1.3 Ambiente Motorola Mobile ADK.

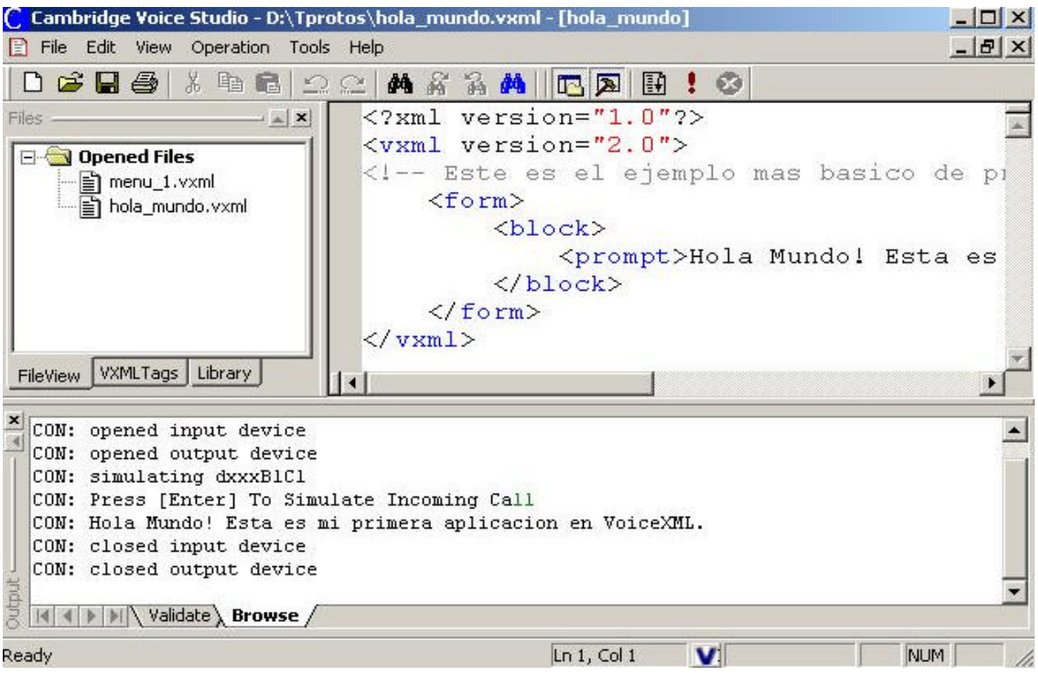


Fig. 5.2.1.4 Ambiente Cambridge Voice Studio.

5.2.2 Aplicación Menú.

La aplicación Menú es un poco más compleja de tipo media, esta ofrece al usuario tres posibilidades de elección entregando el sistema una respuesta, en función a la opción elegida.

```
<?xml version="1.0"?>
<vxml version="2.0">
 <menu id="main_menu" dtmf="true">
   <prompt> Que servicio desea consultar?
         Internet, seguridad, telefonos, escoja alguna de las opciones.
   </prompt>
   <choice next="internet.vxml">Internet</choice>
   <choice next="seguridad.vxml">Seguridad</choice>
   <choice next="telefonos.vxml">Telefonos</choice>
   <choice event="event.myapp.goodbye" dtmf="9">adios.</choice>
    <catch event="noinput nomatch">
            <audio>Lo siento no le he entendido.</audio>
            <reprompt />
     </catch>
 </menu>
</vxml>
```

Fig. 5.2.2.1 Código fuente aplicación Menú.

```
<?xml version="1.0"?>
 <vxml version="2.0">
   <form>
     <block>
       Bienvenido al sistema de Cosultas internet.
     </block>
   </form>
</vxml>
```

Fig. 5.2.2.2 Código fuente aplicación Menú opción Internet.

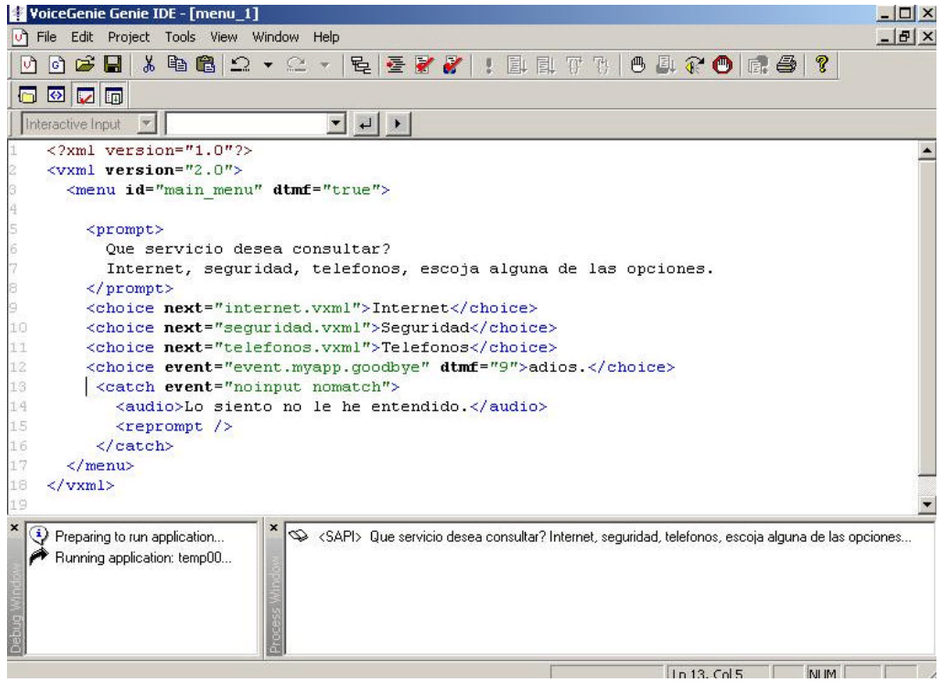


Fig. 5.2.2.3 Menú en ambiente VoiceGenie

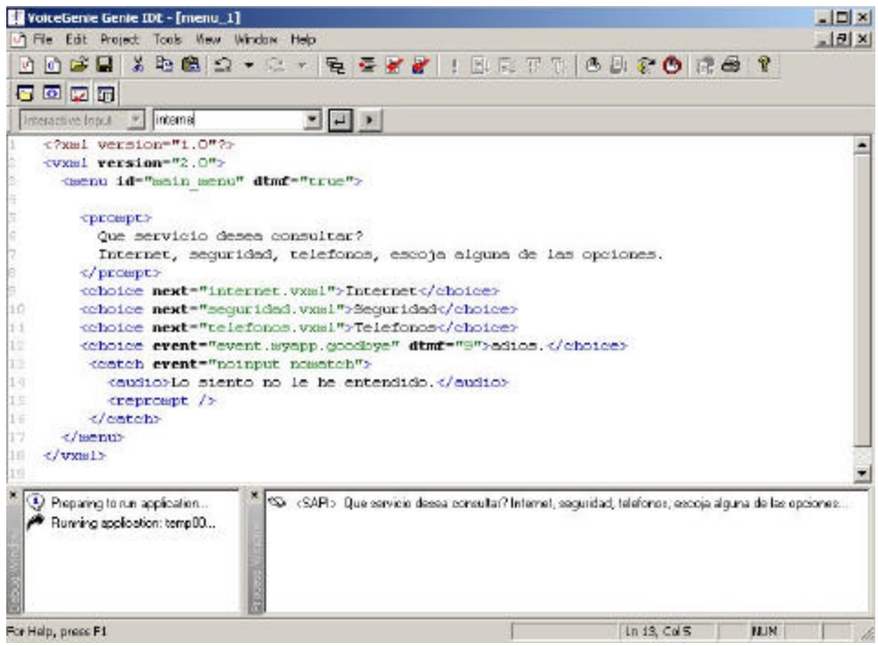


Fig. 5.2.2.4 Ingreso opción Internet en ambiente VoiceGenie

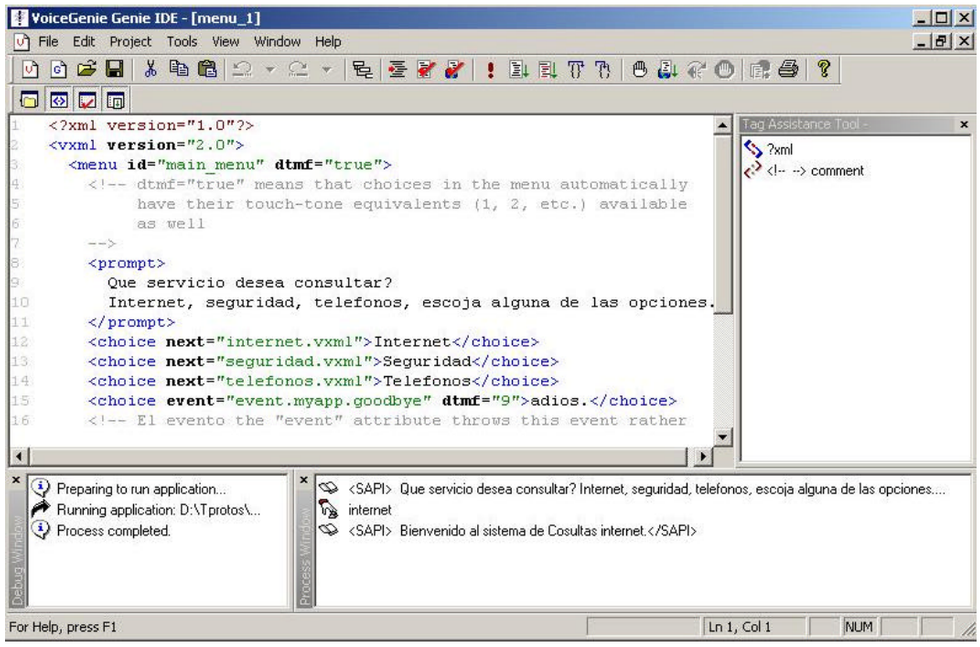


Fig. 5.2.2.5 Respuesta opción Internet en ambiente VoiceGenie

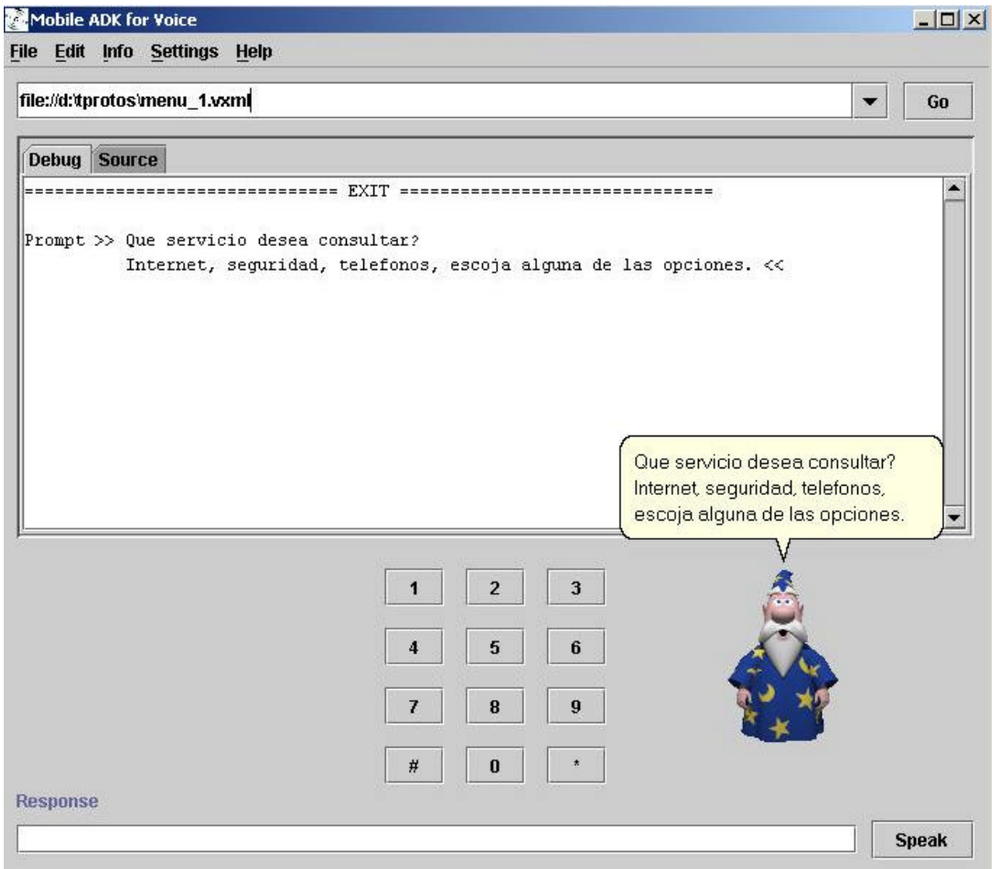


Fig. 5.2.2.6 Menú en Motorola Mobile ADK

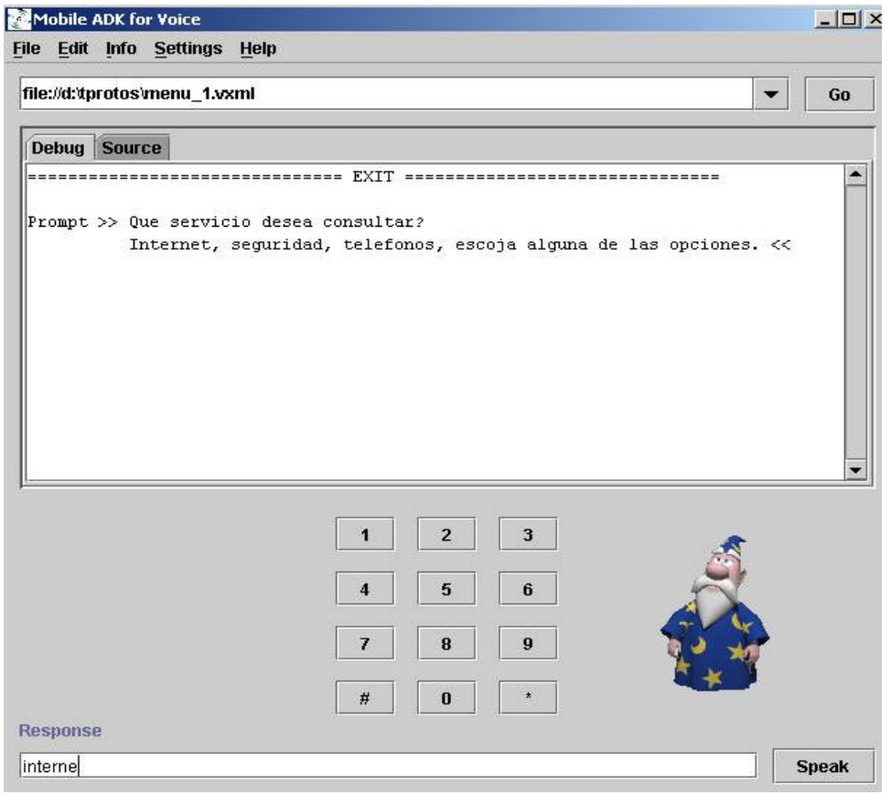


Fig. 5.2.2.7 Ingreso opción Internet en Motorola Mobile ADK

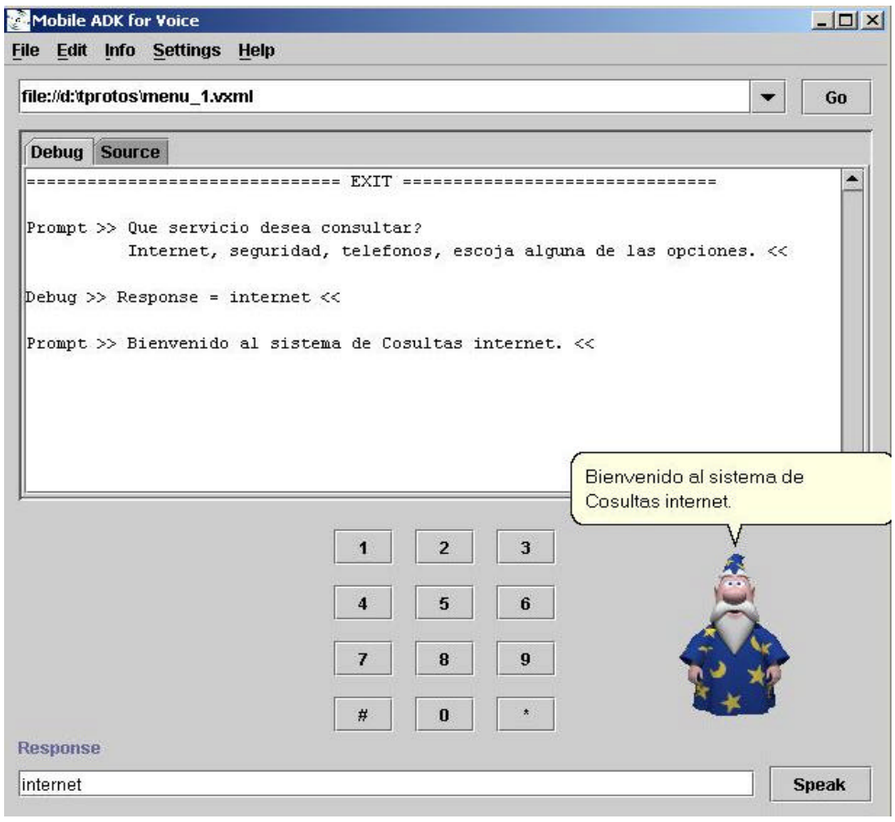


Fig. 5.2.2.8 Respuesta opción Internet en Motorola Mobile ADK

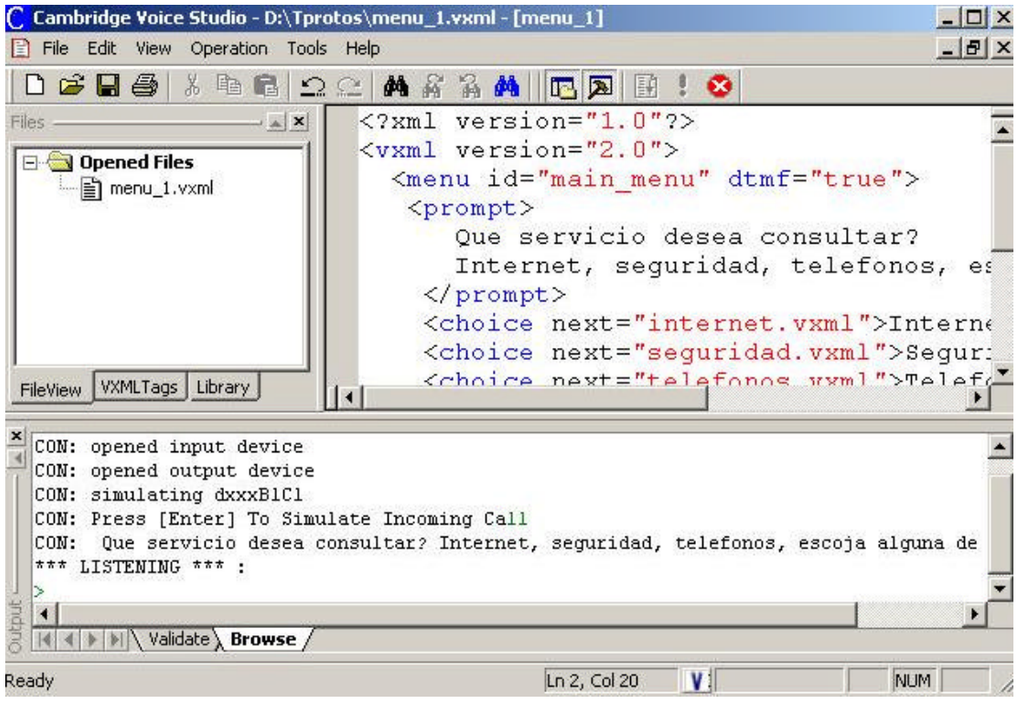


Fig. 5.2.2.9 Menú en Cambridge Voice Studio

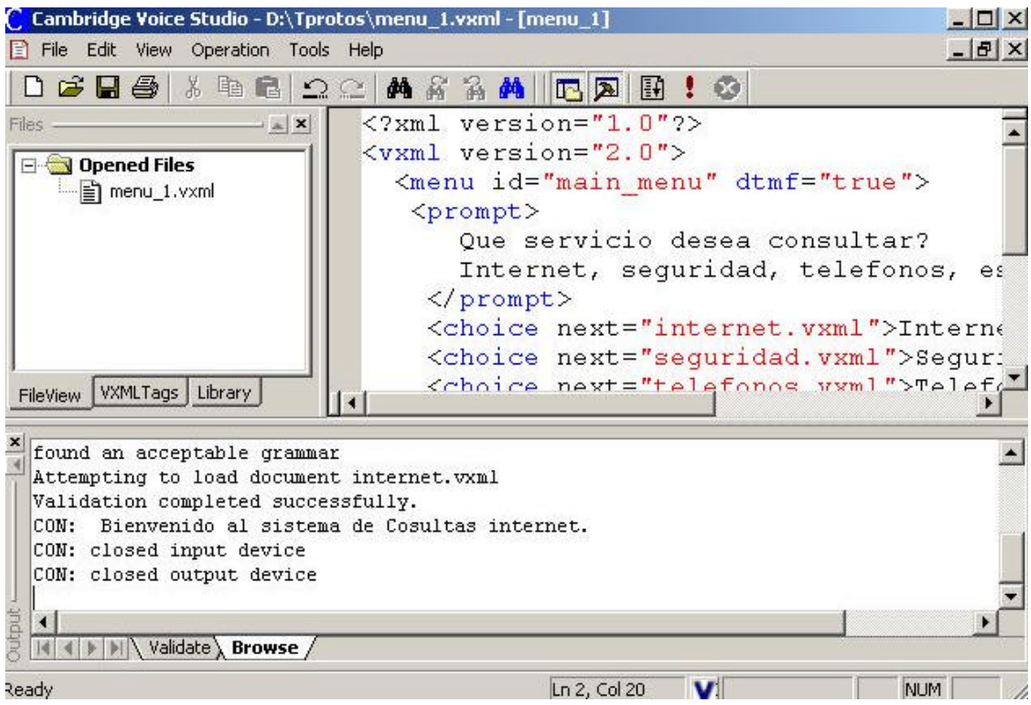


Fig. 5.2.2.10 Opción y respuesta Internet en Cambridge Voice Studio

5.2.3 Correo Automático.

La aplicación Correo Automático, puede ser llamada por una aplicación para que en determinado momento se gatille la acción de envió de correo electrónico.

La aplicación mostrada a continuación, paso el nivel de básico a medio, esto debido a que utiliza rutinas diseñadas por un proveedor especifico y aun no están integrados en todos los ambientes de desarrollo, debido a esto la aplicación en Motorola Mobile ADK en su versión 3.0 no es soportada.

```
<?xml version="1.0"?>
<vxml version="1.0" mode="TTS">

 <form>
   <block>
         Este es un ejemplo de como enviar correo electronico
        automaticamente   a traves de VoiceXML.
   </block>
   <block>
    <script><![CDATA[
      var msg = emailCreateMessage();
      emailSetTo(msg, "rrosas@inf.uach.cl");
      emailSetFrom(msg, "irreal@entelchile.net");
      emailSetBodyText(msg, "esta es una prueba");
      emailSetSubject(msg, "Este es un correo de VOICEXML ");
      emailSendMessage(msg,"smtp.entelchile.net");
    ]]></script>
   </block>
  </form>
</vxml>
```

Fig. 5.2.3.1 Código fuente Correo Automático.

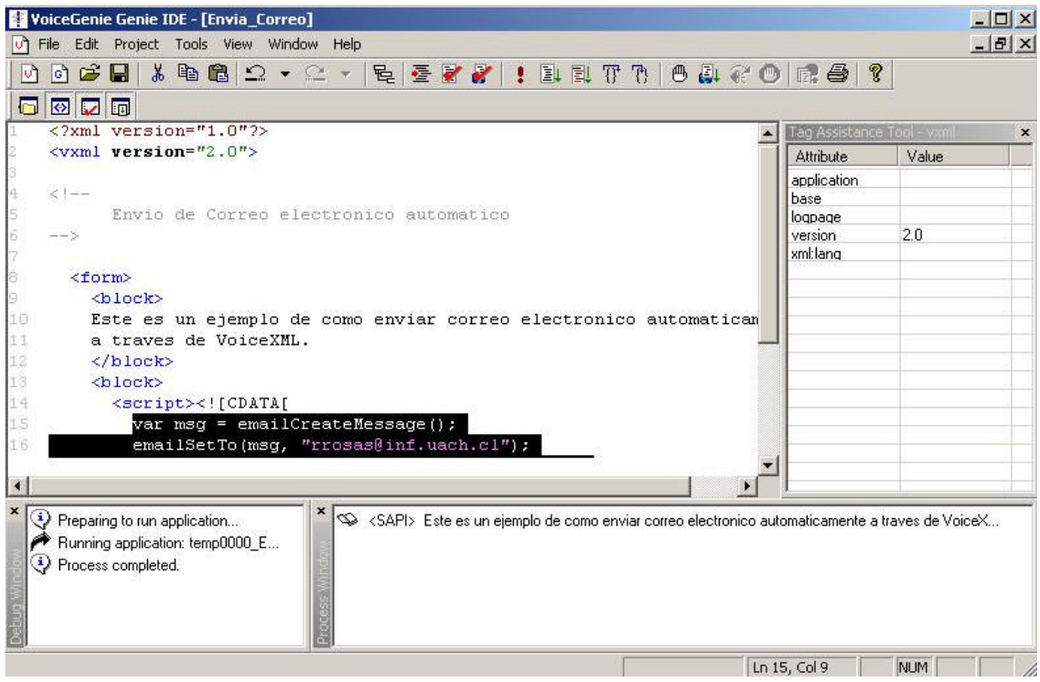


Fig 5.2.3.2 Envío automático de correo electrónico en VoiceGenie.

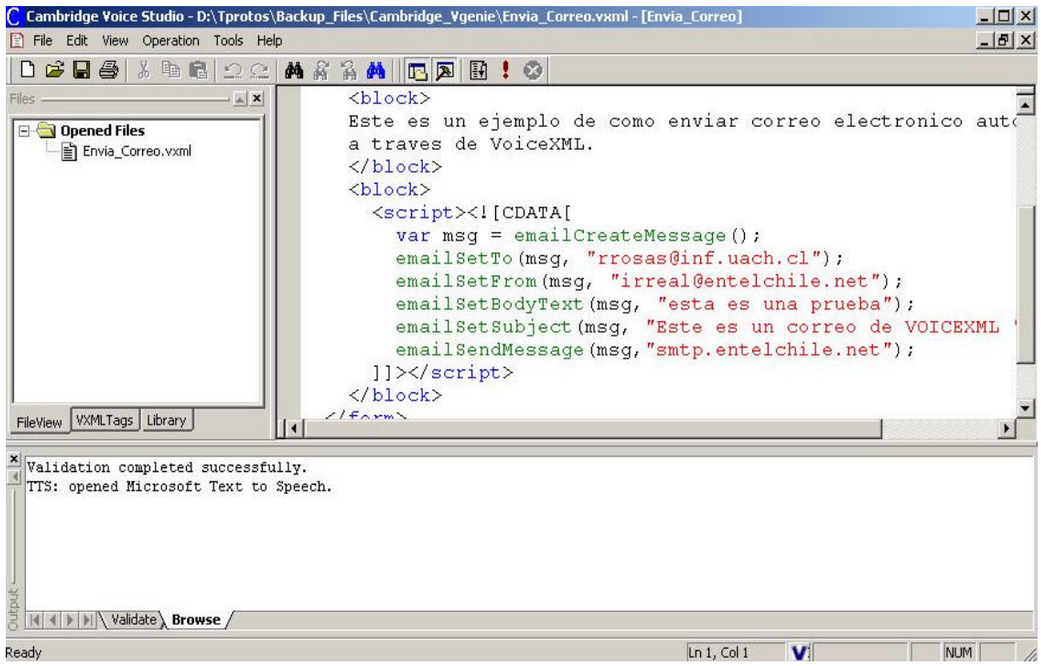


Fig. 5.2.3.3 Envío automático de correo electrónico en Cambridge Voice Studio.

5.2.4 Cuentamática

La aplicación Cuentamática ya presenta un grado de complejidad avanzado, es un pequeño sistema de consulta de saldo, y deposito bancario básico, para realizar dichas acciones accede una base de datos, es indiferente el tipo debido a que su conexión es por ODBC.

De los tres Software utilizados solo el Cambridge Voice Studio permitió realizar una aplicación de estas características.(ver código fuente en anexo I1)

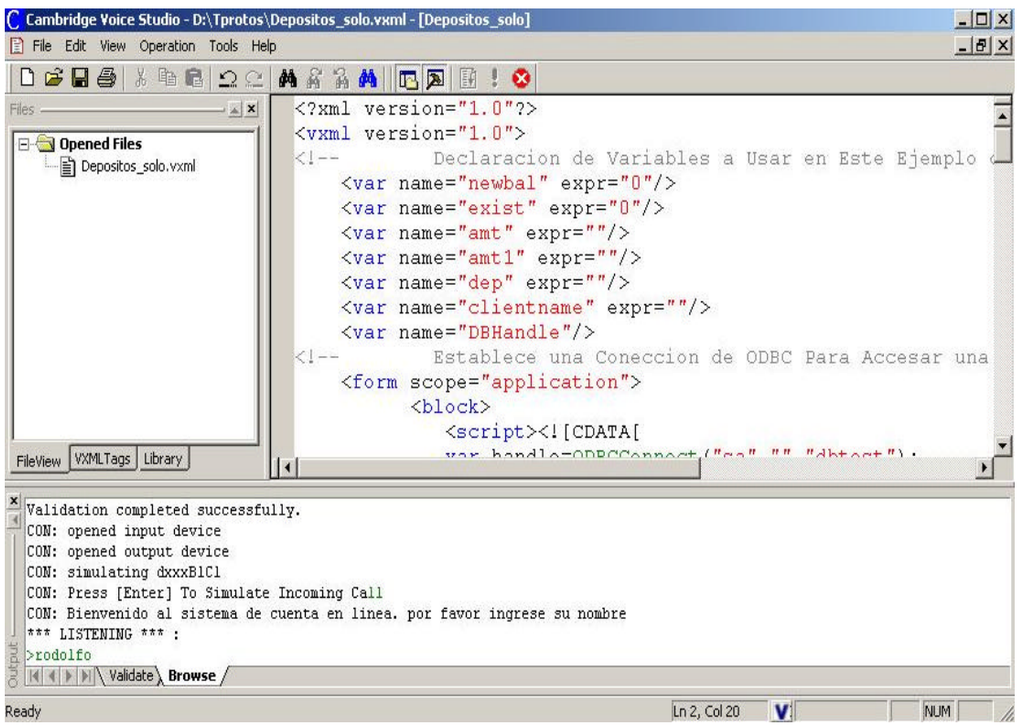


Fig. 5.2.4.1 Ingreso al Sistema.

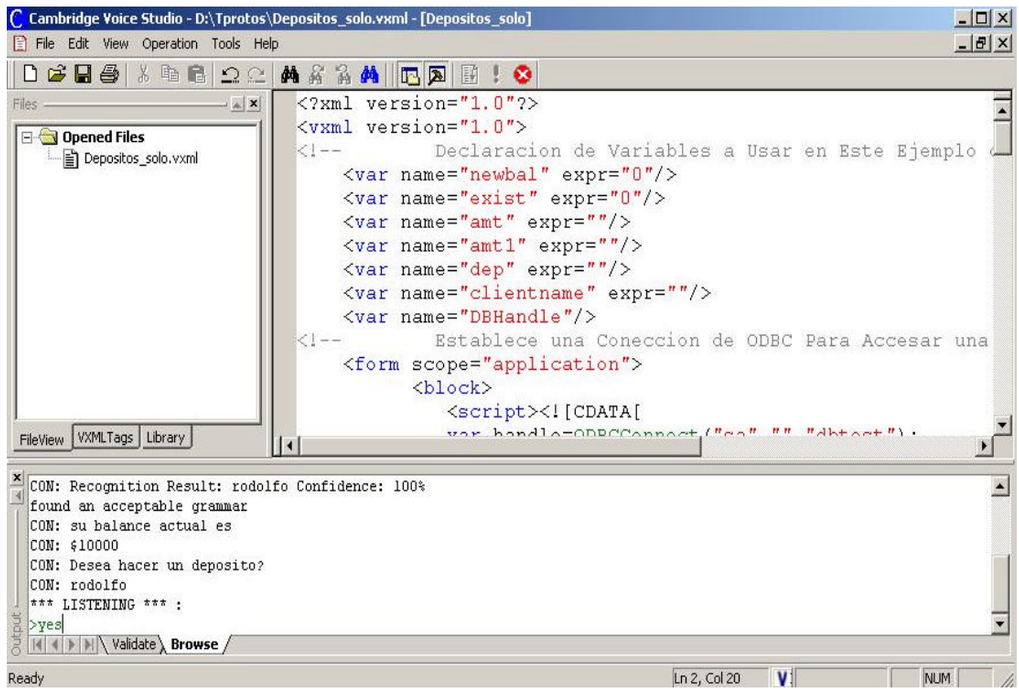


Fig. 5.2.4.2 Selección opción de deposito.

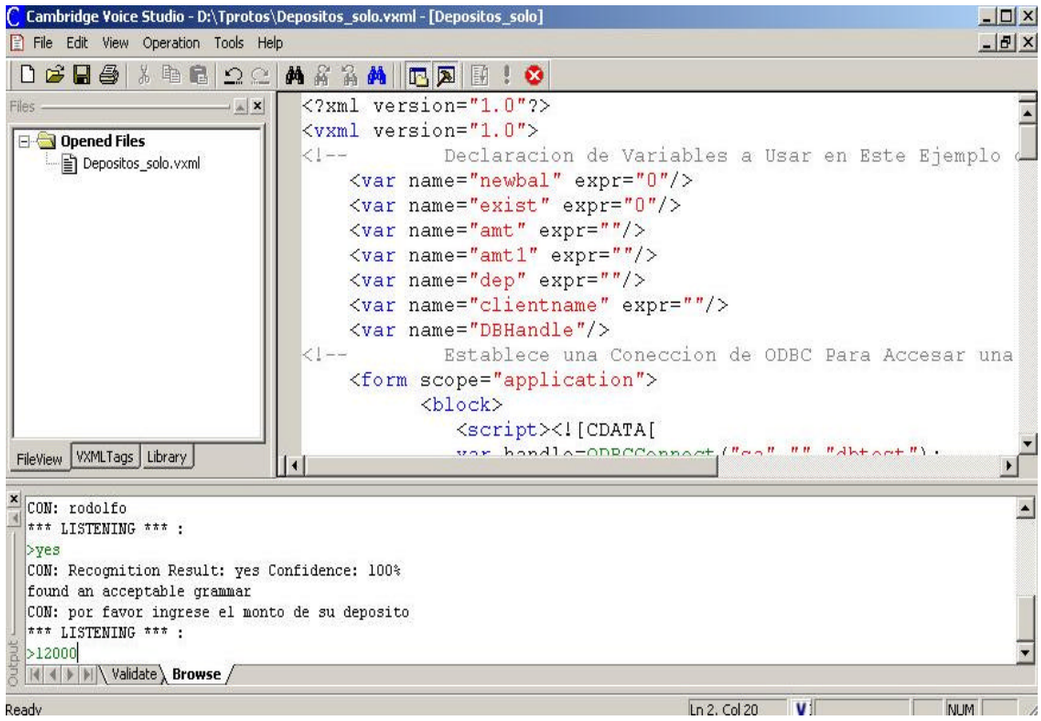


Fig. 5.2.4.3 Ingreso de monto a depositar.

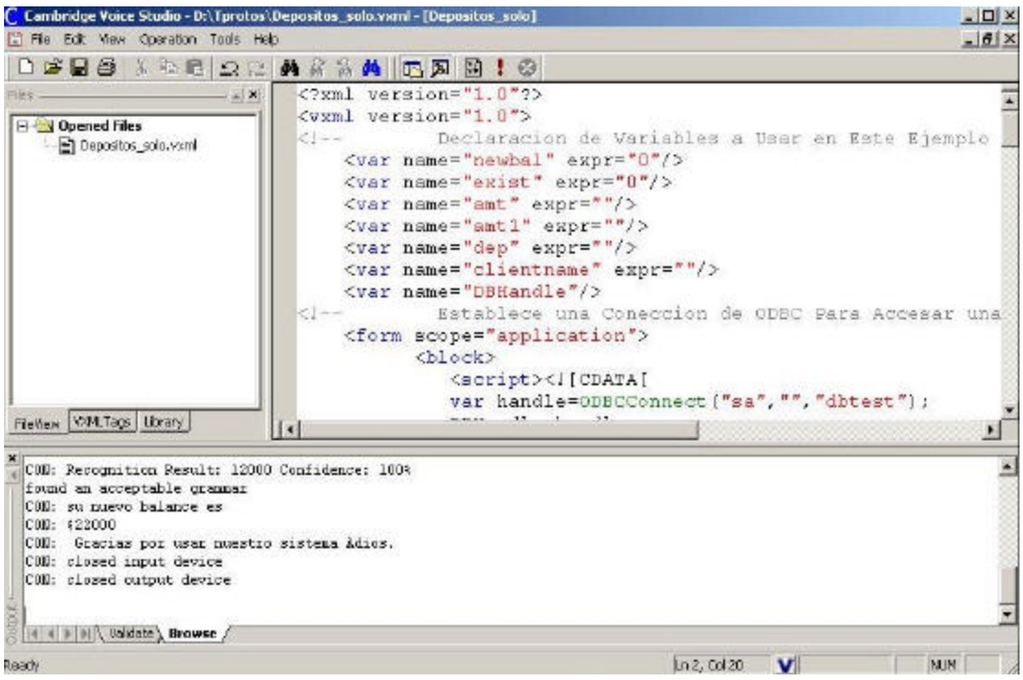


Fig. 5.2.4.4 Balance actualizado y despedida del sistema.

5.2.5 Reservas

La aplicación Reservas tiene una complejidad avanzada, realiza la reserva de un salón de conferencias para una fecha y hora en particular y permite invitar a ciertas personas inscritas en el sistema para dicha charla mediante un correo electrónico. (ver código fuente en anexo I2)

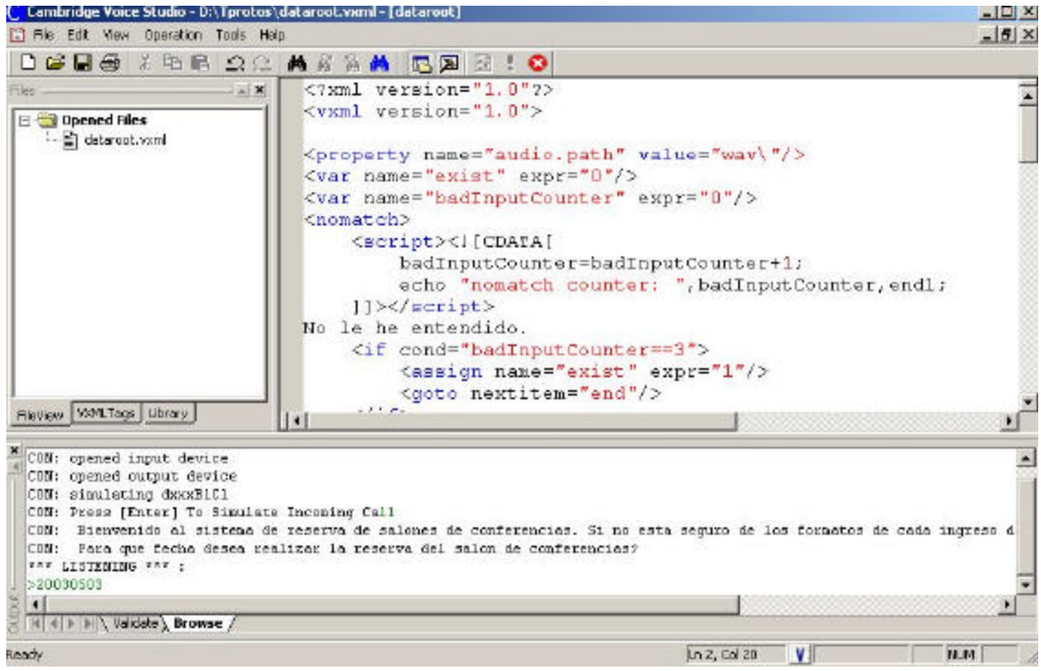


Fig. 5.2.5.1 Ingreso al sistema e ingreso de fecha para la solicitud.

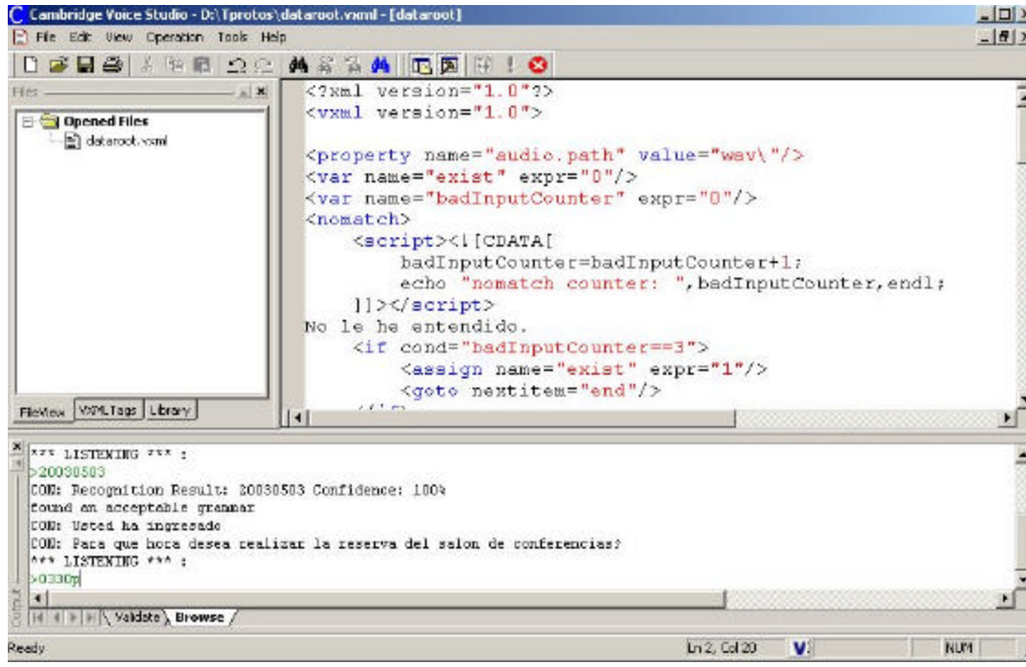


Fig. 5.2.5.2 Ingreso de la hora para la solicitud.

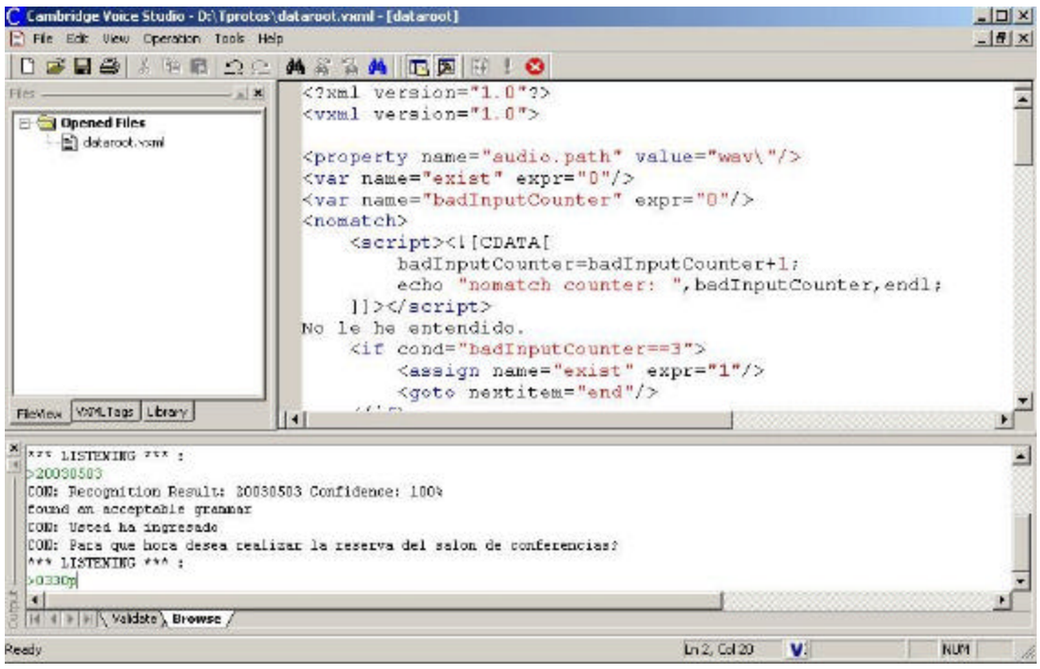


Fig. 5.2.5.3 Confirmación del sistema de la solicitud ingresada.

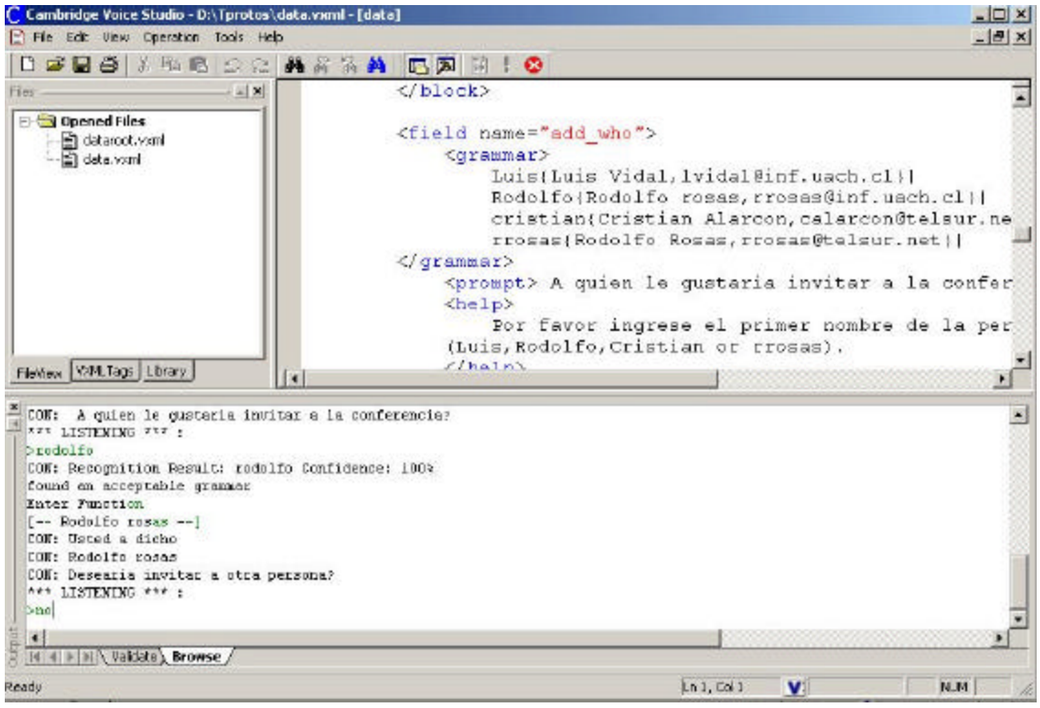


Fig. 5.2.5.4 Consulta del sistema si se desea invitar a alguien más a la Charla.

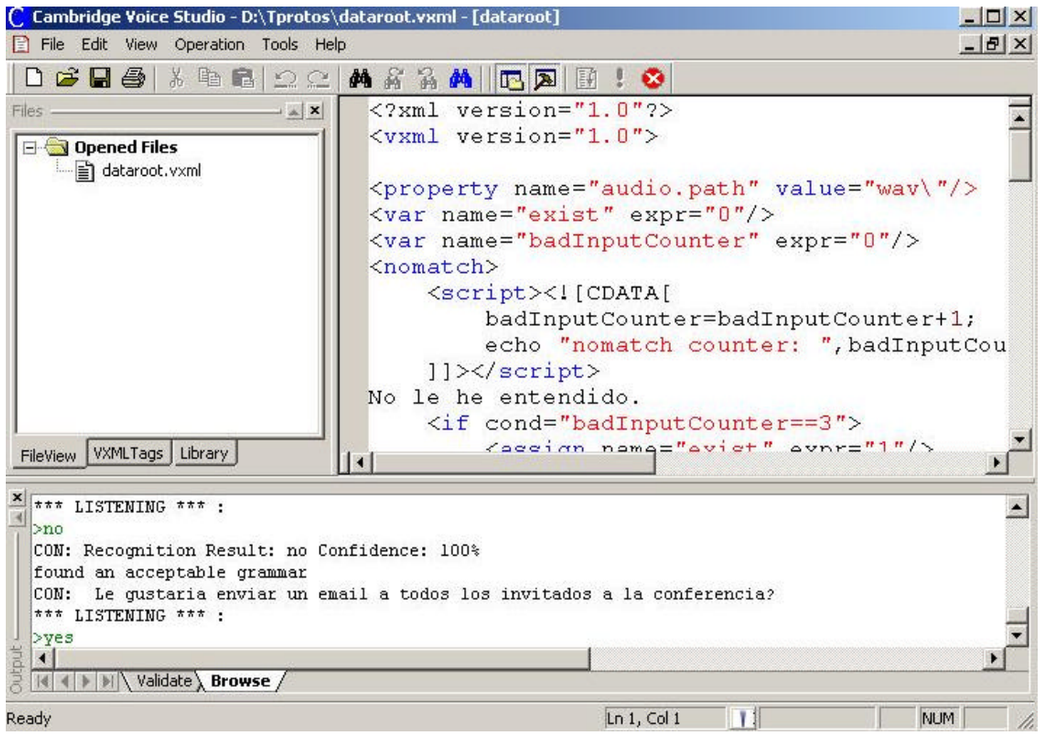


Fig. 5.2.5.5 Consulta si se desea enviar un email de invitación a las personas invitadas.

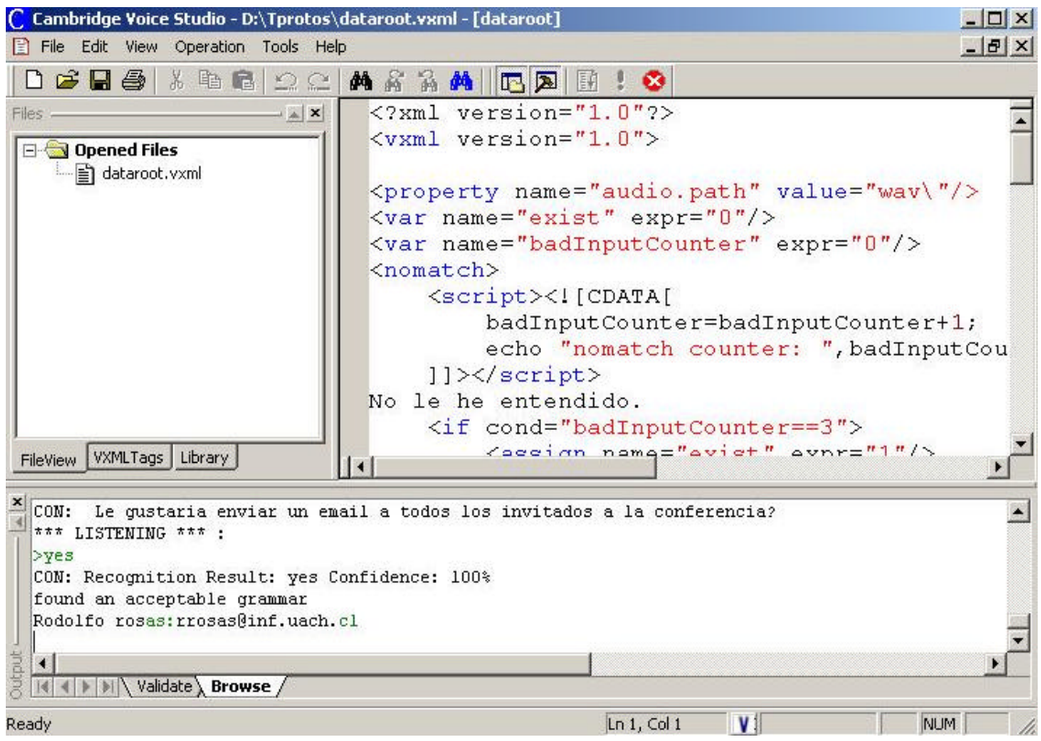


Fig. 5.2.5.6 Realiza el envío de la invitación a través de un email a las personas antes seleccionadas.

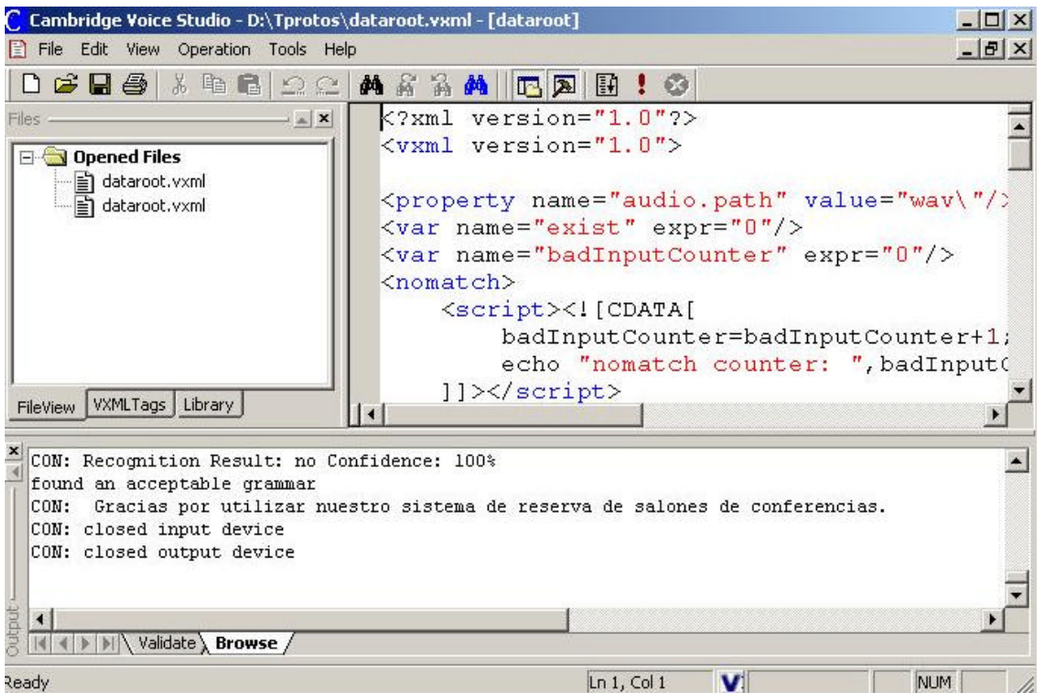


Fig. 5.2.5.7 Se despide el sistema y se sale de la aplicación.

6 IMPLEMENTACION DE ARQUITECTURA VOICEXML.

6.1 Introducción

En este capitulo se entregaran pautas que permitan la implementación real de sistemas CTI basados en VoiceXML, para ello se indican sugerencias al momento de implementar o migrar un sistema CTI, además se programa un sistema de cobro revertido, el cual se basa y compara con un sistema real.

6.2. Estrategias de Decisión.

La decisión a este punto tiene distintos ribetes a considerar, esto debido a que las características necesarias para incurrir en una inversión deben ser analizadas con detenimiento, para esto se señalaran algunas directrices básicas a considerar al momento de evaluar una estrategia de implementación u migración de IVRs.

6.2.1 Comenzar en Pequeño.

Una vez que se ha detectado cuales son los requerimientos de los actuales negocios de la empresa y directrices futuras, se debe comenzar por familiarizarse con el entorno a evaluar, esto mediante la prueba de la tecnología, VoiceXML a diferencia de otras tecnologías de IVR, tiene la característica de que existen servidores gratuitos donde se pueden desarrollar pequeñas aplicaciones y probarlas on-line, esto es a través de una llamada telefónica, pagando solo la duración de esta.

6.2.2 Comprar vs. Desarrollar.

Este es una de los puntos cruciales al momento de decidir, debido a que la premisa a escoger será la que marcara de cierta manera el destino de la empresa en los futuros desarrollos u generación de soluciones para proveer una mayor gama de servicios a sus clientes. El desarrollar lleva consigo el costo de comprar hardware y el software necesario para soportar la tecnología, en cambio el comprar la alternativa de solución que es utilizar un ISP que provea servicios de voz, (ASR, TTS) es una alternativa también a considerar.

6.2.3 Evaluar los Proveedores.

Al ya haber tomado la decisión de cual va a ser la solución a utilizar (comprar o desarrollar), se debe evaluar a los proveedores y la tecnología ofrecida por estos. La mayoría de los proveedores trabajan en asociación con otras empresas (partnerships) esto no se debe desestimar, debido a que los convenios suscritos por los proveedores afectaran directa o indirectamente a la empresa en busca de la solución, esto debido a que las soluciones basadas en voz existentes constan de varias partes y piezas. Existen dos puntos a considerar: el primero es que no se debe sustentar de un unico proveedor de tecnología, debido a que ante una situación inesperada si el proveedor tiene problemas no afecte a la empresa que compro la solución; segundo, escoger un proveedor de tecnología con buen futuro en el mercado, avalado por su experiencia en el área (Ej. IBM, AT&T, etc.).

Para entregar una visión global se señalan en las siguientes tablas, los lideres en el mercado de las soluciones de voz, hardware y software, disponibles en el mercado al momento del desarrollo de este trabajo de tesis.

Tabla 6.2.3.1. Lista de Proveedores de tecnología de Voz y Características Globales.

Tecnología	**Proveedores**
Motores de Reconocimiento de Voz y TTS (Software).	➢ Conversay (ASR[9]/TTS[10]/Embedded) ➢ Enuncia (TTS) ➢ Fonix (ASR/TTS/Embedded) ➢ IBM (ASR/TTS/Embedded) ➢ Inzigo (NLU[11]) ➢ L&H (ASR/TTS/Embedded) ➢ Nuance (ASR/TTS/NLU) ➢ Philips Speech Procesing (ASR/NLU) ➢ Phonetics (ASR) ➢ Poly Information (NLU) ➢ SpeechWorks (ASR/TTS) ➢ Telelogue (NLU)
Tarjetas de Telefonía (Hardware)	➢ Aculabs ➢ Audiocodes ➢ Brooktrout Technology ➢ Dialogic (Intel) ➢ Natural Microsistems
Plataformas VoiceXML	➢ Bevocal ➢ Cisco ➢ Comverse ➢ Conversary ➢ General Magic ➢ IBM DirectTalk ➢ Nuance ➢ Pipebeach ➢ SpeechWorks ➢ Telera ➢ Tellme Networks ➢ Verscape ➢ VoiceGenie
Arquitecturas IVR (Hardware)	➢ Aspect ➢ Avaya ➢ Bevocal ➢ Comverse ➢ Edify ➢ IBM ➢ Intervoice-Brite ➢ Lucent ➢ Nortel Periphonics ➢ Syntellect ➢ Telere
Integración con Call-Center.	➢ Alcatel Genesys ➢ Aspect ➢ Avaya ➢ Cisco

[9] ASR, Autometed Speech Recognition.
[10] TTS, Text to Speech.
[11] NLU, Natural Language Understanding.

Tabla 6.2.3.2. Lista de Proveedores de herramientas de desarrollo de VoiceXML.

Tecnología	**Proveedores**
Herramientas de Desarrollo	➢ Alterego ➢ Covigo ➢ Fonelet ➢ Gold Systems ➢ IBM ➢ IConverse ➢ Intervoice-Brite ➢ Microsoft ➢ Motorota ➢ Nuance ➢ Philips ➢ SpeechWorks ➢ Tellme Networks ➢ Unisys ➢ VoiceGenie ➢ Voxeo

Tabla 6.2.3.3. Lista de Compañías que Ofrecen Soluciones en el Espacio del Consumidor.

Aplicación	**Compañías**
Información a los Consumidores.	➢ AOL by Phone ➢ Audiopoint ➢ Bevocal ➢ Lycos ➢ MapQuest ➢ Sonic Factory ➢ Telera ➢ Tellme Networks ➢ Yahoo! By Phone
Consumidores Comerciales.	➢ AOL MovieFone ➢ NetByTel ➢ ShopTalk ➢ TellMe Networks ➢ TicketMaster ➢ Vicinity
Mensajeria Unificada	➢ AOL By Phone ➢ Etrieve ➢ Evoice ➢ Micostrategy ➢ Openware (Phone.Com/OneBox) ➢ Shoutmail ➢ Soundbite ➢ Webley ➢ Yahoo! By Phone

Tabla 6.2.3.3. Lista de Compañías que Ofrecen Soluciones en el Espacio de las Aplicaciones Empresariales.

Aplicación	**Compañías**
Fuerza de Trabajo para Empresas Móviles	➢ Bevocal ➢ Conita ➢ HeyAnita ➢ iHello ➢ JustTalk ➢ Siebel ➢ Telcos (en Sociedad con otras) ➢ Telera ➢ TellMe Networks ➢ Webversa
Administración de la relacion de clientes (Empresas). (CRM[12])	➢ Appiss ➢ Hanover Communications ➢ Interactive Telesys ➢ Nextel ➢ Placeware ➢ ShopTalk ➢ Siebel ➢ SoundBite ➢ Telcos (en Sociedad con otras) ➢ Telera ➢ TellMe Networks ➢ Vail Systems

.

6.3 Implementación de servicio de cobro revertido empleando un entorno de desarrollo propietario vs. VoiceXML.

Para realizar una comparación totalmente objetiva se programo un sistema de cobro revertido en VoiceXML.

El funcionamiento de la aplicación es similar a un servicio programado en una plataforma comercial provista por la empresa APEX[13] .

[12] CRM Customer Relationship Management.
[13] APEX, Empresa Proveedora de Plataformas IVR, http://www.apexvoice.com

6.3.1 Descripción del entorno de desarrollo comercial APEX.

El entorno de programación provisto por APEX lleva por nombre Omniview. Este es el ambiente para desarrollo de servicios de Omnivox y provee la facilidad de una programación grafica. Este ambiente esta optimizado para UNIX y Windows NT, provee iconos de comandos que pueden ser enlazados para diseñar el flujo de la aplicación. Cada icono desarrolla una función específica como decepcionar un DTMF o grabar un mensaje.

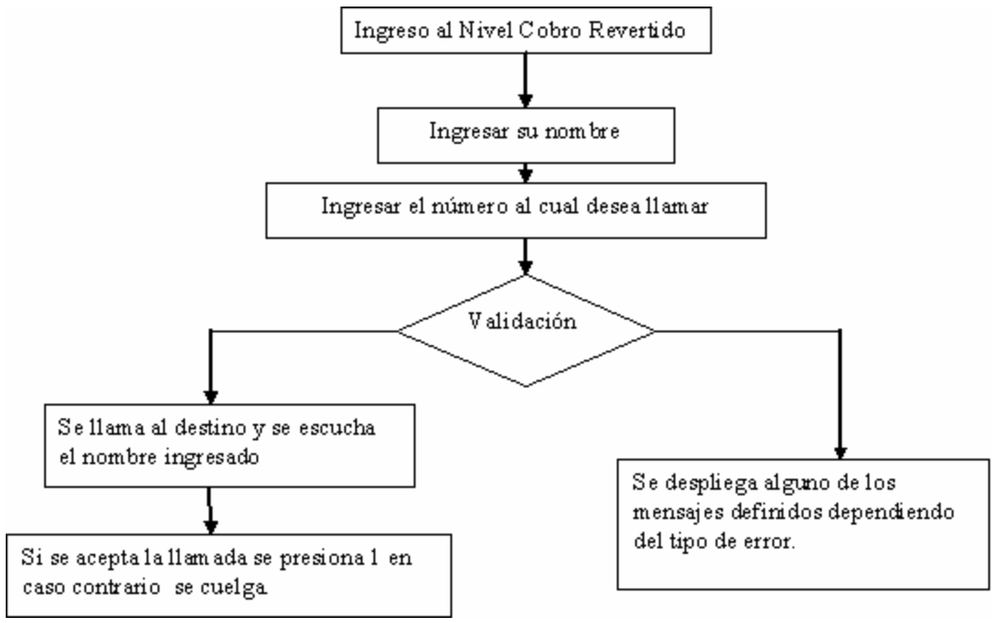
Omniview para UNIX también provee API en lenguaje C, para el desarrollo de las interfaces no graficas (consultas a base de datos, otros sistemas, etc.).

Omniview para Windows NT que tiene las mismas características de programación grafica, y para el desarrollo de APIS se puede desarrollar en Visual Basic y Visual C++.

6.3.2. Descripción de la aplicación.

Para dimensionar el alcance de la aplicación es necesario entender el funcionamiento básico del sistema de cobro revertido, para ello se ilustra en diagrama de flujo en la figura 6.3.2.1.

Parte del código fuente VoiceXML se adjunta en anexo, el código fuente completo no se incluye debido a su similitud con el sistema de cobro revertido real contra el cual se realiza la comparación.



6.3.2.1 Diagrama de flujo Básico Cobro Revertido.

La aplicación cobrorevertido en la plataforma APEX en su entorno grafico se muestra a continuación y posteriormente la aplicación desarrollada en VoiceXML también se muestra en su entorno Gráfico.

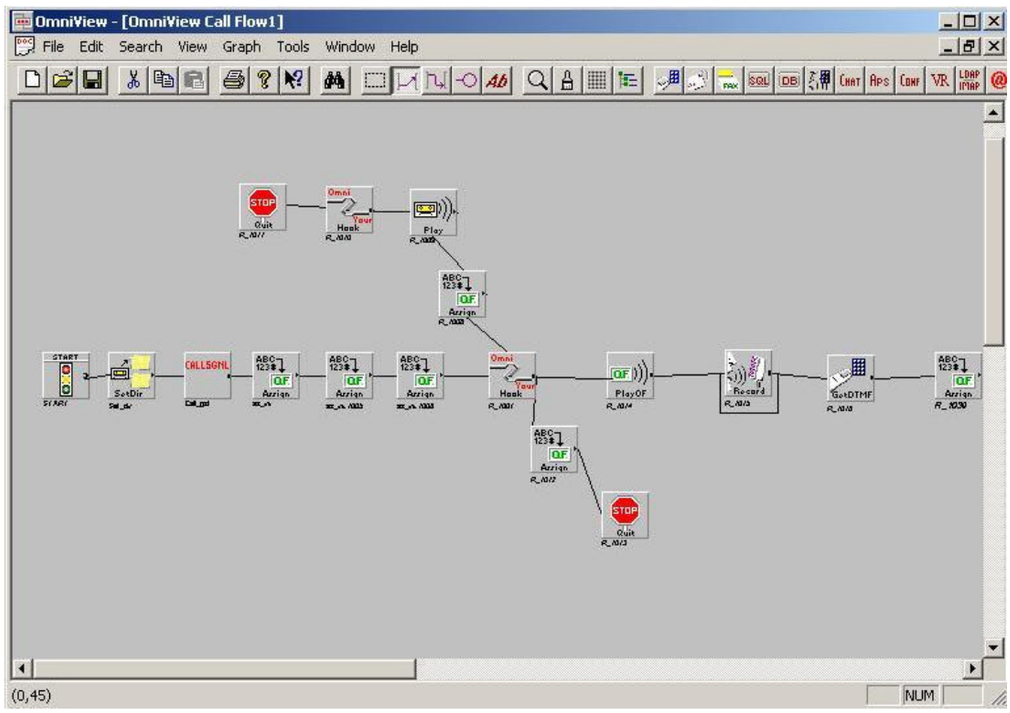


Fig. 6.3.1 Aplicación de Cobro Revertido en Ambiente IVR APEX.

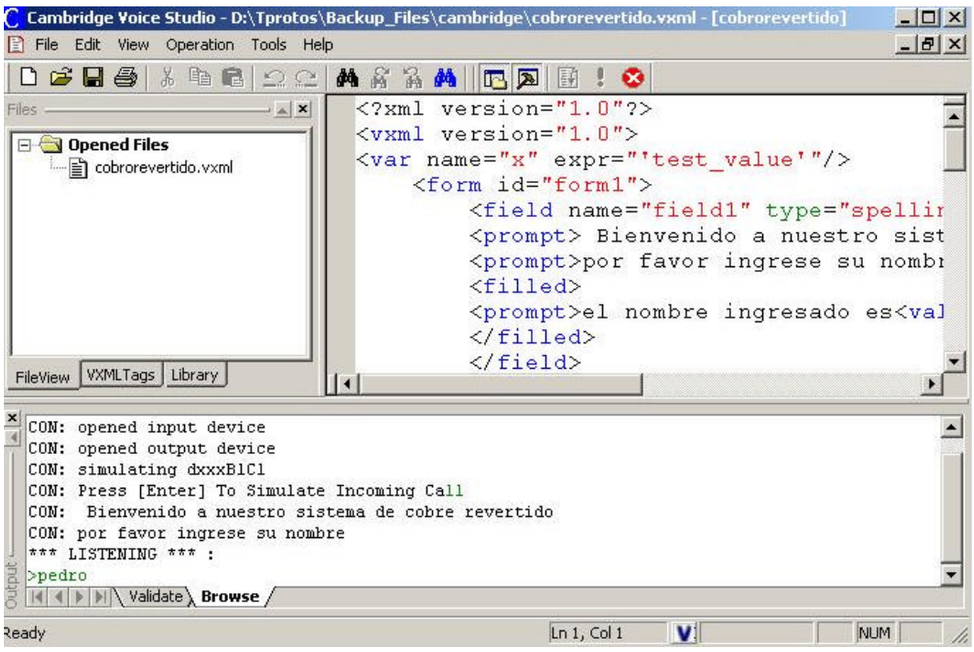


Fig. 6.3.2 Aplicación de Cobro Revertido en Ambiente VoiceXML. (ver código fuente en anexo I3)

6.3.3 Pasos a seguir para migrar a plataforma VOICEXML

A continuación se indica el diagrama de conectividad que debería ser utilizada una aplicación diseñada bajo VoiceXML.

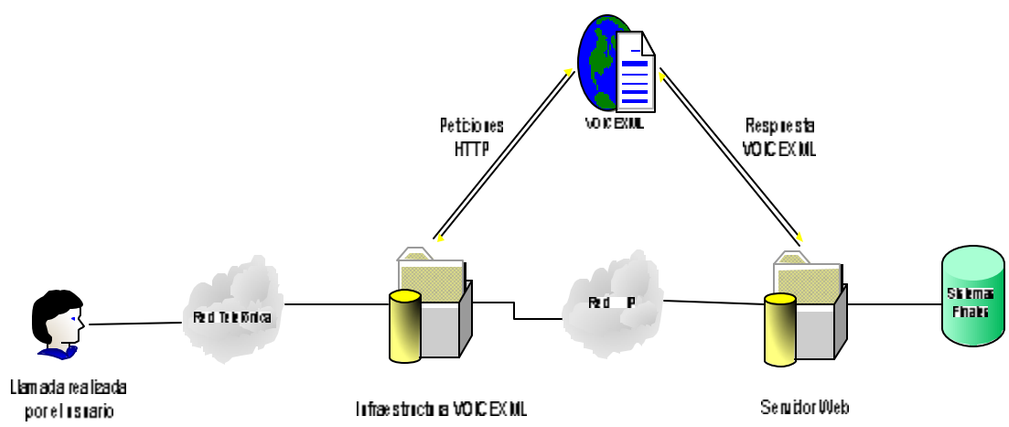


Fig. 6.3.3.1 Diagrama de Conectividad.

La conversación inicia con la llamada entrante a un número específico es atrapada por una aplicación automática de VoiceXML, o una llamada saliente genera

un discado proactivo particular, e inmediatamente se conectan a la aplicación de VoiceXML .Una vez que la llamada es conectada a través de la red telefónica, la infraestructura de VoiceXML actúa como “browser” y comienza a crear series de peticiones http a un Servidor Web tradicional que soporta VoiceXML, Audio, Gramáticas, entregando lo solicitado por la aplicación.

6.3.4 Evaluación.

Las comparaciones básicas a evaluar están en el ámbito de: tiempo de Desarrollo, líneas de código, tiempo de entrenamiento del programador; en la tabla siguiente se concretan estas comparaciones.

Si comparamos una implementación en APEX, el tiempo de entrenamiento para el desarrollo de una aplicación llevaría 60 días aproximadamente, debido a que el consta de dos fases de entrenamiento, una en el ambiente de desarrollo grafico y otra en el de desarrollo de APIS, las cuales son diseñadas en C para el caso de UNIX.

Ahora si comparamos la cantidad de líneas de código el hecho de que se debe diseñar en un ambiente grafico aumenta la cantidad de líneas de código pues se debe programar las APIs necesarias para que el sistema funcione. Aproximadamente esta aplicación tiene 594 líneas.

Posterior a lo antes mencionado después del tiempo de entrenamiento el tiempo de desarrollo requerido para esta aplicación es de 30 días.

En comparación a la aplicación desarrollada en VoiceXML el tiempo de entrenamiento dsminuye a 15 días aproximadamente, debido a que el lenguaje es estándar y de un conocimiento mas cercano a los programadores Web.

La cantidad de líneas de código disminuye considerablemente debido a que el lenguaje le entrega muchas responsabilidades al Gateway para el procesamiento, las líneas de código efectivas escritas para esta aplicación son 51.

Por lo antes mencionado el tiempo de desarrollo en esta plataforma disminuye considerablemente a aproximadamente 15 días.

La alternativa de VoiceXML parte de la base de se necesita la plataforma que soporte la tecnología, para esto necesitamos un Gateway con soporte de VoiceXML, IBM provee un Gateway llamado Websphere Voice Server el cual tiene un costo y adicionalmente otro costo por puerto de IVR, esto hace que la inversión en esta solución pueda parecer alta, pero si se evalúan todos los servicios de valor agregado que serian posibles de implementar se debería tomar en consideración esta alternativa.

7 CONCLUSIONES.

Los sistemas de telefonía proveen una solución eficiente a las necesidades de comunicación. Hoy en día estas necesidades van mucho más allá de las comunicaciones de voz y requieren de servicios cada vez mas especializados, es por ello que se observa el desarrollo de las tecnologías CTI como un factor de gran importancia, especialmente en la generación de nuevos negocios.

La integración de servicios de telefonía y computadores (CTI), hasta hace algunos años ha sido provista en forma exclusiva por empresas empleando soluciones propietarias, que aunque han resuelto muchas necesidades inmediatas, han generado una gran heterogeneidad en los tipos de desarrollo y sistemas implementados.

Es importante notar que luego de investigar y analizar servicios CTI, como sistemas Call Center, sistemas de acceso a información, sistemas de mensajería, etc. se ve la necesidad de contar con estándares, como los desarrollados por organismos como CSTA(conexión entre computadores y sistemas telefónicos), pero aun con estos estándares, las empresas que proveen sistemas de desarrollo comercial implementan sus propios códigos y sistemas de desarrollo, en la mayoría de los casos incompatibles con productos de otras empresas. Es en este punto, en que se hace necesario contar con especificaciones a un nivel más alto que las dadas por el SCTA, las que permitan el desarrollo de sistemas estándares y portables a sistemas provistos por distintos fabricantes.

VoiceXML esta regulado por un estándar de desarrollo normado por la W3C, por bajo el cual todos los programadores diseñan, por lo cual la portabilidad de la aplicaciones es bastante menos engorrosa y barata que con las plataformas comerciales actuales.

Al emplear VoiceXML, no es necesario demasiado conocimiento para comenzar a desarrollar aplicaciones, esta facilidad de desarrollar aplicaciones bajo VoiceXML se debe en gran medida a la idea de emplear las tecnologías para desarrollo en Internet y emplear esto en el desarrollo de sistemas CTI.

En el capitulo 6 se escribieron y probaron algunos ejemplos, luego de analizar la complejidad y tiempo necesario para realizar estos prototipos, se concluye que el desarrollo basado en VoiceXML disminuye los tiempos de aprendizaje e implementación, dado que basta con tener conocimientos de programación en Web para realizar aplicaciones CTI, sin necesidad de un conocimiento acabado en lo relacionado con interfaces de telefonía (protocolos de interconexión, arquitecturas de hardware, etc.).

Con el objeto de evaluar la posibilidad de migrar un sistema real de cobro revertido a una plataforma en VoiceXML, se codifico un servicio muy similar al empleado en forma real, esto se hizo pensando en la posibilidad futura de realizar una exportación de este tipo de servicios a VoiceXML. Luego de implementar el sistema de cobro revertido se llega a la conclusión de que el desarrollo basado en VoiceXML presenta ventajas reales, como: disminución en la codificación, disminución en tiempos de aprendizaje, y además provee una estandarización para desarrollos futuros.

Basados en el sistema de cobro revertido, desarrollado en VoiceXML, y aun cuando el desarrollo es mas simple y presenta las ventajas mencionas anteriormente, se observa que existe un costo inicial al momento de implementar la solución real, el cual es bastante elevado.

Finalmente se concluye que aun cuando VoiceXML presenta ventajas comparativas excepcionales ante sistemas de desarrollo propietarios, aun su implementación pasa por una fuerte inversión económica, la que debe ser sometida a evaluación por parte de las empresas que emplean servicios CTI.

8 BIBLIOGRAFIA.

[Axxon, 2002] Diario On Line, http://axxon.com.ar/c-115InfoMeucciTelefono1.htm

[C.C.I.T.T, 1969] C.C.I.T.T, Especificaciones detalladas del sistema de señalización R.2, Publicado por la Unión Internacional de Telecomunicaciones, 1969.

[Dialogic, 1996] Dialogic Corporation, Carl R. Strathmeyer, “An Introduction To Computer Telephony”, http://www.dialogic.com/company/whitepap/carlieee.htm

[Edgard, 1997] Edgard, Bob. “PC Telephony”. Parity Corporation. USA 1997.

[Elliot, 2001] Elliot B., Simpson K. “Call Center Infrastructure Magic Quadrant 2001”. Gartner. 2001.

[Etsi,1990] Integrated Services Digital Network (ISDN);User-network interface layer 3 Specifications for basic call control, 1990.

[Ivrsoft, 2002] Software for CTI and IVR System Development http://www.ivrsoft.com/

[Martin, 2002] Martin F., Martin L. Álvarez. “Desarrollo de una familia de Aplicaciones CTI (Computer Telephony Integration)”. Universidad de Vigo - Seintel S.L. 28 de Mayo de 2002.

[Martin, 2002] Martin Fernando, Área de Teoría de la señal y Comunicaciones de la Universidad de Vigo. http://wgpi.tsc.uvigo.es/~fmartin/CursoCTI.

[Microsoft, 1996] Microsoft Corporation. "Telephony Application Programming Interface (TAPI) Programmer's Reference". 1996.

[Nmscommunications, 2002] Nmscommunications, http://www.nmscommunications.com/nms/nmssweb.nsf/productlist/QX_2000

[Novell, 1998] Novell Corporation. "Netware Telephony Services Application Programming Interface (TSAPI), Version 2". 1998.

[Nuance, 2002] Nuance While Paper, “Business Case for Voice Authentication”, Nuance Inc.

[Primecti, 2002] Primecti, Developer company, http://www.primecti.com

[Tanembaum, 1991] A. S. Tanembaum. “Redes de Computadores”. Editorial Prentice Hall.

[SS8, 2001] SS8 Network, http://www.ss8.com

[Sharma, 2002] C. Sharma y Kunings, VoiceXML Professional Developers Guide, Editorial Wiley Computer Publishing.

[VIDAL, 2000] Vidal, Luis. “Integración de Sistemas de Telefonía y Computadores (CTI), IX Congreso internacional de Telecomunicaciones, SENACITEL 2000.

[VoiceXMLforum, 2001] VoiceXMLforum,
http://www.voicexmlforum.com/tutorials/intro6.html,
http://www.voicexmlforum.com/tutorials/intro7.html

[W3C, 2001] Voice Extensible Markup Language (VoiceXML) Version 2.0, W3C Working Draft, http://www.w3.org/TR/2001/WD-voicexml20-20011023

ANEXO I CODIGO FUENTE PROTOTIPOS

A continuación se entregan los códigos fuentes de los prototipos diseñados.

I.1 Código Fuente Cuentamatica.

El sistema a continuación señalado fue diseñado de manera de obtener un resultado operacional mínimo, no se considero seguridad, depuración de código, etc, debido a que es te es solo un prototipo.

Deposito.vxml

```
< ?xml version= »1.0 » ?>
<vxml version="1.0" mode="TTS">
<!--            Declaracion de Variables a Usar en Este Ejemplo de VXML    -->
        <var name="newbal" expr="0"/>
        <var name="exist" expr="0"/>
        <var name="amt" expr=""/>
        <var name="amt1" expr=""/>
        <var name="dep" expr=""/>
        <var name="clientname" expr=""/>
        <var name="DBHandle"/>
<!--    Establece una Coneccion de ODBC Para Accesar una Base de Datos
        -->
        <form scope="application">
            <block>
                    <script><![CDATA[
                    var handle=ODBCConnect("sa","","dbtest");
                    DBHandle=handle;
                    if (DBHandle <0)
```

```
                    {
                    echo "Error... no se puede conectar con la DB",endl;
                    return;
                    }
                ]]></script>
            </block>
        </form>
        <form scope="document">
                <field name="fname" type="spelling" scope="document">
                <prompt>Bienvenido al sistema de cuenta en linea.
                por favor ingrese su nombre</prompt>
                 <filled>
                    <script><![CDATA[
                    var cols=0,rows=0,err=0;
                    var i=0;
                    var result=ODBCexecsql(DBHandle,"select Name from
data",*rows,*cols,*err);
                    clientname=result[0].Name;
                    if(rows<0)
                    {
                     echo "No se puede extraer informacionde la DB",endl;
                     return;
                    }
<!--    Revisa en la DB si existe el nombre ingresado dentro de la lista de
clientes. -->
                     for(i=0;i<rows;i++)
                     if(result[i].Name==fname)
                    {
                     exist=1;
```

```
                    }
                    ]]></script>
<!--        Llega al cliente existente, de otro modo crea al  cliente -->
            <if cond="exist==1">
                <goto next="#currentamt"/>
            <else/>
                <script><![CDATA[
                    var cols=0,rows=0,err=0;
<!--Ingresa nombre de usuario en la DB y le asigna $0 como monto -->
<!-- de apertura de cuenta.-->
            var result1=
            ODBCexecsql(DBHandle,"Insert into data
(Name,balance_actual,temp)
                    values('"+fname+"','0','0')",*rows,*cols,*err);
                    ]]></script>
              <goto next="#currentamt"/>
              </if>
            </filled>
            </field>
            <block name="currentamt">
                    <script><![CDATA[
                        var cols=0,rows=0,err=0;
<!--        Entrega el balance actual del usuario     -->
            var result=ODBCexecsql(DBHandle,"select balance_actual from
data
                        where name='"+fname+"'",*rows,*cols,*err);
                        if(rows<0)
                        {
                         echo "no se puede extraer informacion de la
```

```
tabla",endl;
                                    return;
                                    }
                                    amt=result[0].balance_actual;
                                    amt1="$"+amt;
                                    ]]></script>
                            <prompt>su balance actual es<value
expr="amt1"/></prompt>
                            <goto next="#deposit"/>
                    </block>
                    <field name="deposit" type="boolean" scope="document">
                    <prompt>Desea hacer un deposito?<value
expr="fname"/></prompt>
                            <filled>
                                    <if cond="deposit=='yes' || deposit=='1'">
                                    <goto next="#depamt"/>
                                    <else/>
                                    <prompt>gracias<break msecs="1000"/></prompt>
                                    <goto next="#end"/>
                                    </if>
                            </filled>
                    </field>
                    <field name="depamt" type="spelling" scope="document">
                            <prompt>por favor ingrese el monto de su
deposito</prompt>
                            <filled>
                                    <script><![CDATA[
                                            var cols=0,rows=0,err=0;
<!-- Actualiza el monto de del usuario con lo ingresado por el mas lo contenido
```

```
en la cuenta.-->
        var result=ODBCexecsql(DBHandle,"Update data set deposit='"+depamt+
                                "'where name='"+fname+"'",*rows,*cols,*err);
<!--            Entrega informacion del deposito para calcular el nuevo monto
        -->
var result1=ODBCexecsql(DBHandle,"Select deposit from data where
name='"+fname+"'",*rows,*cols,*err);
                                dep=result1[0].deposit;
                                newbal=(val(amt))+(val(dep));
                                newbal="$"+str(newbal);
<!--            Actualiza el actual balance con el nuevo deposito.-->
                                var result2=ODBCexecsql(DBHandle,"Update
data set balance_actual='"+str(newbal)+
                                "'where name='"+fname+"'",*rows,*cols,*err);
<!--            Commits Update  -->
                                ODBCcommit(DBHandle);
                                ]]></script>
                        <prompt>su nuevo balance es<value
expr="newbal"/>
                        Gracias por usar nuestro sistema Adios.</prompt>
                        <goto nextitem="end"/>
                    </filled>
                </field>
<!--            Termina el acceso a la DB y sale de la Applicacion  -->
                <block name="end">
                 <script><![CDATA[ODBCDisconnect(DBHandle);]]></script>
                 <exit/>
                </block>
                </form>
</vxml>
```

I.2 Codigo Fuente Reserva.

Este sistema realiza reservas de una salón de conferencias y notifica si así se desea a los invitado a esta mediante correo electrónico; el paso siguiente en para poner en marcha esta aplicación prestando servicios reales seria almacenar la información en una base datos, para almacenar dichas reservas, esto queda para desarrollos futuros de quien desee investigarlo.

Dataroot.vxml

```
<?xml version="1.0"?>
<vxml version="1.0">

<property name="audio.path" value="wav\"/>
<var name="exist" expr="0"/>
<var name="badInputCounter" expr="0"/>
<nomatch>
        <script><![CDATA[
                badInputCounter=badInputCounter+1;
                echo "nomatch counter: ",badInputCounter,endl;
        ]]></script>
No le he entendido.
        <if cond="badInputCounter==3">
                <assign name="exist" expr="1"/>
                <goto nextitem="end"/>
        </if>
</nomatch>

<noinput>
        <script><![CDATA[
```

```
                badInputCounter=badInputCounter+1;
                echo "noinput counter: ",badInputCounter,endl;
        ]]></script>

        <if cond="badInputCounter==3">
                <assign name="exist" expr="1"/>
                <goto nextitem="end"/>
        </if>
        No le he entendido.
        </noinput>
        <form id="month" scope="document">
                <block>
                        Bienvenido al sistema de reserva de salones de
conferencias.
                        Si no esta seguro de los formatos de cada ingreso diga,help
o
                        ingrese las solicitudes del sistema.
                </block>
      <field name="calendar" type="date" scope="document">
        <grammar> help </grammar>               <!--in-line grammar for help-->
                        <prompt>
                                Para que fecha desea realizar la reserva del salon de
conferencias?
                        </prompt>
        <help>
                El formato para date is yyyymdd.
        </help>
        <filled>
                <prompt>Usted ha ingresado <value expr="calendar"
```

```
class="date"/></prompt>
                <goto nextitem="time"/>
                </filled>
        </field>
        <field name="time" type="time" scope="document">
        <prompt>Para que hora desea realizar la reserva del salon de
conferencias?</prompt>
        <help>
                El formato para time es hhssx.
        </help>
        <filled>
          <prompt>Usted ha ingresad<value expr="time"
class="time"/></prompt>
          <goto nextitem="end"/>
        </filled>
      </field>

                <block name="end">
                        <if cond="exist==1">
                                <prompt>Adios.</prompt>
                                <exit/>
                        <else/>
                                <prompt>Usted a realizado una reserva para
                                <value expr="calendar" class="date"/>
                                        at<value expr="time " class="time"/>
                                </prompt>
                                <goto next="data.vxml"/>
                        </if>
                </block>
        </form>
</vxml>
```

Data.vxml

```
<?xml version="1.0"?>
<vxml version="1.0" application="dataroot.vxml">
<var name="exist" expr="0"/>
<var name="calendar" expr="session.month.calendar.value"/>
<var name="time" expr="session.month.time.value"/>
<var name="badInputCounter" expr="0"/>
<nomatch>
        <script><![CDATA[
                badInputCounter=badInputCounter+1;
                echo "nomatch counter: ",badInputCounter,endl;
        ]]></script>

 No le entiendo?.
        <if cond="badInputCounter==3">
                <goto nextitem="done"/>
        </if>
</nomatch>

<noinput>
        <script><![CDATA[
                badInputCounter=badInputCounter+1;
                echo "noinput counter: ",badInputCounter,endl;
        ]]></script>

        <if cond="badInputCounter==3">
                <goto nextitem="done"/>
        </if>
        No le he entendido.
```

```
</noinput>


<script><![CDATA[


struct emp_info  {name_,email_address};


var attendees = array(0);
var att_count = 0;
var i=0;
var list1="";
function add_employee(raw_result)
{
   echo "Enter Function",endl;
   var comma = strFind(raw_result,",");
   var s_name_ = strSubstring(raw_result,0,comma);
   var i,att_len;
   emp_info data;


        var                                s_email_address                                =
strSubstring(raw_result,comma+1,LEN(raw_result));


   att_len=len(attendees);
   for (i=0;i<att_len;i++)
   {
      echo "Comparing ",attendees[i].email_address," to ",s_email_address,endl;
      if (attendees[i].email_address==s_email_address)
         return -1;
   }
   data.email_address=s_email_address;
```

```
    data.name_=s_name_;


          aadd(attendees,data);
          att_count++;
          return 0;
}
function convertdate(date_str,retval)
{

retval=strSubstring(date_str,4,2)+"/"+strSubstring(date_str,6,2)+"/"+strSubstring(
date_str,0,4);
  echo date_str,":",retval,endl;
  return 0;
}
function converttime(time_str,retval)
{
  retval=strSubstring(time_str,0,2)+":"+strSubstring(time_str,2,2);
  echo time_str,":",retval,endl;
  if (strSubstring(retval,4,1)=="a")
     retval=retval+"AM";


  else
     retval=retval+"PM";
  return 0;
}
]]></script>


          <form id="get_employees" scope="document">
          <!--
```

```
                Este bloque convierte el formato de fecha y hora ingresado por el
usuario,
                para utillizarlo en el cuerpo del correo.
        -->
                <block name="convdate">
                        <script><![CDATA[
                                var conv_date="";
                var datestr=calendar;
                var conv_datestr="";
                var conv_time="";
            var timestr=time;
            var conv_timestr="";


            convertdate(datestr,*conv_datestr);
            calendar=conv_datestr;
                echo "CALENDAR=["+calendar+"]",endl;


            converttime(timestr,*conv_timestr);
                time=conv_timestr;
            echo "TIME=["+time+"]",endl;
                        ]]></script>
                </block>


                <field name="add_who">
        <grammar>
                                Luis{Luis Vidal,lvidal@inf.uach.cl}|
                                Rodolfo{Rodolfo rosas,rrosas@inf.uach.cl}|
                                cristian{Cristian Alarcon,calarcon@telsur.net}|
                                rrosas{Rodolfo Rosas,rrosas@telsur.net}|
```

```
    </grammar>
<prompt> A quien le gustaria invitar a la conferencia?</prompt>
<help>
        Por favor ingrese el primer nombre de la persona que desea invitar
      (Luis,Rodolfo,Cristian or rrosas).
</help>
<filled>
        <script><![CDATA[
        if (add_employee(add_who)==-1)
        {
          echo "Ya ha sido ingresado",endl;
          exist=1;
        }
        else
        {
                exist=0;
        }
        var j=0;

       for (j=0;j<att_count; j++)
        {
        add_who = attendees[j].name_;
      }
        echo"[-- ",add_who," --]",endl;
                        ]]></script>
   <if cond="exist==1">
      <prompt>Usted ya ha ingresado ese nombre.</prompt>
        <clear namelist="employee" />
        <goto nextitem="employee"/>
```

```
            <else/>
            <assign name="badInputCounter" expr="0"/>
            <prompt>Usted a dicho <value expr="add_who"/></prompt>
            <clear namelist="employee" />
                  <goto nextitem="employee"/>
            </if>
    </filled>
            </field>

            <field name="employee">
    <grammar> yes | no </grammar>
    <prompt>Desearia invitar a otra persona?</prompt>
    <filled>
      <if cond="employee=='yes'">
            <assign name="badInputCounter" expr="0"/>
        <clear namelist="add_who" />
        <goto nextitem="add_who" />
      <else/>
            <goto nextitem="conf"/>
      </if>
    </filled>
  </field>

            <field name="conf" type="boolean">
    <grammar> help </grammar>
    <prompt>
            Le gustaria enviar un email a todos los invitados a la conferencia?
    </prompt>
    <help><reprompt/></help>
```

```
        <filled>
                <assign name="badInputCounter" expr="0"/>
           <if cond="conf=='yes'">
              <goto next="#fin"/>
           <else/>
              <goto next="#done"/>
           </if>
        </filled>
        </field>
        </form>

        <form id="fin" scope="document">
        <block name="email_">
           <script><![CDATA[
           var ip="smtp.entelchile.net";
                var frm="irreal@entelchile.net";
           var subject="Invitación a la Conferencia";
                                var body="Usted ha sido invitado a la conferencia
para
                                "+calendar+' at'+' '+time+' ';
                                var body1="en el salon rojo de conferencias.";
                                var l=0;
                                var attend_cnt=len(attendees);

           for (l=0;l<attend_cnt; l++)
           {
                echo attendees[l].name_,":",attendees[l].email_address,endl;
              var rc=emailSend(ip,attendees[l].email_address,
                frm,subject,body+body1);
```

```
            echo rc,endl;
          }
        ]]></script>
        <goto nextitem="done"/>
        </block>

        <block name="done">
        <prompt> Gracias por utilizar nuestro sistema de reserva de salones de conferencias.      </prompt>
        <exit/>
        </block>
        </form>
</vxml>
```

I.3 Código Fuente Cobrorevertido (VoiceXML).

Esta aplicación es simple y se implementó con lo posible de ser probado en el ambiente de desarrollo local, debido a que la prueba de esta aplicación en un servidor real no estuvo al alcance de los recursos disponibles para esta tesis por quien les habla.

Cobrorevertido.vxml

```
<?xml version="1.0"?>
<vxml version="1.0">
<var name="x" expr="'test_value'"/>
        <form id="form1">
                <field name="field1" type="spelling">
                <prompt> Bienvenido a nuestro sistema de cobro
```

```
revertido</prompt>
                <prompt>por favor ingrese su nombre</prompt>
                <filled>
                <prompt>el nombre ingresado es<value expr="field1"/></prompt>
                </filled>
                </field>
        </form>
        <form id="form2">
                <field name="field2" type="spelling">
                <prompt>ingrese el numero al que desea llamar</prompt>
                <filled>
                <prompt>usted esta llamando al<value expr="field2"/></prompt>
                </filled>
                </field>
        </form>
        <form id="test">
         <block>
                <prompt>Espere un momento mientras se realiza su
llamada</prompt>
         </block>
        <!-- hasta aca se realiza recien el ingreso del nombre y el numero a
marcar-->
        <field name="answer" type="boolean">
                <prompt>Le estan llamando por cobro revertido</prompt>
                <prompt>Desea aceptar la llamada de diga yes o no</prompt>
                <help>diga yes o no</help>
        <filled>
        <if cond="answer=='no'" >
                <prompt> Gracias por utilizar nuestro sistema adios.</prompt>
```

```
                </if>
        </filled>
                </field>
                <transfer name="mycall" dest="field2" connecttimeout="30s"
bridge="true">
                <filled>
                <!--<assign name="mydur" expr="mycall$.duration"/>-->
                <if cond="mycall == 'noanswer'">
                <prompt>Lo siento el numero al que usted llama no
contesta.</prompt>
                <elseif cond="mycall == 'busy'"/>
                <prompt>Lo siento el numero a que llama se encuentra
ocupado.</prompt>
                <elseif cond="mycall == 'completed'"/>
                <prompt>su llamada ha sido aceptada, por favor hable.</prompt>
                <else/>
                <prompt>Lo siento, lo se ha podido concretar su llamada,por favor
intentelo mas tarde.</prompt>
                </if>
        </filled>
        </transfer>
</form>
</vxml>
```