ESCUELA SUPERIOR POLITECNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

INFORME DE MATERIA DE GRADUACION

Implementación de un IVR (Interactive Voice Response) utilizando un VoiceXML Browser

Previa a la obtención del Título de:

INGENIERO EN ELECTRONICA Y TELECOMUNICACIONES

Presentada por:

EFREN LENIN GOMEZ CAMPOVERDE

GIOVANNY OCTAVIO IZA GALLEGOS

GUAYAQUIL – ECUADOR

AÑO

2010

AGRADECIMIENTO

A Dios nuestro señor, sin el cual ninguna de nuestras metas sería posible, gracias por darnos la fuerza para seguir adelante.

A nuestra familia, por el apoyo que siempre nos brindaron, especialmente a nuestros padres porque todo lo que hemos logramos es fruto de su esfuerzo, cariño y apoyo incondicional. Porque siempre nos aconsejaron y nos guiaron por el camino de las buenas costumbres.

A nuestros amigos, porque siempre estuvieron cuando los necesitamos, porque demostraron que la verdadera amistad no pone condiciones.

DEDICATORIA

Esta obra va dedicada a Dios y a mis padres; A mi Dios, por darme la vida y las facultades para alcanzar esta meta.

A ti madre, porque con sacrificio me ayudaste a cumplir esta etapa de mi vida.

A ti padre, porque siempre estuviste conmigo, dejaste este mundo terrenal y te convertiste en mi ángel guardián, siempre te recordaré.

Efrén Gómez Campoverde.

A Dios, por Su infinita bondad y por las bendiciones que ha dado a mi vida. A mi familia por su apoyo incondicional y su esfuerzo por hacer de mi una mejor persona. A mis profesores por compartir su conocimiento y haber sido guía en todo este camino. A mis amigos que estuvieron acompañándome y brindándome su ayuda durante tanto tiempo.

Giovanny Iza Gallegos.

DECLARACION EXPRESA

“La responsabilidad del contenido de este Trabajo de Grado, nos corresponde exclusivamente; y el patrimonio intelectual de la misma a la Escuela Superior Politécnica del Litoral”.

(Reglamento de Graduación de la ESPOL)

_____

Efrén Lenín Gómez Campoverde

_____

Giovanny Octavio Iza Gallegos

TRIBUNAL DE SUSTENTACION

Ing. Gabriel Astudillo Brocel

PROFESOR DE LA MATERIA DE GRADUACIÓN

Ing. Patricia Chávez Burbano

PROFESOR DELEGADO POR EL DECANO DE LA FACULTAD

RESUMEN

Este proyecto se basó en la implementación de una central telefónica en la que se implementó un Sistema de Respuesta de Voz Interactiva (IVR) con VoiceXML Browser. Esta plataforma fue diseñada para recibir llamadas cuando el usuario digitara la extensión asignada por la central telefónica Asterisk al IVR, dicha central es la encargada de guiar esta llamada, según lo configurado en su plan de marcado, por los canales AGI hacia la plataforma VoiceXML, Esta plataforma cumplía con las funciones de solicitar los documentos vxml al servidor web y ejecutar la programación de los documentos vxml. La programación de los documentos fue orientada para brindar conexión al usuario con una página web cuyo formato predeterminado para trabajar en esta tesis fue RSS. De este modo el usuario podía tener información actualizada (debido a las ventajas que brinda RSS) de noticias a nivel mundial con solo levantar su teléfono y marcar la extensión del IVR con VoiceXML Browser.

INDICE GENERAL

INDICE DE FIGURAS

INDICE DE TABLAS

INTRODUCCION

La gran demanda de navegación por internet que en los actuales momentos se presenta a nivel mundial, la búsqueda de nuevas tecnologías y la creciente innovación de aplicaciones web, han hecho que gigantes industrias tecnológicas como lo son AT&T, IBM, Lucent, Motorola, entre otras, desarrollen su propio lenguaje extendido basado en voz (VoiceXML).

A pesar de que en sus inicios el desarrollo de VoiceXML era estrictamente comercial, pronto se integraron asociaciones y universidades para contribuir con el desarrollo del lenguaje, esto rápidamente dio resultados positivos.

En la actualidad, VoiceXML permite que los desarrolladores puedan hacer mejoras significativas a sus antiguos IVR's sin necesidad de costosas y caras implementaciones, permitiéndole al usuario el control de la conversación, mediante la implementación de diálogos y sub-diálogos.

Esta tecnología se convertirá en un futuro muy cercano en la forma de optimizar el tiempo en el sector empresarial, dado que reducirá la necesidad de realizar tareas cotidianas de una forma convencional que consume tiempo, por ejemplo una persona cuyo desempeño laboral está sujeto a la constante revisión de la economía actual deberá emplear un tiempo determinado encendiendo su computador, empezando a navegar y buscando la información que requiere en la web, que es generalmente la mejor forma de obtener información a nivel mundial, dicho tiempo

se podría emplear en realizar la misma actividad junto a cualquier otra debido a la facilidad que brinda la tecnología VoiceXML.

CAPITULO 1

ANTECEDENTES Y JUSTIFICACION

1.1 ANTECEDENTES

La constante innovación de servicios y aplicaciones web se ha vuelto más que un lujo, una necesidad, esto debido al constante aumento de la penetración de la internet. Estos factores junto con el anhelado desarrollo tecnológico, han puesto a las compañías en busca de la mejor forma de facilitar al usuario la interacción con las páginas web.

Con el desarrollo de las aplicaciones Código Abierto, específicamente Asterisk en nuestro caso, la telefonía logró un gran avance debido a que la implementación de estos sistemas era tradicionalmente costosa, complicada de llevar a cabo y además carecían de facilidad al momento de implementar una solución basada en software. Con Código Abierto junto con la tecnología VoIP esto se facilitó ya que la implementación se podía hacer sobre las redes ya existentes y además se facilitaba el despliegue de soluciones pues se redujo el costo de implementación en Software a cero. Esta convergencia dio cabida a un amplio número de aplicaciones, mostró efectividad, pero pronto esto volvió a generar precios altos debido a que muchas de sus aplicaciones se volvieron comerciales y muy rígidas, impidiendo así la rápida implantación de servicios y no facilitando los cambios en las redes.

Los métodos de navegación web que actualmente se han desarrollado no son de fácil uso para las personas con discapacidades visuales además ciertos modelos de navegación para dichas personas están todavía en investigación y desarrollo por diferentes universidades, sin embargo con

Asterisk y el enlace con VoiceXML Browser se puede desarrollar de una forma menos costosa y con un tiempo de implementación menor un método para dichas personas, lo cual permitirá una adaptación más rápida, una fácil navegación y compresión de la web.

1.2 JUSTIFICACION

En la actualidad las comunicaciones han evolucionado en forma rápida lo cual ha derivado en la necesidad de disminuir costos y sobre todo el hecho de poseer redes convergentes sobre las cuales se puedan transmitir además de la voz diferentes tipos de datos y servicios, esto originó un mayor interés por el desarrollo en conjunto de diferentes aplicaciones. Bajo este mismo concepto de convergencia, la búsqueda de la vinculación del usuario con las grandes redes mundiales como Internet se ha vuelto un objetivo cada vez más importante y el uso del lenguaje XML en cual se halla VoiceXML, se ha vuelto el sistema base para este entorno de concurrencia.

VoiceXML facilita la accesibilidad a un mayor número de usuarios mientras brinda mayor interacción con Internet mediante la implementación de servidores que soportan las páginas web escritas en lenguaje vxml (VoiceXML), y estas a su vez son interpretados por el VoiceXML Browser permitiendo de este modo la conexión de un usuario de la PBX Asterisk con cualquier página web escrita en lenguaje extendido (xml).

1.3 DESCRIPCION DEL PROYECTO

Para la ejecución de este proyecto se han planteado los siguientes objetivos a cumplir.

1.3.1 Objetivos Generales

Implementar un Sistema de Respuesta de Voz Interactiva (IVR) con un VoiceXML Browser que le permita a un usuario de una IP-PBX Asterisk con capacidades especiales o a empresarios que deseen optimizar su tiempo, poder acceder a servicios en internet bajo el formato RSS.

1.3.2 Objetivos Específicos

- Configurar un servidor web de documentos, que servirá como host de las paginas vxml.
- Implementar una central Asterisk.
- Implementar un interprete VXML que soporte Asterisk, así como un servidor Text-to-Speech, que de ahora en adelante lo llamaremos servidor TTS.
- Definir un apropiado plan de marcado y de configuraciones de la central Asterisk.
- Realizar un método para la obtención de información dentro de archivos web con formato RSS.

La implementación del IVR con el VoiceXML Browser consiste primero en levantar un servidor web que será el que alojará los

documentos vxml y responderá a las peticiones que haga el Interprete VoiceXML, todo esto conforme lo muestra el esquema de la figura 1.1. Donde la Plataforma de Implementación será la central Asterisk.

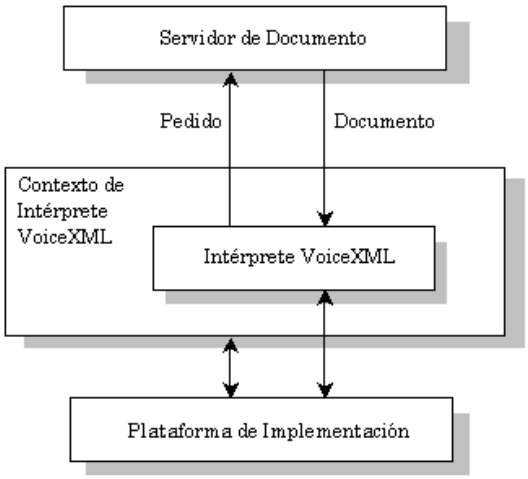

Fig. 1.1 Arquitectura VoiceXML

Una vez que se ha instalado y configurado el servidor Asterisk, definimos que el Intérprete VXML a usar será VoiceGlue, cuya arquitectura es la que se muestra en la figura 1.2

Por medio de la configuración del archivo “extensions.conf”, y haciendo uso del Asterisk Gateway Interface (AGI), podemos orientar las llamadas de los usuarios hacia un servidor web, que si bien puede estar dentro de la misma red, no es necesario que esto se cumpla, por el contrario en este proyecto se utiliza conexiones con páginas web internacionales.

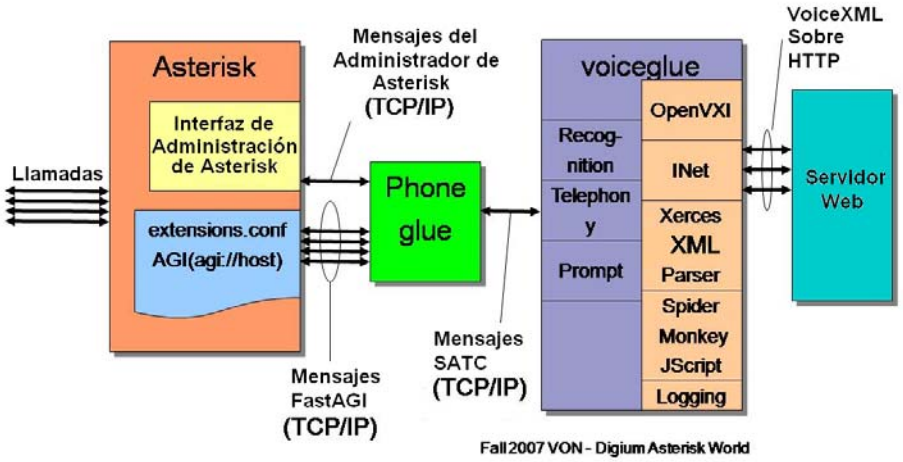

Fig. 1.2 Arquitectura de VoiceGlue

Una manera superficial de saber lo que ocurre es notando que las llamadas que entran a la central Asterisk serán encaminadas hacia el Intérprete VoiceXML según lo indique el plan de marcado, este intérprete a su vez solicita la página vxml, en la cual estará escrita la programación del IVR, al servidor de documento. Luego de que el Intérprete VXML obtenga la información de dicha página empieza la interacción del usuario con la web.

Nuestro Intérprete VXML a emplear será VoiceGlue, el cual ha sido elegido debido a que presenta una mejor resolución de dependencias dentro del Código Abierto y además es un proyecto en conjunto que actualmente no posee costo. VoiceGlue es uno de los pocos proyectos no comerciales que actualmente está arrojando resultados positivos para la comunidad desarrolladora de software.

VOiceGlue posee una gran cantidad de características como lo son:

- Búsqueda completa de documentos HTTP, análisis, y soporte de sintaxis.
- Procesamiento de javascript (ECMAscript).
- Transmisión de parámetros de Asterisk
- Soporte de Cookies
- Búsqueda y reproducción de audio pre-grabado
- Salida TTS (excepto lenguaje de marcado SSML)
- Grabación y recuperación de audio
- Entrada DTMF con gramática SRGS XML
- Soporte de transferencia ciega primitiva
- Etiqueta de datos VXML 2.1

Las características propias del lenguaje VXML que VoiceGlue no soporta son:

- ASR (la entrada del usuario es únicamente DTMF)
- SSML completo soportado en TTS
- Gramática SRGS XML DTMF general completa

1.4 METODOLOGIA

Nuestro sistema es implementado en una distribución de Linux, llamada Ubuntu cuyo kernel es el 8.4.16. Además se implementará un servidor web

Apache, un servidor TTS y se configurará una central telefónica Asterisk para que soporte VoiceGlue. Posterior a esto se realizará la programación del IVR como una página web en lenguaje VoiveXML.

1.5 PERFIL DE LA TESIS

Esta tesis busca introducir el lenguaje VoiceXML como una herramienta poderosa para la interacción hombre-maquina, así como fomentar el desarrollo de las aplicaciones cuyo objetivo sea la convergencia de las diferentes tecnologías, en este caso la telefonía y la navegación web.

El capítulo 2, abordará temas teóricos que nos permitirán interpretar el alcance y las limitaciones de este proyecto.

El capítulo 3, servirá para detallar aspectos técnicos de la implementación del proyecto.

Finalmente en el capítulo 4, se realizarán pruebas para determinar la efectividad y calidad del IVR con VoiceXML Browser.

CAPITULO 2

SISTEMA DE RESPUESTA DE VOZ

INTERACTIVA CON VOICEGLUE EN ASTERISK

2.1 SISTEMA DE RESPUESTA DE VOZ INTERACTIVA TRADICIONAL

Este tipo de sistema se conoce comercialmente como IVR cuyas siglas corresponden a las palabras en inglés Interactive Voice Response. Otro término con el que se conoce es VRU que significa Voice Response Unit. Un IVR es una plataforma muy poderosa que permite el desarrollo de aplicaciones telefónicas, así como brinda facilidades para el diseño, integración, implementación y administración de sistemas telefónicos.

En el mercado de la telefonía IP existen muchas compañías realizando aplicaciones y fabricando plataformas IVR, es de este modo que la comercialización de este tipo de servicio ha hecho posible que hoy en día se tengan IVR's tan poderosos como fáciles de administrar y configurar. Compañías ofrecen IVR's con un lenguaje gráfico amigable, que vienen preparados para el manejo de voz, fax, acceso y escritura a bases de datos, reconocimientos de voz, texto a voz, aplicaciones CTI y que soportan T1s, E1s, ISDN, SS7, VoIP, conferencias y un sin número de característica que demuestran que un IVR es una potente herramienta empresarial. En la figura 2.1 se puede observar la forma en que el usuario accede al IVR a través de la Red de Telefonía Pública [8].

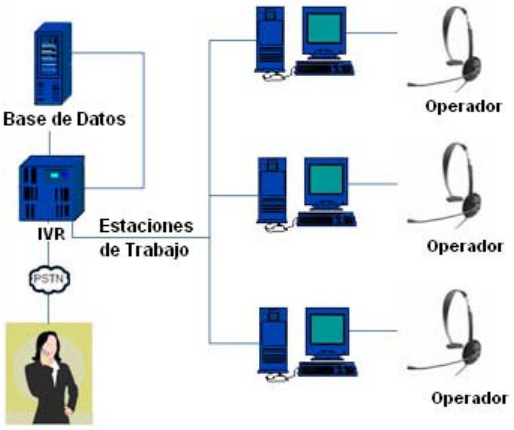

Fig. 2.1 Sistema de Respuesta de Voz Interactiva

Un IVR es comúnmente implementado en empresas que suelen recibir grandes cantidades de llamadas, esto con el fin de reducir la necesidad de personal y los costos que estos le representan a dicha entidad. Básicamente la tecnología IVR se usa para guiar las llamadas hacia el personal que pueda atender las peticiones del usuario que realizó la llamada, optimizando de este modo el servicio.

La forma de proceder del IVR típico es que cuando el usuario realiza una llamada hacia el número de una empresa con dicho servicio, el sistema IVR contesta la llamada y le muestra al usuario una serie de opciones mediante audio pregrabado en archivos. Una vez que el usuario ha elegido una de las opciones, ya sea presionando una tecla de su teléfono (DTMF) o diciendo alguna palabra o frase (ASR), empieza a navegar por el menú hasta encontrar la información precisa que esta solicitando, una vez hecho esto el IVR enruta la llamada hacia el destinatario final.

2.2 SISTEMA DE RESPUESTA DE VOZ INTERACTIVA DE ASTERISK

Un IVR dentro de Asterisk nació como una necesidad de poder realizar lo que las centrales tradicionales podían hacer pero a un menor costo.

Primero se debe tener en cuenta que la configuración de Asterisk como tal no es suficiente para que funcione un IVR, se necesita de un Plan de Marcado que es la agrupación de extensiones dentro de un conjunto de usuarios que comparten las mismas características (también llamado contexto dentro de Asterisk) o en pocas palabras es un enrutador de llamadas. Cada extensión posee un número de prioridades y dentro de estas prioridades una aplicación es llamada. La forma de encaminar las llamadas dentro de un Dialplan es utilizando la palabra reservada “exten”, en otras palabras **exten** le dirá a Asterisk que un usuario que marcó una extensión con un determinado **id** ejecutará comandos según el orden de prioridad que estos posean.

Toda esta configuración del Dialplan se la realiza en el archivo extensions.conf ubicado en la carpeta /etc/asterisk y posee la siguiente sintaxis:

```
[context]
exten => id, priority, command
```

En Asterisk un IVR es interpretado como una extensión más, la diferencia con otras extensiones radica en el uso de diferentes comandos para de esa forma brindar una interacción con el usuario que digitó dicha extensión, lo cual tiene mayor aplicación a usuarios que no estén dentro de nuestra red y requieran información específica y objetiva del abonado de la red Asterisk al que ellos quieren contactar, esto es solo un ejemplo del alcance de un IVR. La realidad es que los IVR´s se ajustan a las necesidades y requerimientos dentro de la empresa que los implementa.

Los alcances de un IVR en Asterisk están enmarcados por las siguientes características:

- Capacidad de trabajar con casi todos los estándares de telefonía tradicional.
- Soporta una gran variedad de protocolos entre los que resaltan: IAX2, SIP, SS7, MGCP, H.323.
- Soporte de todos los codecs estándar: G.711, G.723.1, GSM, ILBC, ADPCM, etc.
- Permite “bridging” entre tecnologías distintas, que es un método mediante el cual se pueden interconectar dos elementos de diferentes fabricantes utilizando dispositivos intermedios entre ellos.

Pero se ve limitada por la capacidad en el hardware y software del servidor sobre el cual esté trabajando Asterisk.

La aplicación más común dentro de empresas que usan Asterisk para un IVR es en las llamadas de un usuario de la red tradicional de telefonía pública y que desea acceder a nuestra red, para poder realizar esta conexión se necesita de unas interfaces.

Dentro de Asterisk la representación de los puertos y su ubicación sería mediante la figura 2.2 [12]

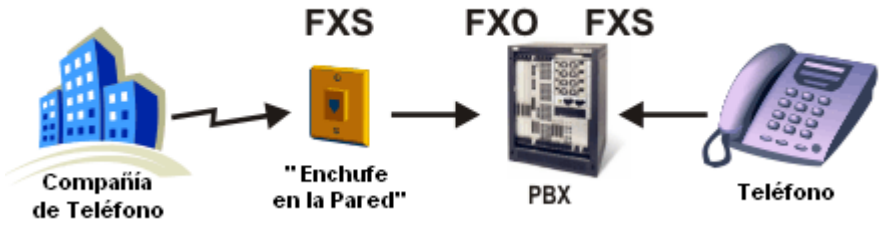

Fig. 2.2 FXS/FXO

FXS (Estación de intercambio externo; Foreign Exchange Station en Inglés) y FXO (Oficina de intercambio externo; Foreign Exchange Office en Inglés) son los nombres de las interfaces usados en las líneas telefónicas analógicas del servicio de telefonía pública.

FXS – Interfaz del abonado, es el puerto que lleva la línea analógica hacia el abonado y alimenta el terminal. En otras palabras, es el “enchufe en la pared” que envía el tono de marcado, corriente para el terminal telefónico y tensión al timbre de llamada. Un FXS visto desde el lado del usuario o abonado "parece ser" una central telefónica. Por tanto, suministra energía al

terminal, envía señales de llamada, detecta si el teléfono está descolgado y comunica voz.

FXO – Interfaz de central, es el puerto del terminal que recibe la línea analógica. Es el enchufe del teléfono o fax, o el enlace de su centralita telefónica analógica. Envía hacia la red una indicación de colgado/descolgado (cierre de bucle). Como el puerto FXO forma parte de un dispositivo, tal como un fax o teléfono, a menudo se denomina al dispositivo como “dispositivo FXO”. Visto desde la central telefónica "parece ser" un teléfono normal, que acepta señales de llamada (corriente alterna de unos 50V para que suene el timbre), se puede colgar y descolgar y recibe señales vocales. Puede descolgarse de dos maneras: 1) abriendo el bucle (loop start) y 2) poniendolo a tierra (ground start).

FXO y FXS van siempre en pares, es decir, son similares a un enchufe macho/hembra.

Finalmente una central Asterisk en la que se haya implementado un IVR lucirá como la figura Fig 2.3 [10].

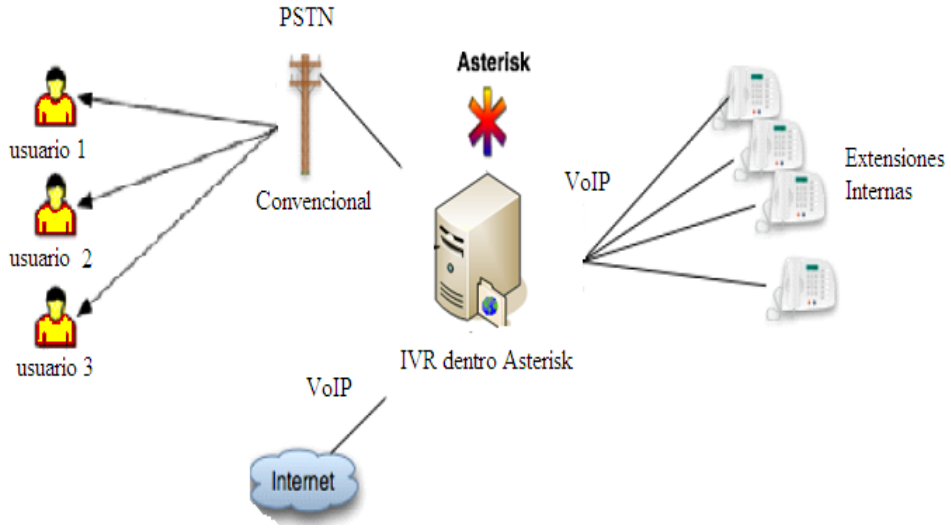

Fig. 2.3 Central Asterisk

2.3 IVR CON VOICEXML BROWSER

En la actualidad ha dejado de ser suficiente que una empresa preste solo algún tipo de servicio utilizando un portal web. En contraste muchas empresas cuyo objetivo es desarrollar estándares están trabajando en sistemas en donde el usuario tenga determinados servicios de información con solo utilizar un teléfono, ya sea que este posea una línea fija o sea un celular. Es así que actualmente VoiceXML se ha convertido en uno de los lenguajes con mayor aceptación por el mercado, debido a sus múltiples ventajas y facilidad de integración con otros sistemas.

2.3.1 VoiceXML

2.3.1.1 Inicios

Según registros de la W3C.org [1] los orígenes de VoiceXML tuvieron lugar en 1995 como un lenguaje de diseño de diálogo basado en XML destinado a simplificar el proceso de solicitud de reconocimiento de voz dentro de un proyecto de AT&T llamado Lenguaje de Marcado Telefónico (Phone Markup Language: PML). Cuando AT&T se reorganizó, los equipos de AT&T, Lucent y Motorola continuaron trabajando en sus propios lenguajes PML.

En 1998, el W3C organizó una conferencia sobre los navegadores de voz. Por este tiempo, AT&T y Lucent tenían diferentes variantes de su PML original, mientras que Motorola había desarrollado VoxML, e IBM estaba desarrollando su propio SpeechML.

Muchos otros asistentes a la conferencia también estaban desarrollando lenguajes similares para el diseño de diálogo, por ejemplo, como TalkML de HP y VoiceHTML de PipeBeach.

El Foro de VoiceXML fue entonces formado por AT&T, IBM, Lucent, y Motorola para juntar sus esfuerzos. La misión del Foro VoiceXML fue definir un lenguaje estándar de diseño de diálogo que los desarrolladores pueden utilizar para crear

aplicaciones de conversación. Eligieron XML como base para este esfuerzo porque les quedó claro que esta era la dirección a la cual la tecnología se estaba encaminando.

En el 2000, el Foro de VoiceXML lanzó VoiceXML 1.0 al público. Poco después, VoiceXML 1.0 se presentó a la W3C como base para la creación de una nueva norma internacional. VoiceXML 2.0 es el resultado de este trabajo basado en las aportaciones de las empresas miembros del W3C, otros grupos de trabajo del W3C, y el público.

2.3.1.2 Introducción

VoiceXML está diseñado para crear diálogos de audio que presenta voz sintetizada, audio digital, el reconocimiento del lenguaje hablado y de entrada DTMF, registro de entrada de habla, telefonía, y las conversaciones de iniciativa mixta que son en las que tanto el usuario como el computador pueden afectar la conversación. Su principal meta es acercar las ventajas de la Web basada en el desarrollo y entrega de contenido a las aplicaciones de respuesta de voz interactiva.

2.3.1.3 Objetivos de VoiceXML

El principal objetivo de VoiceXML es llevar un desarrollo completo de la web y entrega de contenido para aplicaciones

de respuesta de voz. Permite la integración de los servicios de voz con servicios de datos usando el formato cliente-servidor.

Un servicio de voz es considerada como una secuencia de cuadros de diálogo de la interacción entre un usuario y una plataforma de aplicación. Los diálogos son proporcionados por servidores de documentos, que puede ser externo a la plataforma de aplicación. Los servidores de documentos mantienen la lógica de servicio general, realizar operaciones de base de datos, y producen cuadros de diálogo. Un documento VoiceXML especifica cada interacción de diálogo que será realizada por un intérprete VoiceXML. Las selecciones de los usuarios afectan la interpretación del diálogo y son recogidas en solicitudes presentadas a un servidor de documentos.

El servidor de documento responde con otro documento VoiceXML para continuar la sesión del usuario con otros cuadros de diálogo.

VoiceXML es un lenguaje que:

- Minimiza las interacciones cliente/servidor mediante la especificación de las interacciones múltiples por documento.
- Separa el código de interacción de usuario (en VoiceXML) de la lógica de servicio (por ejemplo scripts CGI).
- Promueve la posibilidad de utilizar distintas plataformas de ejecución. VoiceXML es un lenguaje común para los proveedores de contenido, proveedores de herramientas y proveedores de plataforma.
- Es fácil de usar para las interacciones simples, y sin embargo, proporciona las características del lenguaje para apoyar los diálogos complejos.

Los alcances que se pueden llegar a tener con VoiceXML son:

- Salida de voz sintetizada (texto a dialogo; text-to-speech).
- Salida de los archivos de audio.
- Reconocimiento de la entrada hablado.
- Reconocimiento de entrada DTMF.
- Grabación de entrada hablada.
- El control del flujo de diálogo.

- Características de telefonía tales como transferencia de llamadas y desconexión.

2.3.1.4 Requisitos para la Implementación de la Plataforma

Ahora describiremos los requisitos para la implementación de la plataformas de hardware / software que apoyarán un intérprete VoiceXML.

- **Adquisición de documentos.** El contexto de interpretación adquirirá documentos para que el intérprete VoiceXML empiece a actuar. El esquema URI y "http" deben ser compatibles.

En algunos casos, la solicitud de documentos se genera por la interpretación de un documento VoiceXML, mientras que otras peticiones son generadas por el contexto de interpretación en respuesta a eventos fuera del ámbito del lenguaje, por ejemplo, una llamada telefónica entrante. Cuando se realiza una búsqueda de solicitudes de documentos a través de HTTP, el contexto de interpretación se identifica así mismo usando para ello el "User-Agent" con

valores de variable de cabecera "<name> / <version>", por ejemplo, "acme-browser/1.2".

- **La salida de audio**. Una plataforma de aplicación debe ser compatible con salida de audio utilizando archivos de audio y de texto a voz (TTS). La plataforma debe ser capaz de generar libremente secuencias TTS y salida de audio pregrabado. Si un recurso de salida de audio no está disponible, un evento error.noresource debe ser lanzada. Los archivos de audio son referidos por un URI. El lenguaje especifica un conjunto necesario de formatos de archivos de audio que se presentan a continuación en la Tabla I.

Tabla I Formatos de Audio

Formato de Audio	Tipo de Medio
Raw (sin cabecera) 8kHz 8-bit mono mu-law [PCM] Un Canal. (G.711)	audio/basic (from [RFC1521])
Raw (sin cabecera) 8kHz 8 bit mono A-law [PCM] Un Canal. (G.711)	audio/x-alaw-basic
WAV (cabecera RIFF) 8kHz 8-bit mono mu-law [PCM] Un Canal.	audio/x-wav
WAV (cabecera RIFF) 8kHz 8-bit mono A-law [PCM] Un Canal.	audio/x-wav

- **Entrada de audio.** Una plataforma de aplicación es necesaria para detectar y reportar caracteres y/o entrada de habla simultáneamente, y para el control de los intervalos de duración mediante un temporizador, cuya longitud se especifica en un documento VoiceXML. Si un recurso de entrada de audio no está disponible, un evento error.noresource debe ser lanzado.

  - Se debe detectar y reportar caracteres (por ejemplo, DTMF) introducidos por un usuario. Las plataformas deben mantener la forma XML de gramáticas DTMF, las cuales se describen en el W3C en la "Especificación de Gramática para el Reconocimiento de Diálogos" (Speech Recognition Grammar Specification: SRGS).

  - Las plataformas deberían ser capaces de recibir datos de reconocimiento del habla de forma dinámica, pero no todas las plataformas poseen esta característica. Para ser más precisos es necesario decir que comercialmente las plataformas utilizan [SRGS] de W3C, no ocurre lo mismo en Código Abierto, donde la plataforma mas poderosa es OpenVXI y sin embargo esta no es capaz de reconoces habla.

- Debe ser capaz de grabar audio recibido por parte del usuario. La plataforma de aplicación debe ser capaz de hacer la grabación a disposición de una variable de petición. El lenguaje especifica un conjunto necesario de formatos de archivos de audio grabados que debe ser los mismos que fueron indicados en la tabla I.

La plataforma de transferencia debe ser capaz de apoyar la toma de una conexión de terceros a través de una red de comunicaciones, tales como el teléfono.

2.3.1.5 Concepto

Una página VoiceXML se configura para formar una conversación que funcionará como una máquina de estados finitos. El usuario está siempre en un estado de conversación o de diálogo, a la vez. Cada cuadro de diálogo determina el siguiente cuadro de diálogo. Las transiciones se especifican mediante URI, que definen el próximo documento y el diálogo en el uso. Si un URI no se refiere a un documento, se supondrá que se refiere al documento actual. Si no se hace referencia a un cuadro de diálogo, se hará suposición de que se refiere al primer cuadro de diálogo en el documento. La

ejecución se termina cuando en un cuadro de diálogo no se especifica un sucesor, o si tiene un elemento que de manera explícita defina las salidas de la conversación.

2.3.1.6 Diálogos y Subdiálogos

Hay dos tipos de diálogos: las formas y los menús. Las formas definen una interacción que recoge los valores de un conjunto de variables. Cada campo puede especificar una gramática que define los aspectos permitidos para ese campo. Un menú presenta al usuario una variedad de opciones y permite la transición a otro diálogo basándose siempre de dicha elección.

Un subdiálogo es como una llamada de función, ya que proporciona un mecanismo para hacer valer una nueva interacción, dentro de un diálogo.

2.3.1.7 Sesiones

Una sesión empieza cuando un usuario empieza a interactuar con un contexto de interpretación VocieXML, continua con la carga y el procesamiento de los documentos, y termina cuando lo requiera ya sea el usuario, el documento (página vxml), o el contexto de interpretación.

2.3.1.8 Aplicaciones

Una aplicación es un conjunto de documentos que comparten el mismo documento raíz de la aplicación. Cada vez que el usuario interactúa con un documento en una aplicación, el documento raíz de su aplicación también es cargado. El documento raíz de la aplicación permanece cargado mientras el usuario está en transición entre otros documentos, en la misma solicitud, y se descarga cuando el usuario realiza transiciones a un documento que no está en la aplicación. Mientras se carga, las variables del documento raíz de la aplicación están disponibles para los otros documentos como variables de aplicación y sus gramáticas permanecen activas durante la duración de la aplicación. La figura 2.3 muestra un esquema de la transición entre documentos en el momento en que se ejecuta una aplicación.

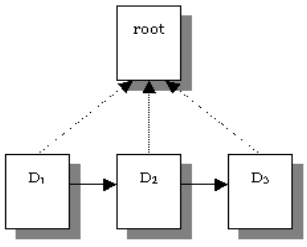

Fig. 2.4 Transición entre documentos dentro de una aplicación.

2.3.1.9 Eventos

Los eventos son aquellos que lanzados por la plataforma bajo una variedad de circunstancias, como las que se presentan cuando el usuario no responde, o responde de manera no entendible, en las solicitudes de ayuda, etc. El intérprete también lanza eventos si encuentra un error semántico en un documento VoiceXML.

2.3.1.10 Grammars

Un grammars especifica una lista de vocabulario admisible (palabras y frases) para que el usuario seleccione en orden la forma de interactuar con la aplicación de VoiceXML. Cada diálogo tiene uno o más Speech y/o grammars asociados a ella.

2.3.1.11 Enlaces

Un enlace especifica una transición que es común a todos los cuadros de diálogo en el ámbito de la relación. Cuando un usuario decide usar un grammar de enlace, el intérprete transfiere el control al destino del enlace.

2.3.1.12 Elementos de VoiceXML

Son aquellos que permiten realizar una programación de un documento VXML, son lo que en lenguajes de programación

tradicional conocemos como comandos o funciones. A continuación se presenta la tabla II, que es una lista de los elementos de VoiceXML [3].

Tabla II Elementos de VoiceXML

Elemento	Propósito
<assign>	Asigna un valor a una variable.
<audio>	Reproduce un audio pregrabado.
<block>	Funciona con un contenedor de código ejecutable.
<catch>	Agarra un evento.
<choice>	Define un item de menú.
<clear>	Limpia una o más variables dentro una forma (form).
<disconnect>	Desconecta una sección.
<else>	Usado en el elemento <if>.
<elseif>	Usado en el elemento <if>.
<enumerate>	Abreviatura de la enumeración de las opciones en un menú.
<error>	Agarra un evento de error.
<exit>	Sale de un sesión.
<field>	Declara un campo de entrada en una forma (form).
<filled>	Una acción es ejecutada cuando los campos son llenados.
<form>	O forma, se usa como dialogo, presenta y recolecta información.
<goto>	Permite dirigirse a otro dialogo ya esa en el mismo o un diferente document.
<grammar>	Especifica si se va a usar reconocimiento de voz o DTMF grammar.

<help>	Agarra un evento de ayuda.
<if>	Simple condición lógica.
<initial>	Declara la lógica inicial una vez que esta dentro de una forma (form).
<link>	Permite la transición a cualquier otro dialogo en otro documento
<log>	Genera un mensaje debug.
<menu>	Permite un dialogo para poder elegir destinos alternativos.
<meta>	Define un item metadata como un par name/value.
<metadata>	Define la informacion metadata usando un esquema metadata.
<noinput>	Agarra un evento que se lanza cuando no hay respuesta en la entrada.
<nomatch>	Agarra un evento que se lanza cuando una busqueda no encuentra lo que ha buscado.
<object>	Interacciona con una extension cualquiera.
<option>	Especifica una opción en un <field>.
<param>	Especifica un parametro en <objetc> o <subdialogo>
<prompt>	Ejecuta el habla sintetizada TTS como salida de audio para el usuario.
<property>	Sirve para configurar la plataforma de implementación.
<record>	Graba un simple audio
<reprompt>	Reproduce un <promp> cuando un campo es visitado de Nuevo despues de haber ocurrido un evento.
<return>	Retorna de un subdiálogo.
<script>	Especifica un bloque de ECMAScript a ser usado.
<subdialog>	Invoca otro diálogo como un subdiálogo del dialogo que se esta ejecutando.
<submit>	Envía valores a un servidor de documentos.

<throw>	Lanza un evento.
<transfer>	Transfiere al usuario que ejecute una llamada hacia otro destino.
<value>	Inserta el valor de una expression en un <prompt>
<var>	Declara una variable
<vxml>	Indica que se trata de un document VoiceXML.

2.3.1.13 Estructura del documento, Ejecución y forma de funcionamiento

Un documento VoiceXML está compuesto principalmente de los mejores elementos a nivel de llamada diálogos. Hay dos tipos de diálogos: las formas y los menús. Un documento también puede tener los elementos <meta> y <metadata>, <var> y <script> elementos, <property> elementos, <catch> elementos, y <link> elementos.

Básicamente la estructura de un documento VoiceXML esta definida de la siguiente manera:

- Un documento contiene uno o mas diálogos
- Una aplicación contiene uno o mas documentos
- Una sesión puede acceder a una o mas aplicaciones

La representación gráfica de la estrucutura jerarquica dentro de VoiceXML se muestra en la figura 2.4.

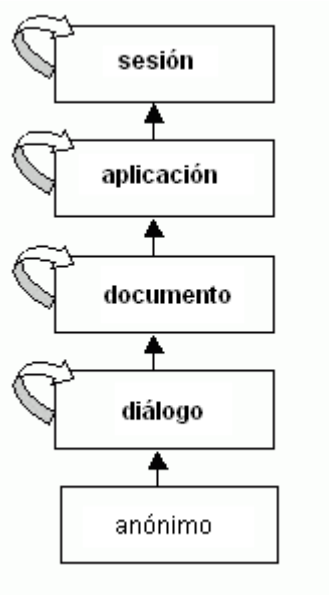

Fig. 2.5 Estructura VoiceXML.

La sintaxis de VoiceXML se basa en el formato ‘tags’ que es el mismo que utiliza HTML y XML. El cual se puede apreciar en el siguiente ejemplo:

< element_name attribute_name="attribute_value">

.....contained items.....

< /element_name>

La programación de aplicaciones VoiceXML se realiza en texto plano y generalmente contiene uno o más documentos, dichos documentos se distinguen por su extensión “.vxml” e inician con las siguientes líneas de condigo:

```
<?xml version="1.0"?>
<vxml version="1.0">
    <form id=“bienvenidos">
        -- contenido --
    </form>
</vxml>
```

Dentro del tag <vxml> el documento se divide en elementos que incluyen diálogos llamados formas. La forma a su vez contiene otros ‘tags’ que ejecutan diversas acciones dentro del programa, para mayor referencia revisar la tabla II.

Como anteriormente se dijo, VoicerXML utiliza varios diálogos que están divididos en formas y menús, estos a su vez contienen elementos como lo son <field>, que son los que reciben la información del usuario para asignar valores a variables, estos dirigen al usuario valiéndose de instrucciones, definen grammars de lo que se puede enunciar y manipulan eventos. Pero las formas y los menús también pueden poseer elementos de control, las cuales no podrán tener tareas de reconocimiento de voz.

La combinación correcta de elementos y respetando los estándares y formatos de la W3C, nos permitirá realizar una gran variedad de programas, teniendo en cuenta que

VoiceXML es una extensión de XML y por lo tanto es compatible con las tecnologías que están basadas en XML como es el caso de RSS.

2.3.2 Asterisk y VoiceGlue

El gran avance tecnológico que ha tenido VoiceXML en la telefonía, ha hecho que se requiera una integración con Asterisk. Es así que existen compañías como i6net que se dedican a desarrollar aplicaciones para que ambas, Asterisk y VoiceXML, puedan ser manejadas dentro de una única plataforma. Por otro lado, se encuentran miembros de la comunidad Código Abierto que trabajan en herramientas similares, de las cuales hemos escogido VoiceGlue para trabajar en esta tesis.

VoiceGlue es un proyecto en que desarrolladores de la comunidad Código Abierto aún siguen trabajando, por lo tanto esta bajo la licencia GPL. En este proyecto utilizaremos VoiceGlue en su versión 0.11, el cual provee de un intérprete VoiceXML para Asterisk usando el intérprete OpenVXI. Este proyecto incluye en su totalidad módulos Código Abierto así como muchas iniciativas de la misma comunidad, cuyo objetivo es crear una solución VXML Código Abierto para los usuarios de Asterisk.

VoiceGlue no está orientado a la gestión de llamadas, tampoco posee un servicio Código Abierto para el reconocimiento automático de voz (ASR), aunque se puede emplear el servicio ASR de LumenVox, el cual es netamente comercial, debido a esto y a que la tesis se basa en emplear software libre y Código Abierto nos hemos visto limitados a no utilizar esta característica. Con excepción de ASR, VoiceGlue posee todas las características de una plataforma VoiceXML 2.0.

Para la utilización de VoiceGlue se requerirá de servicios específicos, como lo son Dynlog, PhoneGlue y la plataforma OpenVXI. Todos estos servicios se encuentran en el paquete de instalación de VoiceGlue 0.11.

El funcionamiento del sistema viene descrito tal como se muestra en la figura 2.5

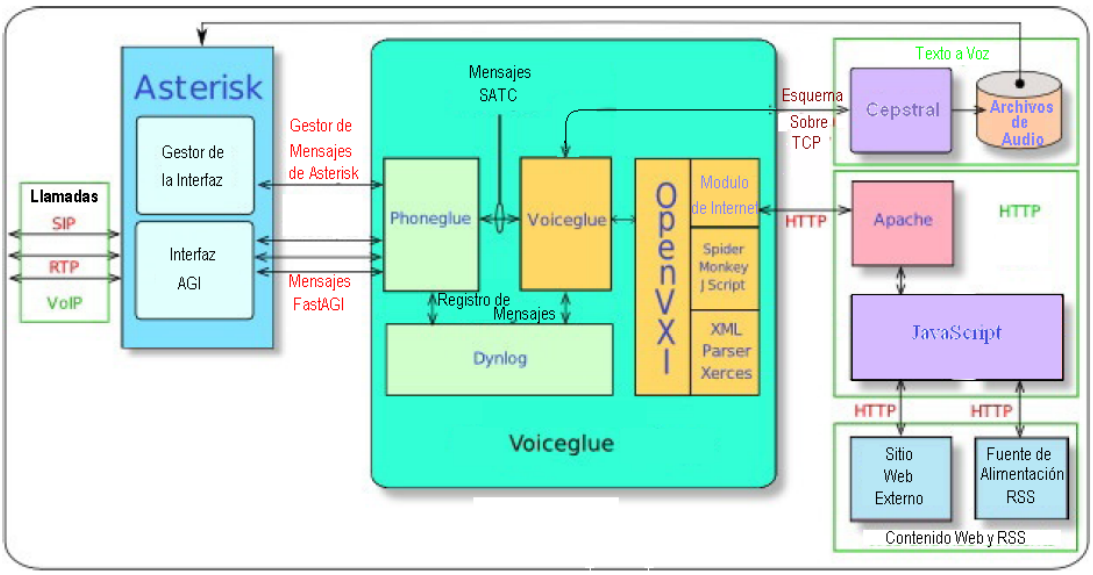

Fig. 2.6 Funcionamiento de VoiceGlue

Al lado izquierdo de la figura se puede observar que llamadas pueden ingresar a nuestro servidor Asterisk, donde la señalización SIP y DTMF son generados por el usuario, y en respuesta a esto el servidor retorna una transmisión RTP de medios de comunicación para acceder al contenido deseado. Es necesario tener en cuenta que debido a la constante búsqueda de convergencia por parte de los desarrolladores, los medios de comunicación no deben estar limitados solo a audio, aunque el alcance de este proyecto si lo esté.

Como se puede observar en la parte central de la fig 2.5, Asterisk está bajo el control de VoiceGlue cuando los archivos de configuración **Manager Interface** y **Agi Interface** son adaptados o configurados para cumplir con el propósito de brindar el servicio VoiceXML Browser. Esto añadirá a Asterisk la capacidad de ejecutar correctamente la voz de los cuadros de diálogo expresada por los documentos de VoiceXML. Dentro de VoiceGlue se encuentran procesos como **PhoneGlue**, que utiliza los puertos configurados en Asterisk Manager para la apertura de los canales FastAGI, el cual le servirá a PhoneGlue para ser notificado acerca de las interacciones remotas que se den, y por la que PhoneGlue solicitará la reproducción de archivos de audio. VoiceGlue, refiriéndonos al proceso interno del VoiceGlue global, está escrito para ser un ente independiente de las aplicaciones de Asterisk, y que utiliza un protocolo específicamente diseñado para comunicarse con

PhoneGlue denominado SATC (Control Telefónico de ASCII Simple, Simple ASCII Telephony Control). VoiceGlue basa su funcionamiento en componentes como los son las librerías OpenVXI, las cuales ofrecen los servicios en los que se basa un VoiceXML Browser. Las librerías OpenVXI utilizadas por VoiceGlue usan algunos componentes de Código Abierto, como lo son:

- XML Parces Xerces, que es un analizador XML, que permite utilizar los componentes de la librería OpenVXI al interprete VoiceXML.
- El Interprete de Java Script Spider Monkey, es usado para interpretar JavaScript's debido a que usualmente se usan estos como componentes para ayudar a la navegación por audio dentro de VoiceXML.
- Modulo de Internet, es usado para obtener nuevos documentos VoiceXML de un servidor Web.

El programa de **Dynlog** recoge todos los registros de los procesos internos PhoneGlue y VoiceGlue. No es estrictamente necesario, pero su ausencia significaría recorrer varios archivos en caso de no saber lo que esta pasando. Los registros se guardan en /var/log/dynlog/dynlog.

Para finalizar es necesario aclarar que este proyecto usará como archivos fuentes, documentos RSS, es decir que nuestro VoiceXML Browser trabajara con la lectura específicamente de estos archivos. Los documentos VoiceXML que interaccionan con el usuario se encuentran alojado en un servidor de web de documentos, en este caso un servidor Apache. Dentro de esta interacción, el usuario solicita indirectamente la lectura de un documento RSS, esta petición se realiza vía HTTP.

Una vez que se tiene como respuesta el documento RSS, un JavaScript implementado nos permite obtener información especifica de dicho documento en formato de texto plano, de este modo dicha información regresa al documento VoiceXML en forma de variable. El destino siguiente será nuestro servidor TTS, en este caso Cepstral, el cual se encargara de transformar el texto plano a archivos de audio que son los que finalmente escuchará el usuario.

2.3.3 RSS

Es un acrónimo para referirse a un grupo de formatos de fuente web codificados en XML. Se emplean para brindar a los suscriptores de información actualizada frecuentemente. Utilizan los siguientes estándares, de hecho de ahí viene el significado de sus siglas:

- Rich Site Summary (RSS 0.91)
- RDF Site Summary (RSS 0.9 y 1.0)

- Really Simple Syndication (RSS 2.0)

La facilidad que brindan es que no necesitan de un navegador web para ver los contenidos RSS, basta con instalar un software diseñado para leer estos contenidos, sin que esto signifique que no se pueda acceder a estos con un navegador web.

Otra de las razones, la cual es la que mas nos interesa para el desarrollo de este proyecto es que esta familia de formatos XML fue desarrollada específicamente para todo tipo de sitios web que se actualicen con frecuencia y por medio del cual se puede compartir la información y usarla en otros sitios web o programas. Esto es lo que se conoce como redifusión o sindicación web. Esto nos permitirá tener información de la web actualizada en nuestro IVR.

Otro punto de extrema importancia es que los documentos RSS que se van a usar para nuestro proyecto deben cumplir sin excepción alguna con el estándar del formato XML. Para la verificación de esto, hemos empleado la página de la W3C (www.W3.org) y hemos verificado que las páginas que utilizamos cumplan con el formato apropiado. La figura 2.6 muestra la validación del documento RSS.

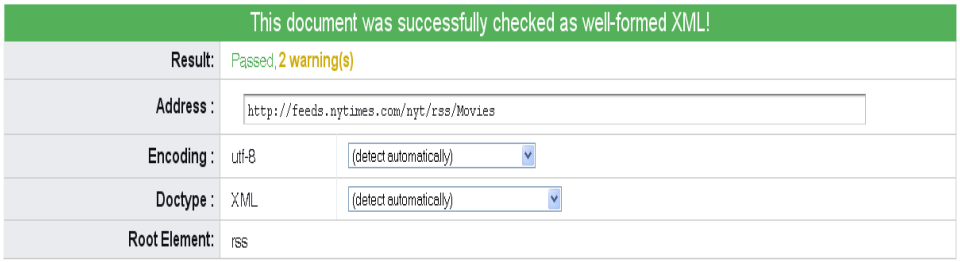

Fig. 2.6 Validación de RSS

CAPITULO 3

IMPLEMENTACION

3.1 INTRODUCCION

Es importante destacar el uso de las soluciones basadas en software libre, como el hecho de que ayudan a reducir costos, nos facilita la corrección de errores gracias a la siempre colaborativa comunidad Código Abierto y nos permite desarrollar aplicaciones basadas en aplicaciones ya existente.

En el caso más puntual, el objetivo de esta tesis es brindar un soporte a personas cuya capacidades especiales no les permitan leer información en la web, del mismo modo son impedidos de poder navegar a través de ella, ya que los tradicionales navegadores solo permiten la interacción por medio de dispositivos como lo son un teclado, un ratón y una pantalla.

La implementación de sistema IVR liga a dos partes muy importantes, una es VoiceGlue cuyos recursos están netamente destinados a proporcionar un intérprete de lenguaje VoiceXML. El otro es Asterisk cuyo trabajo será vincular sus usuarios para que puedan disponer de ese servicio, además de eso, prestará sus canales AGI para la interacción entre el usuario y el intérprete.

Dicho esto es necesario considerar que se usará un ordenador que funcionará como central Asterisk, en el mismo ordenador realizaremos la instalación del Intérprete VoiceXML y el servidor web (en nuestro caso Apache).

3.2 HARDWARE

Definir el tipo de hardware que se utilizará es una decisión muy importante, que debe estar basada en aspectos como el tiempo de uso diario que se le dará al servidor Asterisk, la cantidad de llamadas que el mismo servidor procesará y si fuese el caso la cantidad de datos que se almacenarán en la base de datos ligada con Asterisk. Esto nos dará una visión mas clara del tipo de tarjeta madre, memorias RAM, dispositivos de almacenamiento y fuente de poder que usaremos.

3.2.1 Servidores

Se ha trabajado con dos servidores, un servidor Asterisk y un servidor web. Ambos servidores fueron instalados en un mismo ordenador, cuyas características son las siguientes:

Tabla III Características del Ordenador

CPU	Intel Pentium 4 de 3Ghz
RAM	512 MB
Disco Duro	20 GB
Tarjeta de Red	10/100 Mpbs

3.2.2 Teléfono IP

En nuestras pruebas hemos utilizado solo un teléfono IP, que es aquel que se encuentra actualmente en uso en el laboratorio de telecomunicaciones, el equipo GXP2000 que se muestra en la figura 3.1.

Fig. 3.1 GXP2000

3.3 SOFTWARE

3.3.1 Servidor Apache

La siguiente tabla muestra detalles sobre la instalación del servidor Apache:

Tabla IV Servidor Apache

Sistema Operativo	Ubuntu 8.04 – Linux 2.6.24-24 generic
Software Servidor Web	Apache 2
Base de Datos	Mysql server 5.0
Add-on	Phpmyadmin

3.3.2 Servidor Asterisk

A continuación se detallan los componentes instalados para el funcionamiento del servidor Asterisk.

Tabla V Servidor Asterisk

Sistema Operativo	Ubuntu 8.04 – Linux 2.6.24-24 generic
Software IP PBX	Asterisk versión 1.4.26
Plataforma VoiceXML	VoiceGlue 0.11

Para el correcto funcionamiento de Asterisk se instalaron las siguientes librerías:

bison

flex

libncurses5-dev

zlib1g-dev

libssl-dev

libnewt-dev

libiksemel-dev

gcc g++

libstdc++6

3.3.3 Plataforma VoiceXML

La siguiente tabla muestra los componentes enumerados según el orden de instalación que se usaron para el correcto funcionamiento del Interprete VoiceXML, estos componentes vienen dentro del paquete de instalación de VoiceGlue 0.11 y se instalan automáticamente cuando se ejecuta el script de instalación y son los siguientes:

Tabla VI Componentes VoiceGlue

Componente	Contenido
1. Cam-Scom	Librería Perl para IPC.
2. Dynlog	Programa de Perl que provee un registro de programas ejecutados.

3. Vgluefftw	Librería Perl para acceder a FFTW.
4. Satc	Librería Perl para la comunicación con PhoneGlue.
5. Voiceglue-Conf	Librería Perl para la configuración de VoiceGlue.
6. Phoneglue	Programa Perl que proporciona una interfaz de alto nivel para Asterisk.
7. SRGSDTMF	Librería Perl para el procesamiento de gramáticas (Grammar's) DTMF.
8. libvglue-headers	Librerías de cabecera VoiceGlue C/C++.
9. openvxi-3.4+vglue	Librerias OpenVXI modificadas para VoiceGlue C/C++.
10. libvglue	Librerías de VoiceGlue C/C++.
11. Vxglue	Librería Perl para la interfaz con OpenVXI.
12. voiceglue	Programa Perl que proporciona la interfaz con VoiceGlue.

Previo a la instalación de VoiceGlue fue necesario la instalación las siguientes librerías para el correcto funcionamiento de Interprete VoiceGlue:

gcc g++

libxerces28-dev

SpiderMonkey

flite

sox

libsox-fmt-all

wget

libssl-dev

libbsd-resource-perl

libmodule-build-perl

libfftw3-dev

pkg-config

3.3.4 Softphone's

Un softphone es software que simula el funcionamiento de un teléfono, en este proyecto hemos utilizado 3 diferentes softphone's. Uno que soporta solo extensiones SIP, cuyo nombre es **X-Lite**, otro que soporta solo extensiones IAX llamado **Kiax** y un tercero **Zoiper**, que era capaz de soportar ambos tipos de extensiones, IAX y SIP.

3.4 INSTALACION

3.4.1 Instalación de Servidor Web Apache

Dicha instalación la realizamos mediante el modo gráfico de Ubuntu.

Siguiendo los pasos a continuación detallados:

1. Accedemos al gestor de paquetes de Ubuntu llamado Synaptic Package Manager, utilizamos la herramienta de búsqueda, en ella escribimos "Apache" y seleccionamos la opción "Apache2" que nos arrojará como resultado de la búsqueda. Ahora escogemos la opción gráfica "Aplicar" del gestor de paquetes de Ubuntu, tal como se muestra en la figura 3.2.

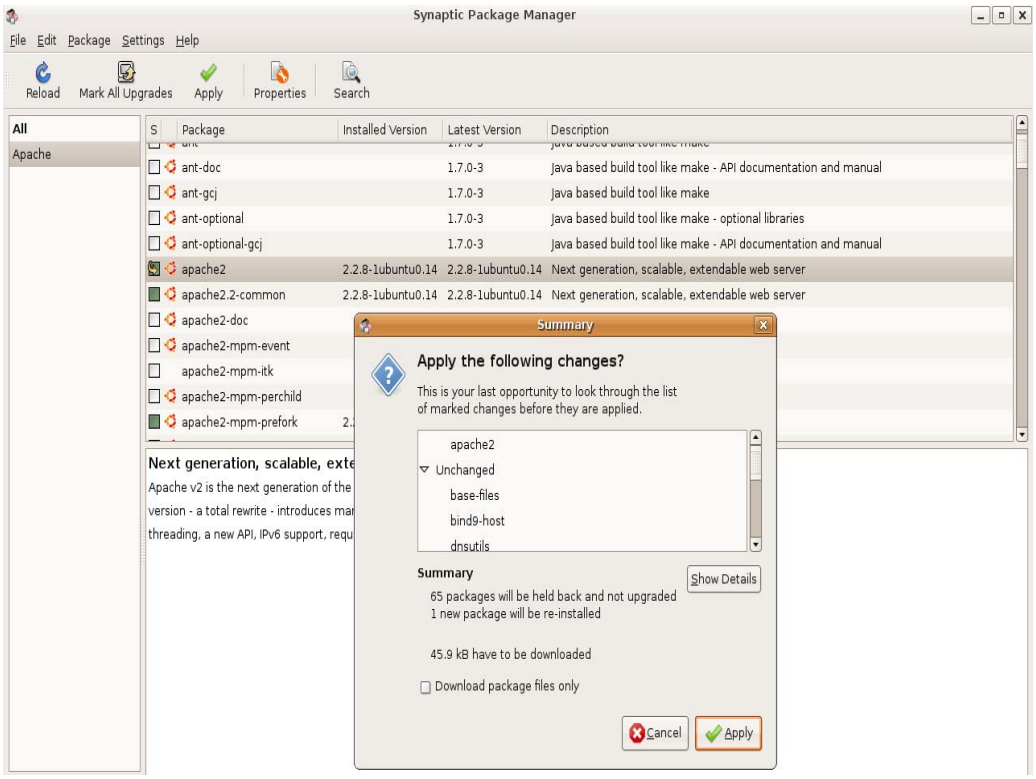

Fig. 3.2 Proceso de Instalación de Apache

2. Una vez que se ha instalado el servidor web, procedemos a instalar la base de datos, para esto realizaremos la búsqueda de “mysql server” y escogeremos la opción “mysql-server-5.0”, e instalamos de la misma manera que lo hicimos con Apache. A la mitad del proceso se realizará una petición de clave. Completamos ese requerimiento y continuamos con la instalación.

3. Para finalizar colocamos en la herramienta gráfica de búsqueda “phpmyadmin” y seleccionamos “phpmyadmin”, procedemos de la misma manera que antes para continuar con la instalación. Al final

de la instalación nos preguntará con que servidor deseamos vincular la instalación, escogemos la opción “Apache” y esperamos que termine la instalación.

Es necesario aclarar que todas las instalaciones se deben hacer como usuario “root”.

3.4.2 Instalación de Librerías Base para Asterisk.

De ahora en adelante consideraremos que la ejecución de comandos se realizará en la consola de Ubuntu. Lo siguiente es para la instalación de las librerías requeridas por Asterisk:

```
apt-get install bison
apt-get install flex
apt-get install libncurses5-dev
apt-get install zlib1g-dev
apt-get install libssl-dev
apt-get install libnewt-dev
apt-get install libiksemel-dev
apt-get install gcc g++
apt-get install libstdc++6
```

3.4.3 Instalación de Librerías para VoiceGlue.

Para la instalación de las librerías utilizadas por VoiceGlue ejecutamos lo siguiente:

apt-get install libxerces28-dev

apt-get install SpiderMonkey

apt-get install flite

apt-get install sox

apt-get install libsox-fmt-all

apt-get install wget

apt-get install libssl-dev

apt-get install libbsd-resource-perl

apt-get install libmodule-build-perl

apt-get install libfftw3-dev

apt-get install pkg-config

3.4.4 Instalación de Asterisk.

Nos colocamos en el directorio /usr/src y descargamos los paquetes de Asterisk.

cd /usr/src

wget http://downloads.digium.com/pub/asterisk/asterisk-1.4-current.tar.gz

wget http://downloads.digium.com/pub/asterisk/asterisk-addons-1.4-current.tar.gz

Después descomprimimos los paquetes ejecutando los siguientes comandos:

tar xzvf asterisk-1.4-current.tar.gz

```
tar xzvf asterisk-addons-1.4-current.tar.gz
```

Entramos en la primera carpeta que descomprimimos y compilamos Asterisk:

```
cd /usr/src/asterisk-1.4.2
make clean
./configure
make
make install
make samples
make config
```

Ahora llegó el turno de compilar los add-on's:

```
cd /usr/src/asterisk-addons
make clean
./configure
make
make install
make samples
```

Este ultimo no es necesario, solo lo ejecutamos si deseamos tener configuraciones de ejemplo.

Esto bastará para poder tener una central telefónica con las características necesarias para este proyecto.

3.4.5 Instalación de VoiceGlue.

Antes de proceder con la instalación es necesario primero crear un usuario “asterisk” en Ubuntu. Este usuario es el que ejecuta Voiceglue como proceso porque necesita compartir archivos de audio con Asterisk. Esto es posible, incluso si Asterisk y VoiceGlue están en máquinas separadas, pero para que esto funcione deben hacer coincidir procesos uid/gid de el usuario “asterisk” con el Host donde esta ejecutándose Asterisk. El usuario “asterisk” debe estar en el mismo host que VoiceGlue.

Después de que todos los requisitos han sido instalados, procedemos a descargar y descomprimir VoiceGlue 0.11

```
cd /usr/src
wget http://www.voiceglue.org/voiceglue_0.11.tar.gz
tar –xzvf voiceglue_0.11.tar.gz
```

Entramos en la carpeta que acabamos de descomprimir y ejecutamos el siguiente:

```
doc/install-voiceglue
```

Si hemos instalado todos los requerimientos antes descritos, no tendremos problema en este punto. Ya que lo que hace ese comando es ejecutar un Script de instalación. Después de esto ya podemos ser capaces de ejecutar los siguientes guiones:

/etc /init.d/dynlog

/etc /init.d/phoneglue

/etc/init.d/voiceglue

Es necesario que se respete el orden de ejecución de los comandos anteriores, ya que de lo contrario tendremos problemas para arrancar estos procesos.

3.4.6 Instalación de Zoiper.

Para poder usar Zoiper primero es necesario descargarlo e instalarlo. Los descargamos de http://www.zoiper.com/softphone/. Nosotros usaremos el paquete "zoiper-communicator-free-alsa_1.0-1ubuntu9_i386.deb", una vez descargado para instalarlo solo basta con dar clic derecho sobre el paquete y escoger la opción **Open With "GDebi Package Installer"**, se abrirá una ventana y escogemos instalar paquete. En la siguiente figura se muestra el softphone Zoiper.

Fig. 3.3 Interfaz gráfica de Zoiper

3.5 CONFIGURACION DE ARCHIVOS DE ASTERISK y VoiceGlue

3.5.1 Configuración de archivo “sip.conf”

Este archivo contiene las configuraciones de las extensiones SIP que se van a emplear. Esta ubicado en el directorio “/etc/asterisk/sip.conf”. El “sip.conf” permite modificar y controlar muchos aspectos sobre las peer SIP que se van a usar, dichas modificaciones se pueden hacer globalmente o a una extensión en específico. Debido a que nosotros no nos basamos exclusivamente en el uso de canales SIP, vamos a realizar una configuración básica que nos servirá para poder comprobar que el IVR con VoiceXML Browser puede recibir una llamada de una extensión SIP.

La configuración que usamos fue la siguiente:

[401]

type=friend

host=dynamic

secret=1234

callerid=401 <401>

context=phoneglue

Donde **type** indica que el usuario puede recibir y hacer llamadas, **host** indica en este caso que se le solicitará al usuario registrarse para poder entrar al servidor Asterisk, **secret** le otorga una contraseña que servirá al usuario al momento de registrarse en el servidor, el **callerid** vincula una identificación para la extensión que se esta usando, en este caso ambas con iguales, el **context** indica el contexto al que pertenece el usuario, en este caso es necesario que sea phoneglue, para que de este modo pueda llamar a la extensión del IVR con VoiceXML Browser que también estará en el mismo contexto. Recordemos que los usuarios de Asterisk por defecto pueden realizar llamadas dentro del mismo contexto.

3.5.2 Configuración de archivo “iax.conf”

Este archivo contiene las configuraciones de las extensiones IAX que se van a emplear. Esta ubicado en el directorio “/etc/asterisk/iax.conf”.

El archivo de “iax.conf” es tratado del mismo modo que el archivo “sip.conf” en este proyecto, es decir solo empleamos una configuración básica, la cual es la siguiente.

[Efren]

type=friend

host=dynamic

secret=1234

context=phoneglue

callerid=Efren <123>

3.5.3 Configuración de archivo “extensions.conf”

Se encuentra ubicado en el directorio /etc/asterisk/extensions.conf.

Este archivo le indica a Asterisk el plan de marcado, el cual es suma importancia dentro de una central telefónica, porque nos dice el plan de numeración que usará la central telefónica Asterisk para manejar cada contexto y por lo tanto cada usuario.

El archivo “extensions.conf” permite establecer configuraciones para el uso de las extensiones, así como también permite definir variables y realizar la programación de un plan de marcado. Nuestro proyecto solo se centrará en el uso de un solo contexto el cual se haya en el plan de marcado de la siguiente forma.

[phoneglue]

exten=>1,1,Answer()

exten=>1,2,Agi(agi://localhost/url=http%3A%2F%2Flocalhost%2FIVR.vxml&arg1=foo&arg2=bar)

exten=>1,3,Hangup()

Para el resto de configuraciones adoptaremos las que se encuentran por defecto en “extensions.conf” después de la instalación del servidor Asterisk.

Volviendo la configuración del contexto phoneglue, lo que sucederá es que cuando un usuario que se encuentra en el mismo contexto marque “1”, el IVR con VoiceXML Browser será quien le conteste. La ultima línea, que posee la prioridad 3, indica que el usuario tendrá la potestad de colgar cuando el desee.

3.5.4 Configuración de archivo “manager.conf”

El servicio PhoneGlue necesita registrarse en el administrador de Asterisk con usuario y contraseña igual a “phoneglue”. Por lo que es necesario configurar el archivo “manager.conf” de Asterisk que se encuentra en /etc/asterisk/manager.conf. A continuación se detalla las líneas de configuración necesarias:

```
[general]
    displaysystemname = yes
    enabled = yes
    port = 5038
[phoneglue]
    secret=phoneglue
    read = system,call,log,verbose,command,agent,user
```

write = system,call,log,verbose,command,agent,user

3.5.5 Configuración de archivo “voiceglue.conf”

Ubicado en /etc/voiceglue.conf. Este archivo contiene la definición de ast_sound_dir, que significa que aquí es donde se escribe la dirección donde se guardarán los archivos que posteriormente serán reproducidos por el Intérprete VoiceXML. Dicha línea de código es la siguiente:

/var/lib/asterisk/sounds

En esta dirección se creará automáticamente una carpeta llamada VoiceGlue, la cual contendrá otras cuatro carpetas, que contendrán los archivos de audio antes mencionados.

Además este archivo puede tener líneas adicionales, las cuales servirán como mapa para que las llamadas entrantes se dirijan a un url especifico. Pero debido a que nosotros emplearemos comandos AGI estas líneas serán ignoradas, debido a eso hemos omitido este tipo de configuración.

3.5.6 Configuración del programa “voiceglue_tts_gen”

La importancia de este programa es fundamental, puesto que cada vez de el servicio TTS es requerido, VOICEGLUE ejecuta el programa /usr/bin/voiceglue_tss_gen.

La programación por defecto del programa es la siguiente (la podemos ver ejecutando el comando “gedit /usr/bin/voiceglue_tss_gen”);

```
#!/usr/bin/perl --       -*-CPerl-*-
$file = $::ARGV[$#ARGV];
system ("flite", @::ARGV);
system ("mv", $file, $file . ".16khz.wav");
system ("sox", $file . ".16khz.wav", "-r", "8000", $file);
```

Debido a que la programación antes mostrada fue hecha para el uso del TTS FLITE, y en este proyecto se emplea TTS de Cepstral, nos vimos en la obligación de cambiarla por la siguiente programación:

```
#!/usr/bin/perl --       -*-CPerl-*-
# Cepstral interface
$file = $::ARGV[$#ARGV];
system ("/usr/local/bin/swift", "-m", "text", "-o",$file ,"-e","utf-8",$ARGV[1]);
```

Donde la última línea de código nos permite usar el TTS en español con la codificación “UTF-8” que entre otras cosas distingue el uso de las tildes y la letra “ñ”.

CAPITULO 4

IMPLEMENTACION DEL IVR CON

VOICEXML BROWSER

Una vez que se han instalados los componentes necesarios, procederemos a la implementación del IVR con VoiceXML Browser. En este capítulo detallaremos los pasos para el correcto funcionamiento del IVR.

4.1 ADMINISTRACION DE SERVICIOS

A continuación detallaremos un conjunto de comandos para consola que nos permiten iniciar o detener el servicio de Astersik.

/etc/init.d/asterisk start	Inicia el servicio Asterisk.
/etc/init.d/asterisk stop	Detiene el servicio Asterisk.
/etc/init.d/asterisk restart	Detiene y vuelve a Iniciar el servicio Asterisk.

Para poder emplear el Intérprete VoiceXML (VoiceGlue), es necesario iniciar los servicios dynlog, phoneglue y voiceglue. De la siguiente manera:

/etc/init.d/dynlog start	Inicia el servicio Dynlog.
/etc/init.d/phoneglue start	Inicia el servicio PhoneGlue.
/etc/init.d/voiceglue start	Inicia el servicio VoiceGlue.

Es necesario que siempre se respete este orden de ejecución de los comandos al iniciar cada servicio, de lo contrario se podrían presentar errores y no se iniciarían los servicios. Estos tres servicios funcionan como procesos internos del Intérprete VoiceGlue.

Si se desea detener los servicios se debe ejecutar los comandos en el siguiente orden.

/etc/init.d/voiceglue stop	Detiene el servicio VoiceGlue.
/etc/init.d/phoneglue stop	Detiene el servicio PhoneGlue.
/etc/init.d/dynlog stop	Detiene el servicio Dynlog.

Otros comandos de consola que nos permiten ingresar y salir del CLI Asterisk.

asterisk -r	Ingresa al CLI de Asterisk.
asterisk -c	Inicia Asterisk e ingresa al CLI de Asterisk.
asterisk -rx 'comando'	Ejecuta un comando sin ingresar al CLI de Asterisk.

Dentro del CLI de Asterisk los comandos que más usamos fueron los siguientes:

agi debug on	Muestra constantemente el estado de los canales AGI.
sip show peers	Muestra es estado de los usuarios SIP.
iax2 show peers	Muestra el estado de los usuarios IAX.
dialplan reload	Vuelve a cargar el plan de marcado.
restart now	Reinicia el servicio de Asterisk.
stop now	Detiene el servicio de Asterisk.
reload	Vuelve a carga todas las configuraciones de Asterisk.
exit	Para salir del CLI de Asterisk.

4.2 CONFIGURACION DE LOS USUARIOS DE LA CENTRAL TELEFONICA

En el capítulo anterior se llevó a cabo la configuración de los archivos "iax.conf" y "sip.conf", ahora procederemos a configurar los softphone y el teléfono IP.

Nosotros realizamos las pruebas con 3 tipos diferentes de softphone, pero ahora solo mostraremos la configuración de uno, el que hemos escogido para mostrar es "Zoiper", además mostraremos la configuración de una línea del teléfono IP Grandstream GXP2000.

4.2.1 Configuración de Zoiper

Para poder utilizar Zoiper primero necesitamos poseer una cuenta Zoiper, así que accedemos a la página oficial de Zoiper y creamos una.

Ahora procedemos a configurar una cuenta IAX, para esto primero ejecutamos Zoiper. En nuestro caso debemos primero dirigirnos al menú de Ubuntu, dar clic en la viñeta Aplicaciones, posarnos en Internet y al desplegarse el menú escogemos Zoiper Communicator.

Una vez abierto el programa ingresamos los datos de nuestra cuenta zoiper y presionamos en botón "Sign In". Esto nos llevara al programa principal del softphone, en el cual nos vamos a dirigir a la viñeta que dice Zoiper y vamos a escoger la opción Preferences, se nos

desplegará una nueva pantalla, ahora nos dirigimos a IAX accounts, y presionamos donde dice **Create new IAX account**, escogemos el nombre de la Cuenta y presionamos OK. Procedemos a llenar los datos en los campos correspondientes.

Fig. 4.1 Configuración de Zoiper

Como la figura lo muestra, debemos llenar los datos Server Hostname/IP, el cual deberá ser la dirección IP del Servidor Asterisk. El Username, es el nombre de usuario que configuramos en el "iax.conf", en este caso "Efren". De igual forma se debe colocar en el campo Password el secret relacionado con el usuario "Efren", así como el Caller ID Name y el Caller ID Number. Después de terminar con la configuración presionamos REGISTER, y esperamos que se conecte a nuestra central.

4.2.2 Configuración de GXP2000

Este dispositivo lo configuraremos por medio de un navegador web. Abrimos el navegador y escribimos la dirección IP asignada al teléfono. Un menú se nos mostrará y pedirá una contraseña, ingresamos la contraseña del teléfono. Hecho esto ingresaremos a administrar el teléfono.

Fig. 4.2 Configuración de GXP2000

Una vez que ingresemos se nos mostrará un menú para la configuración del teléfono. La siguiente figura nos muestra los parámetros de configuración para la extensión IP que usamos en este proyecto.

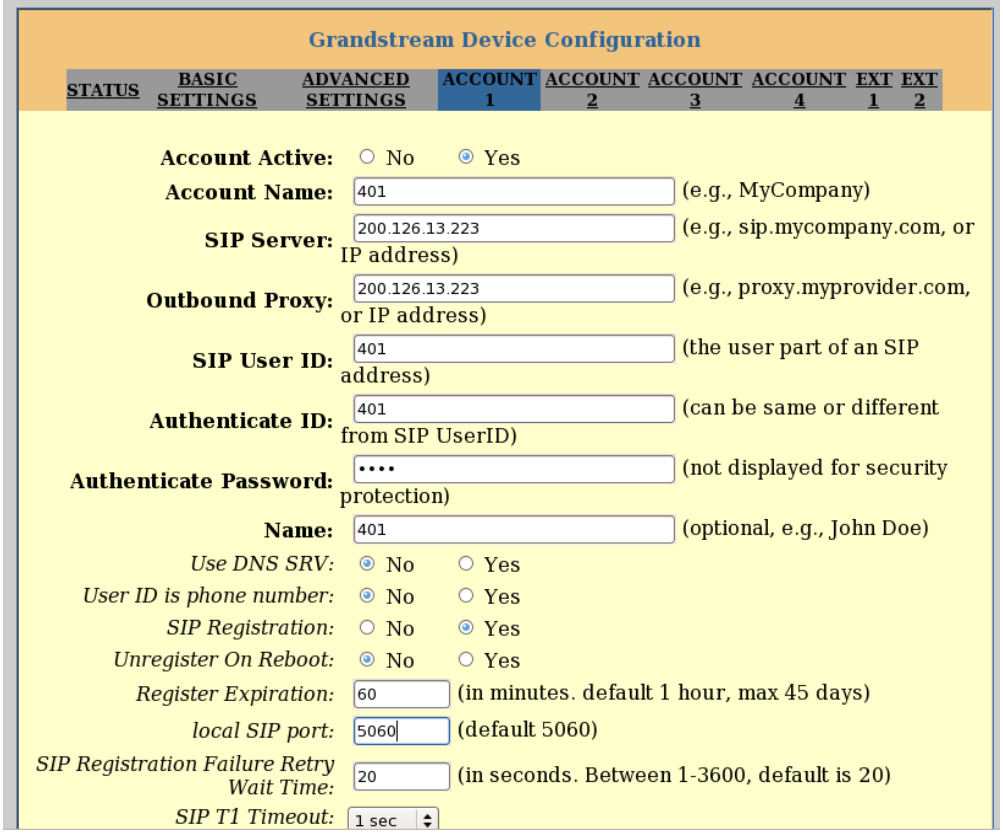

Fig. 4.3 Configuración de GXP2000 – cuenta sip

Después de haber realizados los cambios en la configuración del teléfono, debemos hacer una actualización y posteriormente reiniciar el teléfono; esto lo hacemos dando un clic en el botón update que esta al final de la pagina de configuración y luego aparecerá otra ventana donde está el botón Reboot.

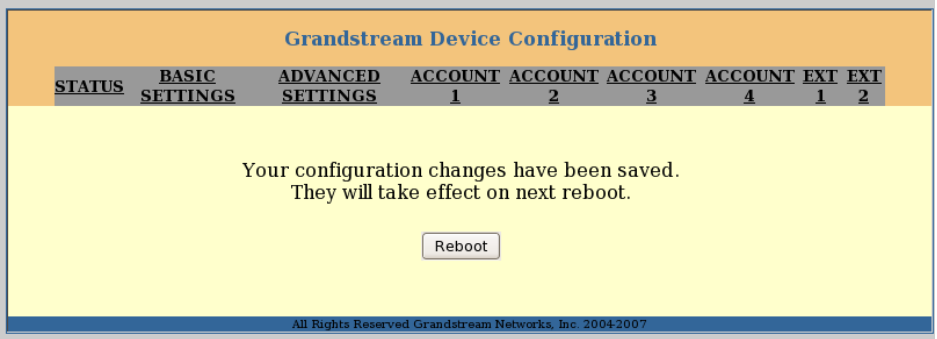

Fig. 4.4 GXP2000 Configurado

4.3 PROGRAMACION DEL IVR

La programación del IVR deberá ser hecho en documentos VoiceXML cuya extensión es ".vxml" y que debe cumplir con los formatos antes especificados Nuestro IVR tendrá el siguiente comportamiento:

Al recibir una llamada de un usuario, las llamadas serán atendidas por un documento llamado "IVR.vxml", que se encargará de proveer un menú basado en los siguientes temas:

4) El Universo
4) Cine
4) Blog ESPOL
4) Diario Espana

El documento **"IVR.vxml",** que es el encargado de mostrarnos la lista de opciones que el usuario puede seleccionar, posee el siguiente contenido:

```
<?xml version="1.0" encoding="UTF-8"?>

<vxml version="2.1" xmlns:voxeo="http://community.voxeo.com/xmlns/vxml">

  <property name="confidencelevel" value=".35"/>
  <property name="sensitivity" value="0.35"/>

 <form id="Intro">

   <block>
    <prompt bargein="true">
                    Bienvenido al I Ve ere con vois equis eme ele brouser
       implementado por Efrén Gómez y Yovani Izza.
    </prompt>
    <goto next="#Topic"/>
  </block>
 </form>


<menu id="Topic" dtmf="true">
 <property name="inputmodes" value="dtmf"/>
 <prompt>
                    A continuación le presentamos cuatro fuentes importantes
de
       noticias, elija una de ellas para comenzar.
   <enumerate>

                     Para      <value expr="_prompt"/>, presione <value
expr="_dtmf"/>.
   </enumerate>
 </prompt>
<choice next="#universo">
   El Universo
 </choice>
<choice next="#cine">
   Cine
 </choice>
<choice next="#blog">
   Blog de la ESPOL
 </choice>
<choice next="#esp">
   Noticias de España
 </choice>
</menu>

<form id="universo">
  <block>
   <prompt>Tu has elegido EL universo </prompt>
```

```
    <goto next="eluniverso.vxml"/>
  </block>
</form>
<form id="cine">
  <block>
    <prompt>Tu has elegido Cine</prompt>
    <goto next="cine.vxml"/>
  </block>
</form>
<form id="blog">
  <block>
    <prompt>Tu has elegido Blog de la ESPOL</prompt>
    <goto next="blogespol.vxml"/>
  </block>
</form>
<form id="esp">
  <block>
    <prompt>Tu has elegido Noticias de España</prompt>
    <goto next="espana.vxml"/>
  </block>
</form>

</vxml>
```

Una vez que el usuario selecciona alguna de las opciones procede a los documentos que le corresponden, estos documentos son “eluniverso.vxml”, “cine.vxml”, “blogespol.vxml”. Los contenidos de cada uno de estos archivos se detallan a continuación junto con su funcionalidad:

“eluniverso.vxml”

Este archivo permite al usuario conocer sobre las noticias nacionales en diferentes ámbitos, como el caso de Política, Sucesos, Deportes, etc. Se puede elegir cada una de las opciones, con la posibilidad de volver a escuchar sobre algunas de las noticias o volver al menú principal si el usuario lo requiere. Su contenido es el siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>

<vxml version="2.1" xmlns:voxeo="http://community.voxeo.com/xmlns/vxml">

 <script src="parseRSSData.js" fetchtimeout="15s"/>
  <property name="confidencelevel" value=".35"/>
  <property name="sensitivity" value="0.35"/>

  <var name="topicRSS"/>
  <var name="dataRSS"/>
  <var name="descriptionArrayRSS"/>
  <var name="fetchRSS"/>
  <var name="resultCount"/>
  <var name="titleArrayRSS"/>
 <form id="Intro">

   <block>
    <prompt bargein="true">
          Bienvenido a la página del universo, aquí podras encontrar
          cualquier tema al día.
    </prompt>
    <goto next="#Topic"/>
  </block>
 </form>


<menu id="Topic" dtmf="true">
 <property name="inputmodes" value="dtmf"/>
 <prompt>
          Para comenzar, por favor elija un tema del cual desee escuchar
          noticias.
   <enumerate>

           Para        <value expr="_prompt"/>, presione <value expr="_dtmf"/>.
   </enumerate>
 </prompt>
<choice next="#portada">
   Portada
 </choice>
<choice next="#politica">
   Politica
 </choice>
<choice next="#elpais">
   El pais
 </choice>
<choice next="#opinion">
   Opinion
 </choice>
```

```
<choice next="#migrantes_tema">
  Migracion
 </choice>
<choice next="#deportes">
  Deportes
 </choice>
 <choice next="#Sucesos">
  Sucesos
 </choice>
 <choice next="#ivrorigin">
  Volver al I Ve eRe
 </choice>
</menu>

<form id="portada">
  <block>
    <prompt>Tu has elegido </prompt>
    <assign name="document.topicRSS" expr="'portada'"/>
          <audio expr="document.topicRSS + '.wav'">
          <value expr="document.topicRSS"/>
          </audio>
    <goto next="#Headlines"/>
  </block>
</form>
<form id="politica">
  <block>
    <prompt>Tu has elegido </prompt>
    <assign name="document.topicRSS" expr="'politica'"/>
          <audio expr="document.topicRSS + '.wav'">
          <value expr="document.topicRSS"/>
          </audio>
    <goto next="#Headlines"/>
  </block>
</form>
<form id="elpais">
  <block>
    <prompt>Tu has elegido </prompt>
    <assign name="document.topicRSS" expr="'elpais'"/>
          <audio expr="document.topicRSS + '.wav'">
          <value expr="document.topicRSS"/>
          </audio>
    <goto next="#Headlines"/>
  </block>
</form>
<form id="opinion">
  <block>
    <prompt>Tu has elegido </prompt>
    <assign name="document.topicRSS" expr="'opinion'"/>
```

```
          <audio expr="document.topicRSS + '.wav'">
          <value expr="document.topicRSS"/>
          </audio>
    <goto next="#Headlines"/>
  </block>
</form>
<form id="migrantes_tema">
  <block>
    <prompt>Tu has elegido </prompt>
    <assign name="document.topicRSS" expr="'migrantes_tema'"/>
          <audio expr="document.topicRSS + '.wav'">
          <value expr="document.topicRSS"/>
          </audio>
    <goto next="#Headlines"/>
  </block>
</form>
<form id="deportes">
  <block>
    <prompt>Tu has elegido </prompt>
    <assign name="document.topicRSS" expr="'deportes'"/>
          <audio expr="document.topicRSS + '.wav'">
          <value expr="document.topicRSS"/>
          </audio>
    <goto next="#Headlines"/>
  </block>
</form>
<form id="Sucesos">
  <block>
    <prompt>Tu has elegido </prompt>
    <assign name="document.topicRSS" expr="'sucesos'"/>
          <audio expr="document.topicRSS + '.wav'">
          <value expr="document.topicRSS"/>
          </audio>
    <goto next="#Headlines"/>
  </block>
</form>


<form id="ivrorigin">
  <block>
    <prompt>Ahora Sera redireccionado al I Ve eRe</prompt>
    <goto next="IVR.vxml"/>
  </block>
</form>


 <form id="Headlines">
           <var name="num" expr="0"/>
```

```
        <block name="fetchData">

            <data name="fetchRSS" srcexpr="'http://servicios.eluniverso.com/' +
'rss/' + document.topicRSS + '.xml'"/>
            <assign                                    name="document.dataRSS"
expr="fetchRSS.documentElement"/>
            <assign                                  name="document.resultCount"
expr="getCount(fetchRSS)"/>
            <assign                          name="document.descriptionArrayRSS"
expr="assignArray(fetchRSS, 'description', 'item')"/>
            <assign                          name="document.descriptionArrayRSS"
expr="document.descriptionArrayRSS.slice(1, -1);"/>
            <assign                                  name="document.titleArrayRSS"
expr="assignArray(fetchRSS, 'title', 'item')"/>
            <assign                                  name="document.titleArrayRSS"
expr="document.titleArrayRSS.slice(1, -1);"/>

                        Tenemos
                <audio expr="document.resultCount + '.wav'">
                <value expr="document.resultCount"/>
                </audio>
                        artículos sobre
                <audio expr="document.topicRSS + '.wav'">
                <value expr="document.topicRSS"/>
                <break size="small"/>
                </audio>
                        Todo esto es cortesía del universo punto com

                <foreach item="article" array="document.titleArrayRSS">

                        <prompt>
                                <value expr="article"/>
                                <break size="medium"/>
                        </prompt>


                        <prompt>
                                <value
expr="document.descriptionArrayRSS[num]"/>
                                <break size="medium"/>
                        </prompt>

                        <assign name="num" expr="Add1(num)"/>



                </foreach>
```

```
                <prompt bargein="true">
                        Ya no hay mas artículos disponibles sobre el tema
                <audio expr="document.topicRSS + '.wav'">
                <value expr="document.topicRSS"/>
                <break size="small"/>
                </audio>

                        Para mayor información por favor visite w w w punto
                         el universo punto com

                        <break size="small"/>

                        Ahora retornaremos al menú principal, siéntase libre
                         de escoger otro tema o de colgar cuando le parezca.
                </prompt>

                <goto next="eluniverso.vxml"/>

        </block>

 </form>


</vxml>
```

"cine.vxml"

Este archivo permite al usuario conocer sobre la información concerniente a actores, actrices, detalles de bandas sonoras, entre otros. Al igual que el documento vxml correspondiente a El Universo, se puede elegir cada una de las opciones, con la posibilidad de volver a escuchar sobre algunas de las noticias o volver al menú principal si el usuario lo requiere. Su contenido es el siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>

<vxml version="2.1" xmlns:voxeo="http://community.voxeo.com/xmlns/vxml">
```

```
<script src="parseRSSData.js" fetchtimeout="15s"/>
 <property name="confidencelevel" value=".35"/>
 <property name="sensitivity" value="0.35"/>

 <var name="topicRSS"/>
 <var name="dataRSS"/>
 <var name="descriptionArrayRSS"/>
 <var name="fetchRSS"/>
 <var name="resultCount"/>
 <var name="titleArrayRSS"/>
<form id="Intro">

  <block>
   <prompt bargein="true">
        Usted se esta conectando a w   w   w punto cinissimo punto com,
        Ahora obtendra información al día sobre el mundo del cine.

   </prompt>
   <goto next="#Topic"/>
 </block>
</form>


<menu id="Topic" dtmf="true">
 <property name="inputmodes" value="dtmf"/>
 <prompt>
        Por favor digite un numero de su telefono para comenzar.
  <enumerate>

         Para        <value expr="_prompt"/>, digite <value expr="_dtmf"/>.
  </enumerate>
 </prompt>
<choice next="#actores">
        Información sobre actores
 </choice>
<choice next="#actrices">
        Información sobre actrices
 </choice>
<choice next="#bandas-sonoras">
        Detalles sobre las bandas sonoras
 </choice>
<choice next="#cortometrajes">
        noticias sobre cortometrajes
 </choice>
<choice next="#libros">
        Escuchar acerca de libros entorno al cine
 </choice>
 <choice next="#ivrorigin">
```

```
  Volver al I Ve eRe
 </choice>
</menu>

<form id="actores">
  <block>
    <prompt>Su elección fue </prompt>
    <assign name="document.topicRSS" expr="'actores'"/>
          <audio expr="document.topicRSS + '.wav'">
          <value expr="document.topicRSS"/>
          </audio>
    <goto next="#Headlines"/>
  </block>
</form>
<form id="actrices">
  <block>
    <prompt>Su elección fue </prompt>
    <assign name="document.topicRSS" expr="'actrices'"/>
          <audio expr="document.topicRSS + '.wav'">
          <value expr="document.topicRSS"/>
          </audio>
    <goto next="#Headlines"/>
  </block>
</form>
<form id="bandas-sonoras">
  <block>
    <prompt>Su elección fue </prompt>
    <assign name="document.topicRSS" expr="'bandas-sonoras'"/>
          <audio expr="document.topicRSS + '.wav'">
          <value expr="document.topicRSS"/>
          </audio>
    <goto next="#Headlines"/>
  </block>
</form>
<form id="cortometrajes">
  <block>
    <prompt>Su elección fue </prompt>
    <assign name="document.topicRSS" expr="'cortometrajes'"/>
          <audio expr="document.topicRSS + '.wav'">
          <value expr="document.topicRSS"/>
          </audio>
    <goto next="#Headlines"/>
  </block>
</form>
<form id="libros">
  <block>
    <prompt>Su elección fue </prompt>
    <assign name="document.topicRSS" expr="'libros'"/>
```

```
        <audio expr="document.topicRSS + '.wav'">
        <value expr="document.topicRSS"/>
        </audio>
   <goto next="#Headlines"/>
  </block>
</form>

<form id="ivrorigin">
  <block>
    <prompt>Ahora Sera redireccionado al I Ve eRe</prompt>
    <goto next="IVR.vxml"/>
  </block>
</form>


 <form id="Headlines">
        <block name="fetchData">
          <var name="num" expr="0"/>

            <data  name="fetchRSS"  srcexpr="'http://www.cinissimo.com/'  +
'category/' + document.topicRSS + '/feed/'"/>
            <assign                              name="document.resultCount"
expr="getCount(fetchRSS)"/>
            <assign                        name="document.descriptionArrayRSS"
expr="assignArray(fetchRSS, 'description', 'item')"/>
            <assign                        name="document.descriptionArrayRSS"
expr="document.descriptionArrayRSS.slice(1, -1);"/>
            <assign                              name="document.titleArrayRSS"
expr="assignArray(fetchRSS, 'title', 'item')"/>
            <assign                              name="document.titleArrayRSS"
expr="document.titleArrayRSS.slice(1, -1);"/>


                    Tenemos
                <audio expr="document.resultCount + '.wav'">
                <value expr="document.resultCount"/>
                </audio>
                    artículos sobre
                <audio expr="document.topicRSS + '.wav'">
                <value expr="document.topicRSS"/>
                <break size="small"/>
                </audio>
                    Cortesía de  cinissimo punto com


                <foreach item="article" array="document.titleArrayRSS">

                    <prompt>
```

```
                    <value expr="article"/>
                    <break size="medium"/>
                </prompt>

                <prompt>
                    <value
expr="document.descriptionArrayRSS[num]"/>
                    <break size="medium"/>
                </prompt>

                <assign name="num" expr="Add1(num)"/>


            </foreach>

            <prompt bargein="true">
                No hay mas artículos disponibles sobre el tema
            <audio expr="document.topicRSS + '.wav'">
            <value expr="document.topicRSS"/>
            <break size="small"/>
            </audio>

                Para mayor información por favor visite w  w  w
punto cinissimo punto com

                <break size="small"/>

                Ahora retornaremos al menú inicial, siéntase libre de
escoger otro tema o de colgar cuando desee.
            </prompt>

            <goto next="cine.vxml"/>

        </block>

 </form>


</vxml>
```

“blogespol.vxml”

En este archivo permite revisar información sobre un blog de la ESPOL el cual posee noticias de la ESPOL, sobre tecnología, noticias de la IEEE. En este documento como en los anteriores se puede elegir cada una de las opciones, además de la posibilidad de volver a escuchar sobre algunas de las noticias o volver al menú principal si el usuario lo requiere. Su contenido es el siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>

<vxml version="2.1" xmlns:voxeo="http://community.voxeo.com/xmlns/vxml">

 <script src="parseRSSData.js" fetchtimeout="15s"/>
  <property name="confidencelevel" value=".35"/>
  <property name="sensitivity" value="0.35"/>

  <var name="topicRSS"/>
  <var name="dataRSS"/>
  <var name="descriptionArrayRSS"/>
  <var name="fetchRSS"/>
  <var name="resultCount"/>
  <var name="titleArrayRSS"/>
 <form id="Intro">

   <block>
    <prompt bargein="true">
          Bienvenido al blog Espol, un sitio hecho por sus estudiantes.

    </prompt>
    <goto next="#Topic"/>
  </block>
 </form>


<menu id="Topic" dtmf="true">
 <property name="inputmodes" value="dtmf"/>
 <prompt>
          Para comenzar, elije uno de los blog que te presentamos.
   <enumerate>
```

```
          Para        <value expr="_prompt"/>, presione <value expr="_dtmf"/>.
  </enumerate>
 </prompt>
<choice next="#pump">
  entretenimiento
</choice>
<choice next="#comenpump">
  Oír comentario sobre entretenimiento
</choice>
<choice next="#tecnologia">
  tecnología en la Espol
</choice>
<choice next="#noticias">
  Noticias ESPOL
</choice>
<choice next="#ieee">
  Noticias de la I triple E
</choice>
<choice next="#chicas">
  cosas femeninas
 </choice>
 <choice next="#ivrorigin">
  Volver al I Ve eRe
 </choice>
</menu>

<form id="pump">
  <block>
    <prompt>Tu has elegido entretenimiento </prompt>
    <assign name="document.topicRSS" expr="'ccazanas/feed/'"/>
    <goto next="#Headlines"/>
  </block>
</form>

<form id="comenpump">
  <block>
    <prompt>Tu has elegido oír comentario sobre entretenimiento</prompt>
    <assign name="document.topicRSS" expr="'ccazanas/comments/feed/'"/>
    <goto next="#Headlines"/>
  </block>
</form>

<form id="tecnologia">
  <block>
    <prompt>Tu has elegido tecnología en la Espol </prompt>
    <assign name="document.topicRSS" expr="'tecnoblog/feed/'"/>
    <goto next="#Headlines"/>
```

```
  </block>
</form>

<form id="noticias">
  <block>
    <prompt>Tu has elegido Noticias ESPOL </prompt>
    <assign name="document.topicRSS" expr="'feed/rss/'"/>
    <goto next="#Headlines"/>
  </block>
</form>

<form id="ieee">
  <block>
    <prompt>Tu has elegido Noticias de la I triple E </prompt>
    <assign name="document.topicRSS" expr="'dvillalb/?feed=rss2'"/>
    <goto next="#Headlines"/>
  </block>
</form>

<form id="chicas">
  <block>
    <prompt>Tu has elegido cosas femeninas </prompt>
    <assign name="document.topicRSS" expr="'diversin100/feed/'"/>
    <goto next="#Headlines"/>
  </block>
</form>

<form id="ivrorigin">
  <block>
    <prompt>Ahora Sera redireccionado al I Ve eRe</prompt>
    <goto next="IVR.vxml"/>
  </block>
</form>

 <form id="Headlines">
          <block name="fetchData">
            <var name="num" expr="0"/>

             <data name="fetchRSS" srcexpr="'http://blog.espol.edu.ec/' +
document.topicRSS"/>
             <assign name="document.dataRSS"
expr="fetchRSS.documentElement"/>
             <assign name="document.resultCount"
expr="getCount(fetchRSS)"/>
             <assign name="document.descriptionArrayRSS"
expr="assignArray(fetchRSS, 'description', 'item')"/>
             <assign name="document.descriptionArrayRSS"
expr="document.descriptionArrayRSS.slice(1, -1);"/>
```

```
        <assign name="document.titleArrayRSS"
expr="assignArray(fetchRSS, 'title', 'item')"/>
        <assign name="document.titleArrayRSS"
expr="document.titleArrayRSS.slice(1, -1);"/>


                Tenemos
            <audio expr="document.resultCount + '.wav'">
            <value expr="document.resultCount"/>
            </audio>
                artículos sobre el blog que seleccionaste

            <break size="small"/>

                Toda la información proviene de alumnos de la ESPOL.

            <foreach item="article" array="document.titleArrayRSS">

                <prompt>
                    <value expr="article"/>
                    <break size="medium"/>
                </prompt>

                <prompt>
                    <value
expr="document.descriptionArrayRSS[num]"/>
                    <break size="medium"/>
                </prompt>

                <assign name="num" expr="Add1(num)"/>


            </foreach>


            <prompt bargein="true">
                Ya no hay mas artículos disponibles sobre el tema que
escogiste

            <break size="small"/>


                Para mayor información por favor visite w w w blog
punto espol punto edu punto ec

                <break size="small"/>
```

```
                    Ahora retornaremos al menú principal, siéntase libre de
escoger otro tema o de colgar cuando le parezca.
                </prompt>

                <goto next="blogespol.vxml"/>

        </block>

 </form>


</vxml>
```

"espana.vxml"

Por medio de este documento podemos tener información sobre noticias internacionales y de España. El usuario pude seleccionar una de las opciones, con la posibilidad de volver a escuchar sobre algunas de las noticias o volver al menú principal si el usuario lo requiere. Su contenido es el siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>

<vxml version="2.1" xmlns:voxeo="http://community.voxeo.com/xmlns/vxml">

 <script src="parseRSSData.js" fetchtimeout="15s"/>
  <property name="confidencelevel" value=".35"/>
  <property name="sensitivity" value="0.35"/>

  <var name="topicRSS"/>
  <var name="dataRSS"/>
  <var name="descriptionArrayRSS"/>
  <var name="fetchRSS"/>
  <var name="resultCount"/>
  <var name="titleArrayRSS"/>

 <form id="Intro">
```

```
  <block>
   <prompt bargein="true">
        Bienvenido al portal de información, ahora usted tendrá información
        acerca de españa cortesía del diario español LEVANTE.

   </prompt>
   <goto next="#Topic"/>
  </block>
 </form>


<menu id="Topic" dtmf="true">
 <property name="inputmodes" value="dtmf"/>
 <prompt>
        Para comenzar, escoja uno de los temas que le presentamos.
  <enumerate>

         Para elegir       <value expr="_prompt"/>, presione <value
expr="_dtmf"/>.
  </enumerate>
 </prompt>
<choice next="#espania">
        información sobre españa
</choice>
<choice next="#internacionales">
         noticias internacionales
</choice>
<choice next="#sucesos">
        Sucesos
</choice>
<choice next="#fiestas">
        Fiestas y espectáculos
</choice>
<choice next="#ense">
        Enseñanza
</choice>
<choice next="#gente">
        Gente
 </choice>
 <choice next="#ivrorigin">
  Volver al I Ve eRe
 </choice>
</menu>

<form id="espania">
  <block>
   <prompt>Tu has elegido información sobre españa </prompt>
   <assign name="document.topicRSS" expr="'6'"/>
```

```
    <goto next="#Headlines"/>
  </block>
</form>

<form id="internacionales">
  <block>
    <prompt>Tu has elegido noticias internacionales</prompt>
    <assign name="document.topicRSS" expr="'7'"/>
    <goto next="#Headlines"/>
  </block>
</form>

<form id="sucesos">
  <block>
    <prompt>Tu has elegido Sucesos </prompt>
    <assign name="document.topicRSS" expr="'10'"/>
    <goto next="#Headlines"/>
  </block>
</form>

<form id="fiestas">
  <block>
    <prompt>Tu has elegido Fiestas y espectáculos </prompt>
    <assign name="document.topicRSS" expr="'24'"/>
    <goto next="#Headlines"/>
  </block>
</form>

<form id="ense">
  <block>
    <prompt>Tu has elegido enseñanza </prompt>
    <assign name="document.topicRSS" expr="'22'"/>
    <goto next="#Headlines"/>
  </block>
</form>

<form id="gente">
  <block>
    <prompt>Tu has elegido Gente </prompt>
    <assign name="document.topicRSS" expr="'41'"/>
    <goto next="#Headlines"/>
  </block>
</form>

<form id="ivrorigin">
  <block>
    <prompt>Ahora Sera redireccionado al I Ve eRe</prompt>
    <goto next="IVR.vxml"/>
```

```
  </block>
</form>

 <form id="Headlines">
          <block name="fetchData">
            <var name="num" expr="0"/>

             <data name="fetchRSS" srcexpr="'http://www.levante-emv.com/' +
'elementosInt/' + 'rss/' + document.topicRSS"/>
             <assign name="document.dataRSS"
expr="fetchRSS.documentElement"/>
             <assign name="document.resultCount"
expr="getCount(fetchRSS)"/>
             <assign name="document.descriptionArrayRSS"
expr="assignArray(fetchRSS, 'description', 'item')"/>
             <assign name="document.descriptionArrayRSS"
expr="document.descriptionArrayRSS.slice(1, -1);"/>
             <assign name="document.titleArrayRSS"
expr="assignArray(fetchRSS, 'title', 'item')"/>
             <assign name="document.titleArrayRSS"
expr="document.titleArrayRSS.slice(1, -1);"/>


                          Tenemos
                    <audio expr="document.resultCount + '.wav'">
                    <value expr="document.resultCount"/>
                    </audio>
                          artículos sobre el tema seleccionado

                    <break size="small"/>

                          Toda la información es obtenida de levante e m v punto
com

                    <foreach item="article" array="document.titleArrayRSS">

                          <prompt>
                                <value expr="article"/>
                                <break size="medium"/>
                          </prompt>

                          <prompt>
                                <value
expr="document.descriptionArrayRSS[num]"/>
                                <break size="medium"/>
                          </prompt>

                          <assign name="num" expr="Add1(num)"/>
```

```
                </foreach>

                <prompt bargein="true">
                        No hay mas artículos disponibles

                <break size="small"/>


                        Para mayor información por favor visite w w w levante
raya e m  v punto com

                        <break size="small"/>

                        Ahora retornaremos al menú principal, siéntase libre de
escoger otro tema o de colgar cuando le parezca.
                </prompt>

                <goto next="espana.vxml"/>

        </block>

 </form>


</vxml>
```

Dentro de todos los documentos vxml utilizados, se encuentra un script con el nombre de parseRSSData.js, el cual permite obtener la información necesaria para realizar el TTS, dicho script realiza lo siguiente:

Primero obtiene la cantidad de artículos que se han publicado mediante el formato RSS, luego de esto una vez obtenido dicha cantidad se almacena el contenido de cada artículo en un arreglo, una vez realizado este arreglo ya obtenemos la información, la cual nos va a permitir realizar TTS de cada uno

de los contenidos de los artículos publicados en las páginas con formato RSS.

El contenido del script parseRSSData.js es el siguiente:

```
function getCount(d)  {
      var x=d.getElementsByTagName("item");

   return x.length;
   }

      function assignArray(d, n, r) {

        var j=(d.getElementsByTagName(r).length + 2);
        var DATAarray;

           DATAarray = new Array();
           for(var i = 0; i < j; i++) {



           try {
      DATAarray[i] = d.getElementsByTagName(n).item(i).firstChild.data;
           }

           catch(e)
           {

            DATAarray[i] = '';
           }

     }
        return DATAarray;
}


function Add1(n){
   n=n+1;
return n;
}
```

Dentro del CLI de Asterisk, se utilizará el comando AGI DEBUG ON que permitirá revisar los canales AGI y el comportamiento que estos presentan en el momento en el que se realice la interacción de Asterisk con VoiceGlue, lo que se observa en el CLI de Asterisk será lo siguiente:

Cuando el usuario realiza la llamada al IVR:

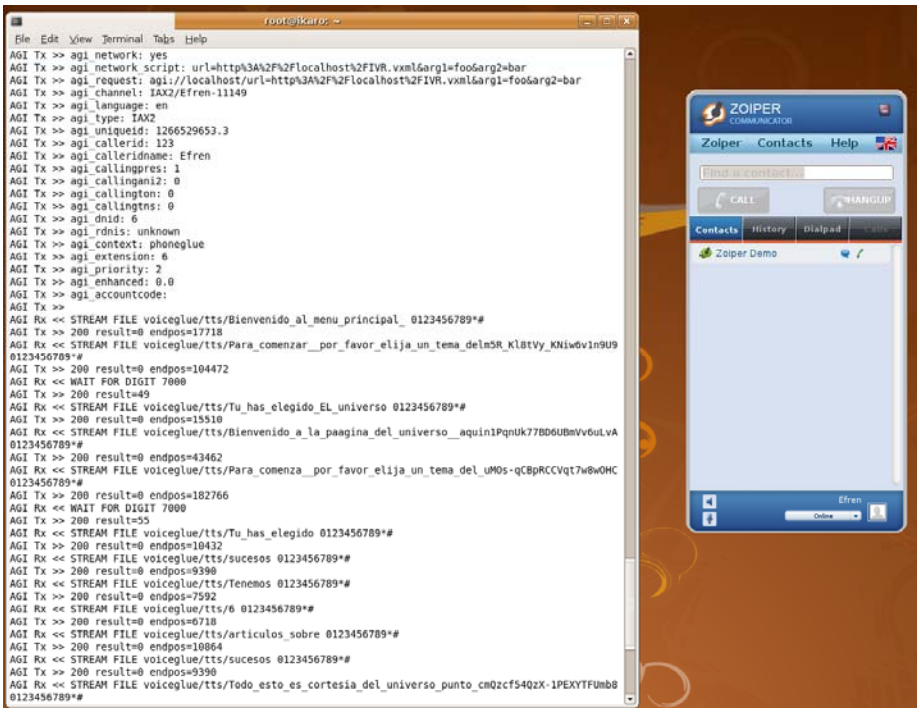

Fig. 4.5 CLI de Asterisk – Interacción Usuario-IVR

4.4. PRUEBAS Y MEDICIONES

4.4.1 Uso de Memoria de Asterisk-Voiceglue

La memoria empleada por el sistema tiene un uso promediado de 240Mb, esto es sin acceder a ninguna opción del IVR. Una vez que se accede al IVR y sus opciones la carga de Memoria sube como se muestra a continuación en la Tabla VII.

Tabla VII Uso de Memoria

Página accedida	Usado [Mb]	Libre [Mb]	Total [Mb]
IVR Principal	488	2543	3031
Universo Portada	475	2556	3031
Universo Política	468	2563	3031
Levante Portada	492	2539	3031
Levante Sucesos	496	2535	3031

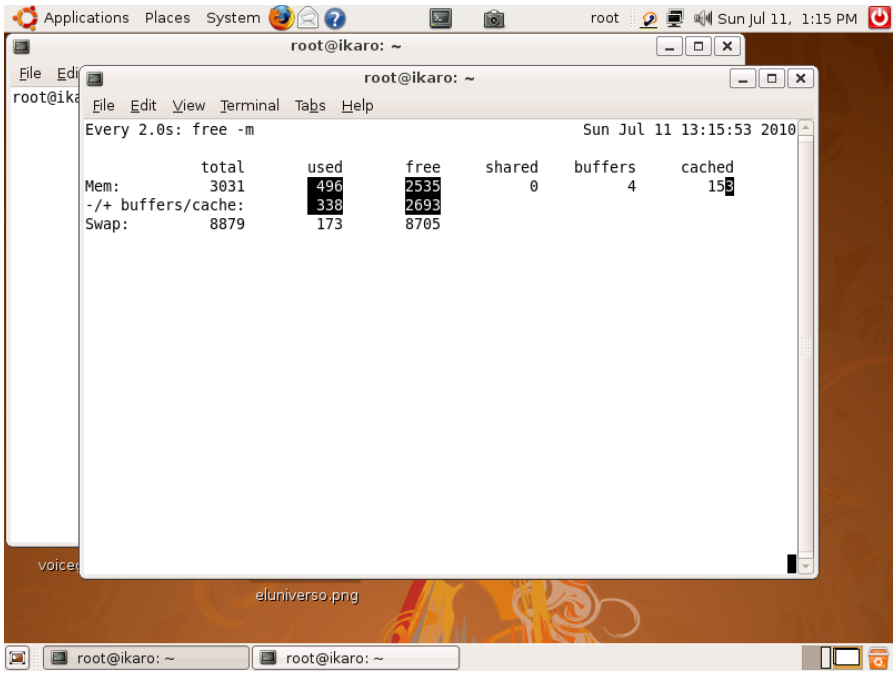

Fig. 4.6 Memoria IVR principal

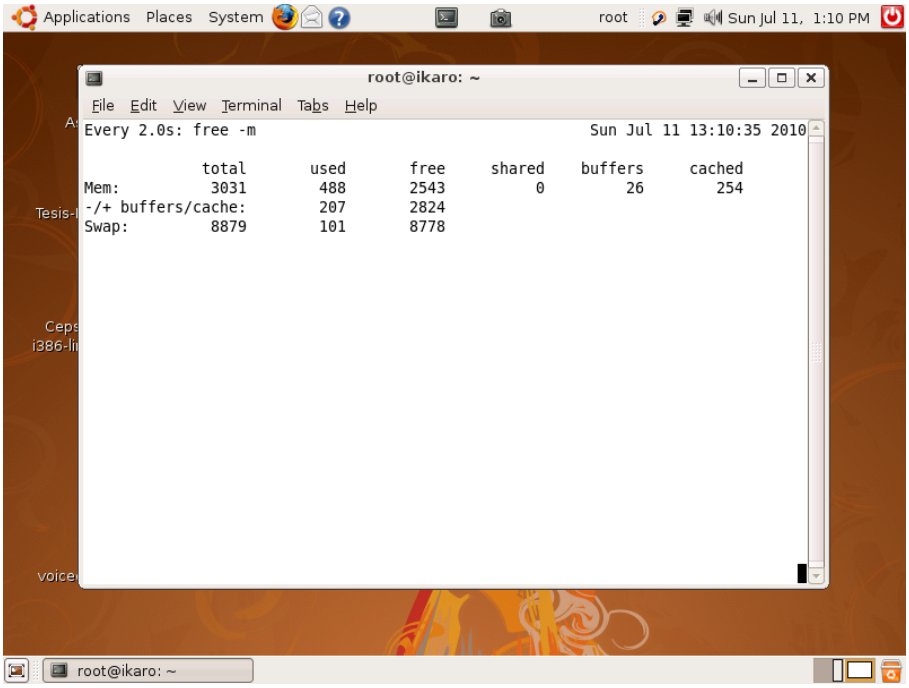

Fig. 4.7 Memoria Levante Sucesos

4.4.2 Uso de Ancho de Banda de Asterisk-Voiceglue

El ancho de banda que el sistema ocupa lo podemos observar usando un pequeño programa ejecutado desde consola llamado bwm-ng y el valor utilizado que nos indica es de aproximadamente 15kbps. La Tabla X nos indica el uso del ancho de Banda por parte del IVR y sus opciones:

Tabla VIII Uso de Ancho de Banda

Página accedida	Rx [Kbps]	Tx [Kbps]	Total [Kbps]
Universo Portada	13.04	9.71	22.75
Universo Política	9.13	6.35	15.48
Levante Portada	10.07	6.47	16.54
Levante Sucesos	9.31	6.35	15.66

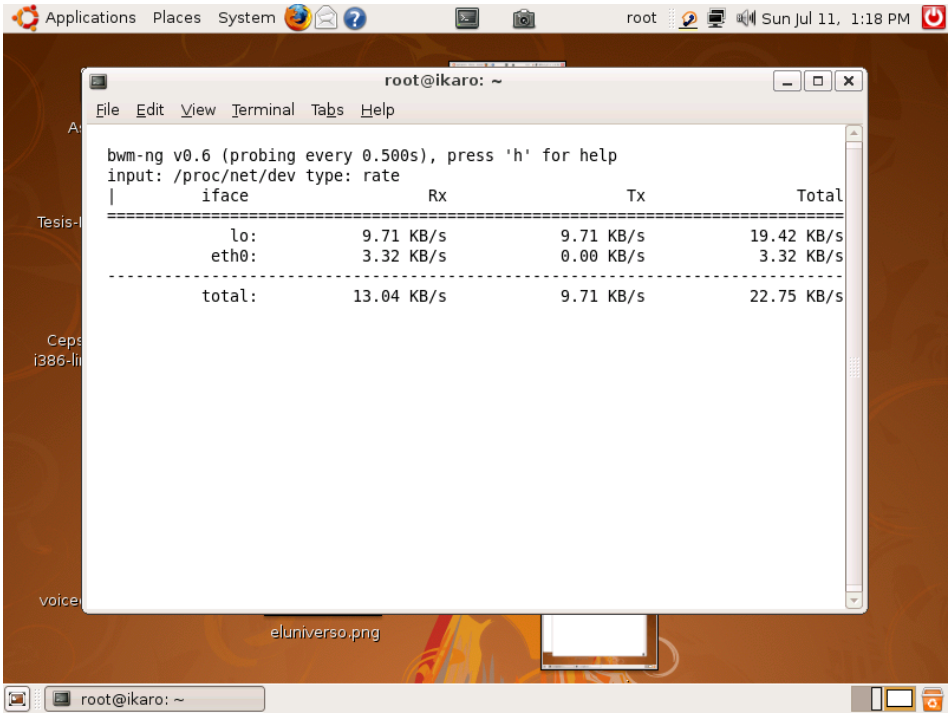

Fig. 4.8 Ancho de Banda UNIVERSO portada

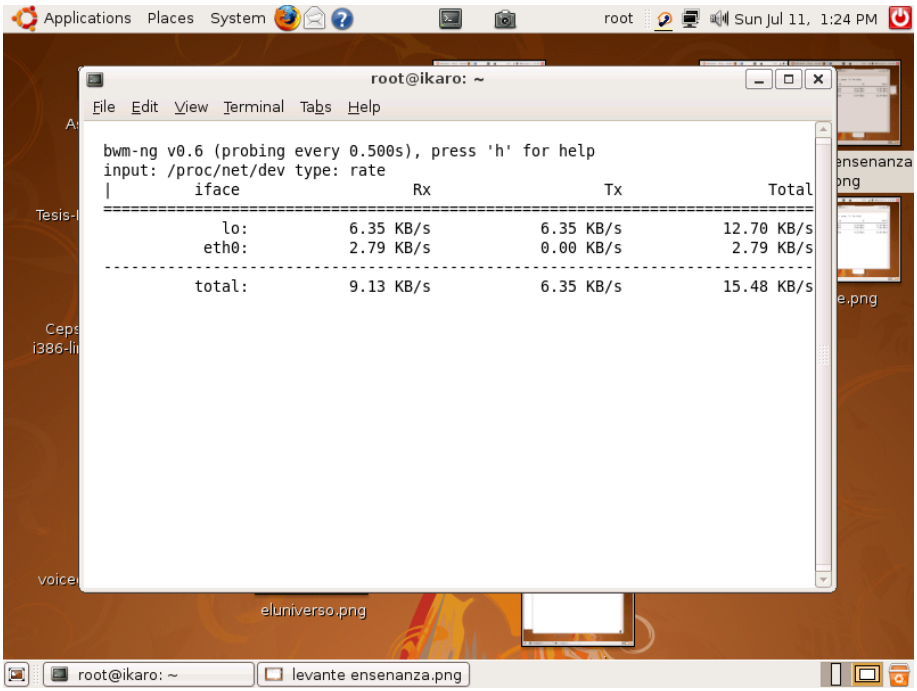

Fig. 4.9 Ancho de Banda UNIVERSO Política

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

1) El lenguaje extendido VXML se encuentra actualmente en auge, esto debido a que ha tenido grandes desarrollos durante los primeros años en los que fue creado la organización W3C, pero los desarrollos que se han dado en los últimos tiempos han sido muy cerrados con la existencia de poca disponibilidad y costos para las mejores aplicaciones y servicios como el caso de reconocimiento de voz **ASR** (Automatic Speech Recognition) que es desarrollado por pocas organizaciones y que posee un precio en las licencias, vale la pena aclarar que VoiceXML no realiza reconocimiento de voz, su función es la de interactuar con el software encargado de esto, por lo tanto VoiceXML manda a reconocer voz al software y este la traduce a texto según las especificaciones del documento VoiceXML.

2) VXML debería ser estudiado como mecanismo tecnológico para las telecomunicaciones. Consideramos que es un campo que se volverá fructífero si se lograse resolver la dificultad de las restricciones de los formatos XML que hemos empleado, ya que actualmente los estándares no son empleados de forma correcta. Este tipo de formato es ampliamente utilizado por grandes empresas de la información, las cuales presentan noticias que se van actualizando de forma constante, por medio de esto los usuarios pueden estar al tanto de los sucesos de forma prácticamente instantánea y utilizando nuestro servicio de TTS la información puede ser receptadas por todas las personas incluyendo algunas con capacidades especiales como es el caso de personas con deficiencia visual, para los cuales este método de acceder a la información sería el más eficiente.

3) El uso de TTS dentro de Asterisk no significa mucha carga de procesamiento, pero pese a que en nuestras pruebas no se produjeron errores por demanda múltiple podemos concluir por observación que el sistema es proclive a fallos con una carga excesiva.

4) Por observación podemos concluir que el sistema no requiere un extenso ancho de banda para funcionar eficientemente, por el contrario el uso de memoria estará condicionado a la cantidad de información que exista en una página web al momento de ser convertido de texto a audio por el servidor TTS. En otras palabras si existe una gran cantidad de artículos por leer(mayor a 20) el uso de memoria se incrementará hasta que el servidor TTS procese la conversión de todos estos artículos a audio.

Recomendaciones

1) Emplear un Sistema Operativo basado en Debian debido a la ventaja de que existe una mejor resolución de dependencias y mejores repositorios para la instalación de los requerimientos de VoiceGlue, en nuestro caso Ubuntu 8.04 fue nuestra mejor opción ya que presentaba mayor facilidad al momento de instalación.

2) Si se desea leer una pagina web a través de Asterisk usando VoiceGlue, se debe verificar el formato de la página web utilizando validadores como por ejemplo [21] y [22] que son los más apropiados debido a que tienen la fiabilidad de ser organizaciones claramente conocidas

3) La necesidad de un buen servidor debido a que los recursos utilizados pueden saturarlo y provocar que el sistema colapse. Es necesario que dicho servidor sea montado de forma física y no de forma virtual, ya que un servidor o maquina virtual consume la mayoría de recursos. La recomendación basados en nuestro servidor sería:

Disco Duro: 20 Gb.

Memoria RAM: Dependiendo de la carga de transacciones del servidor seria recomendable un mínimo de 1Gb.

Procesador: Pentium 4 a 3 GHz

Ancho de Banda: 512 Kbps dedicados exclusivamente a Asterisk.

4) Un posible trabajo a futuro basándose en ASR, sería la implementación de un sistema de respuesta inteligente utilizando técnicas de reconocimiento biométrico de voz, que dependiendo de la programación le permitiría al sistema reconocer diferente tipos de voces y acentos dándole la capacidad de identificar a un individuo en especial, así como la posibilidad de adaptarse a otros tipos de habla. Las aéreas de aplicación de este sistema son variadas, tales como: atención a personas con capacidades especiales, servicios de información automática, sistemas de seguridad, interacción con computadoras para propósitos específicos.

GLOSARIO DE TERMINOS

IVR: Son las siglas de Interactive Voice Response, que se traduce del inglés como Respuesta de Voz Interactiva. También se utiliza el término VRU (Voice Response Unit). Consiste en un sistema telefónico que es capaz de recibir una llamada e interactuar con el humano a través de grabaciones de voz y el reconocimiento de respuestas simples, como "sí", "no" u otras.

DIALPLAN: Un dialplan es la configuración que permite determinar el tratamiento que debe darse a un número discado.

VOIP: Voz sobre Protocolo de Internet, también llamado Voz IP, VozIP, VoIP (por sus siglas en inglés), es un grupo de recursos que hacen posible que la señal de voz viaje a través de Internet empleando un protocolo IP (Internet Protocol).

OPEN SOURCE: Código abierto (en inglés open source) es el término con el que se conoce al software distribuido y desarrollado libremente. El código abierto tiene un punto de vista más orientado a los beneficios prácticos de compartir el código.

PBX: Es cualquier central telefónica conectada directamente a la red pública de teléfono por medio de líneas troncales para gestionar, además de las llamadas internas, las entrantes y/o salientes con autonomía sobre cualquier otra central telefónica.

XML: XML son las siglas de Extensible Markup Language, una especificación/lenguaje de programación desarrollada por el W3C. XML es una versión de SGML, diseñado especialmente para los documentos de la web. Permite que los diseñadores creen sus propias etiquetas, permitiendo la definición,

transmisión, validación e interpretación de datos entre aplicaciones y entre organizaciones.

VXML: VoiceXML es un lenguaje de etiquetado que permite crear diálogos con los que se puede interactuar escuchando comandos hablados, controlables a través de entradas de voz. VoiceXML se encarga de convertir habla en texto y para ello utiliza, entre otros mecanismos; SRGS (Gramática de Reconocimiento del Habla).

TTS: Text-to-Speech, convierte el texto normal del lenguaje en palabras.

GRAMMAR: Especifica una lista de vocabulario admisible para que el usuario seleccione en orden la forma de interactuar con la aplicación de VoiceXML.

DTMF: (Dual Tone Multifrecuency) Multifrecuencia de doble tono. Tonos en diferentes hertz que utiliza una telefonía para marcar números. Cada número u opción del teléfono tiene su tono que es identificado en la telefonía.

SRGS: Speech Recognition Grammar Specification por sus siglas en ingles, tiene como función principal, permitir que una aplicación de voz indique a un reconocedor, qué es lo que tiene que escuchar, es decir, palabras, modelos en los que estas palabras surgen, lenguaje hablado de cada palabra, etc.

W3C: El World Wide Web Consortium, abreviado W3C, es un consorcio internacional que produce recomendaciones para la World Wide Web.

AGI: El AGI (Asterisk Gateway Interface) permite extender las funcionalidades de Asterisk mediante el uso de lenguajes de programación. El AGI sirve de enlace entre las aplicaciones externas y el núcleo de Asterisk.

RSS: RSS es una familia de formatos de fuentes web codificados en XML. Se utiliza para suministrar a suscriptores de información actualizada frecuentemente. El formato permite distribuir contenido sin necesidad de un navegador, utilizando un software diseñado para leer estos contenidos RSS.

ASR: Reconocimiento automático del habla en español y Authomatic Speech Recognition en ingless, (ASR) es una tecnología que permite a una computadora identificar las palabras que una persona habla a un micrófono o teléfono. El "santo grial" de la investigación ASR es permitir que una computadora para reconozca en tiempo real con una precisión del 100% todas las palabras que habla cualquier persona, independientemente del tamaño del vocabulario, el ruido, las características del altavoz y el acento.

BIBLIOGRAFIA

[1] W3C, Voice Extensible MarkupLanguage (VoiceXML) Version 2.0-Overview, http://www.w3.org/TR/voicexml20/,16 Marzo 2004

[2] VoiceGlue, Voiceglue Architecture Diagram, http://voiceglue.org/wiki/doku.php?id=voiceglue_architecture_diagram, 14 Enero 2009

[3] W3C, Burnett, Voice Extensible Markup Language (VoiceXML) Version 2.0, http://www.w3.org/TR/voicexml20/, 16 Marzo 2004

[4] VoiceGlue, VoiceGlue Release Notes, http://voiceglue.org/wiki/doku.php?id=voiceglue_0.11_release_notes, 27 Agosto 2009

[5] Kimberlee Kemble, Voice-Enabling Your Web Sites, http://www.ibm.com/developerworks/websphere/library/techarticles/0111_kemble/0111_kemble.html, 30 Noviembre 2001

[6] CompassTech, Qué es el IVR, http://www.compasstech.com.mx/ct-html/que_es_ivr.html, 12 Enero 2010

[7] Wikipedia, Interactive Voice Response, http://es.wikipedia.org/wiki/Interactive_Voice_Response, 21 Enero 2010

[8] IVoice, Diagrama IVR, http://www.ivoice.com.mx/images/ivr2.jpg, 3 Febrero 2010

[9] Voip-Info, How to build a Dialplan in Asterisk, http://www.voip-info.org/wiki/view/Asterisk+howto+dial+plan, 30 Noviembre 2008

[10] Ibarra Gorrotxategi, Introducción Asterisk – IVR en AEL2, http://www.saghul.net/blog/documentos-cc/eside-words-07/charla.pdf, 15 Enero 2010

[11] Wikitel, FXS y FXO, http://www.wikitel.info/wiki/FXS, 28 Enero 2010

[12] 3CX, Qué significan los términos FXS y FXO, http://www.3cx.es/voip-sip/fxs-fxo.php, 16 Enero 2010

[13] Pineda Ruiz, Morales, Desarrollo de un sistema de Información basado en VoiceXML, http://postgrados.ufg.edu.sv/doc/VoiceXML.pdf , 4 Febrero 2010

[14] Mundo-Contact, La Industria IVR se transforma: el rostro cambiante de las plataformas, http://www.mundo-contact.com/Acervo/IVR_se_transforma.pdf, 17 Enero 2010

[15] Falaschi, Esposito, Read my feed - from RSS to SIP, http://smile.ing.uniroma1.it/sapientel/uploads/ReadMyFeed/readmyfeed.pdf, 22 Enero 2010

[16] Wikipedia, RSS, http://es.wikipedia.org/wiki/RSS, 3 Febrero 2010

[17] VoiceGlue, VoiceGlue 0.11 Installation Instruction, http://voiceglue.org/wiki/doku.php?id=voiceglue_0.11_installation_instructions, 29 Enero 2010

[18] VoiceGlue, VoiceGlue 0.11 User Guide http://voiceglue.org/wiki/doku.php?id=voiceglue_0.11_user_guide, 10 Febrero 2010

[19] Gutierrez T, Reconocimiento de Voz y VXML, http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/gutierrez_t_a/capitulo2.pdf, 16 Enero 2010

[20] Ciencia Forense.cl, Revista Online de Criminalistica, http://www.cienciaforense.cl/csi/index2.php?option=com_content&do_pdf=1&id=35, 24 Enero 2010

[21] RSS BOARD, Validador de Formato RSS, http://www.rssboard.org/rss-specification, 10 Enero 2010

[22] W3C, Validador Formato RSS, http://validator.w3.org/, 15 Enero 2010