



INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

ESCOM



Trabajo Terminal:

“Arquitectura de Autentificación por Biométrico de Voz, basado en Ontologías”

Nº de registro:
20080012

Objetivo:

Analizar, diseñar y desarrollar una arquitectura de autentificación basada en dos aspectos principales: ontologías y el reconocimiento de un biométrico (voz) para garantizar con esto un margen de error más reducido y un mínimo de falsos positivos.

Presentan:

**Díaz Trejo Diana
Guzmán Ríos Rafael Ángel
Villanueva Chávez Joel**

Director:

M. en C. Chadwick Carreto Arellano

México D.F., a 10 de junio de 2009



INSTITUTO POLITÉCNICO NACIONAL ESCUELA SUPERIOR DE CÓMPUTO



ESCOM

Nº de registro: **20080012**

Serie: Amarilla

Junio de 2009

Reporte Técnico

“Arquitectura de Autentificación por Biométrico de Voz, basado en Ontologías”

**Díaz Trejo Diana [1]
Guzmán Ríos Rafael Angel [2]
Villanueva Chávez Joel [3]**

Resumen:

El presente es el reporte técnico que se elaboró para el trabajo terminal, incluye los fundamentos, diseño, análisis, pruebas y funcionamiento así como referencias bibliográficas.

Palabras Clave:
Identificación, Autentificación, VoXML, Ontología.

México D.F., a 10 de junio de 2009

Advertencia:

Este reporte contiene información desarrollada por la Escuela Superior de Cómputo del Instituto Politécnico Nacional a partir de datos y documentos con derechos de propiedad y por lo tanto su uso queda restringido a las aplicaciones que explícitamente se convengan

La aplicación no convenida exime a la Escuela de su responsabilidad técnica y da lugar a las consecuencias legales que para tal efecto se determinen.

Información adicional sobre este reporte final podrá obtenerse en el Departamento de Diseño Curricular y Metodología de la Escuela Superior de Cómputo del Instituto Politécnico Nacional, situada en: Av. Juan de Dios Bátiz s/n, Tel. 57-29-60-00 ext. 52021.

AGRADECIMIENTOS

Dedico estas tesis a mis padres, por haberme apoyado en este camino, por medio de sus consejos, cariño y educación.

En especial quiero agradecer a mi mamá por ser una mujer fuerte, por su esfuerzo, dedicación y sacrificios que sé que tuvo que realizar para sacar adelante a mi familia, gracias mamá por impulsarme y haberme dado lo mejor y motivarme a completar mi carrera de manera exitosa, soy afortunada de tener una madre como tú a mi lado, hoy acaba un ciclo en mi vida pero vendrán otros más que espero compartir contigo.

Agradezco a mi papá por confiar en mí y por el esfuerzo de levantarse cada mañana a llevarme a la escuela.

A mis amigos, que he tenido a lo largo de mi vida y que con su confianza, amistad y paciencia me han enseñado a no darme por vencida, gracias a mi amigo David por estar en momentos tan cruciales en mi vida brindándome su apoyo incondicional, por mostrarme el valor de la humildad, cariño y por tantas aventuras que vivimos juntos.

A Rafa y a Joel por su amistad y la oportunidad de trabajar con ellos.

A mis compañeros y maestros de la escuela por haberme brindado conocimiento y disciplina a lo largo de estos 4 años.

DIANA DÍAZ TREJO

AGRADECIMIENTOS...

Al PADRE, que me ha permitido darle gloria con la culminación de este ciclo de mi vida y a quien debo todo lo que tengo y lo que soy. Al que me ha educado y que con su ejemplo, dedicación, consejo y trabajo me ha enseñado el significado de la vida y su verdadero propósito.

A la MADRE, que con su amor ha protegido mi camino y que siempre está intercediendo por mí. A la que me ha enseñado el significado del verdadero amor y que ha llenado mi vida de bendiciones y ganas de salir adelante.

Al HOMBRE, que en su forma de hermano, amigo o compañero me ha ayudado con su buen ejemplo como un modelo a seguir y con sus errores, mostrándome el camino que he de evitar.

A la MUJER, que en su forma de hermana, amiga o compañera supo darme su amor, cariño y ternura y ha contribuido a elevar mi alma hacia Dios.

Al PROFESOR, que dedico su tiempo, conocimientos y su apoyo para que yo fuera una buena profesionista y poder servir así a mis hermanos...

A todos ellos GRACIAS!!!

RAFAEL ÁNGEL GUZMÁN RÍOS

AGRADECIMIENTOS

A mí MADRE que día a día con su amor, entrega y ejemplo de vida me mostró el significado del vivir con amor respeto y siempre dando una cara optimista ante toda situación, a aquella mujer que jamás me criticó y me supo entender, a aquel ángel que siempre ha intercedido e intercederá por mí, a aquella que maestra que me inculcó el amor por el trabajo y la excelencia, a aquella que jamás me ha dado la espalda y me supo educar con maestría y con infinito amor, a quien debo todo lo que hoy soy, a la única mujer que jamás me dejara de amar y a la que siempre he de estar eternamente agradecido.

A mí PADRE ejemplo de sapiencia, de rectitud, bondad, disciplina y justicia, por mostrarme siempre el sendero correcto, por siempre decirme la verdad tal cual es no importando cuan siniestra u horrible sea, por el amor que siempre me ha tenido y tendrá, al hombre que siempre me impulso al triunfo, que pese a las diferencia que tiene conmigo me apoya y da toda su confianza, al hombre que me inculcó rectitud y amplio criterio, al mejor padre del mundo, al hombre que me inculcó la serenidad paciencia el temple y el coraje, al hombre que me mostro el principio de la madurez, al hombre con el que siempre estaré en deuda.

A mis hermanos fuente de ternura apoyo, bondad, sinceridad, nobleza, sencillez, por siempre poder hablar con ellos con desenfado, por siempre oír mis penas alegrías y tristezas, por siempre brindarme un hombro en donde llorar, un momento para compartir, una experiencia que contar, un fracaso que superar, a ellos que día a día me mostraron el significado de la hermandad y del apoyo incondicional y del amor puro y verdadero a ellos que siempre han estado y estarán para mí, a ellos que siempre han sido y serán mi bendición.

A la única mujer que no compartiendo mi sangre supo amarme comprenderme y estimarme, a aquella mujer que ha sido fuente de inspiración y de luz en mi vida, a ella que pese a todas las dificultades no me ha abandonado y a compartido su camino junto al mío, a aquella que me dio la oportunidad de conocer el amor de pareja, a aquella que me hizo ver mis errores no importando cuan grandes fueran a aquella que dejara huella en mi corazón, muchas gracias.

A todos aquellos hombres y mujeres que bajo distintas figuras nombres y significados (amigos, familiares, maestros o extraños), que han dado lucidez, brillo y enseñanzas a mi vida a todos aquellos con los que pude compartir un poco de mí y ellos a su vez dejaron huella en mí por medio de sus diversos gestos de amor: ayuda, apoyo confianza, bondad cariño, respeto o bondad, a todos ellos mil gracias.

JOEL VILLANUEVA CHÁVEZ

CONTENIDO TEMATICO

OBJETIVOS	5
INTRODUCCIÓN	6
CAPÍTULO 1. ESTADO DEL ARTE.....	7
1.1 ANTECEDENTES	7
1.1.1 SISTEMAS DE IDENTIFICACIÓN, AUTENTIFICACIÓN Y CONTROL DE ACCESO.....	7
1.1.2 SISTEMAS DE AUTENTIFICACIÓN BIOMÉTRICA.....	8
1.1.3 VERIFICACIÓN DE VOZ.....	8
1.2 PROBLEMÁTICA	9
1.3 SOLUCIONES EXISTENTES	11
CAPÍTULO 2. ANÁLISIS	15
2.1 PROPUESTA DE SOLUCIÓN	15
2.2 ANÁLISIS DE REQUERIMIENTOS	16
2.2.1 REQUERIMIENTOS FUNCIONALES	16
2.2.2 REQUERIMIENTOS NO FUNCIONALES	17
2.3 ESTUDIO DE VIABILIDAD	19
2.3.1 VIABILIDAD ECONÓMICA	19
2.3.2 VIABILIDAD TÉCNICA	22
2.3.3 ALTERNATIVAS	22
2.4 MODELADO UML.....	23
2.4.1 MODELADO GENERAL DE LA ARQUITECTURA	23
2.4.2 MODELADO DE LA CAPA DE USUARIO	28
2.4.3 MODELADO DE LA CAPA DE NAVEGACIÓN.....	30
2.4.4 MODELADO DE LA CAPA ZONA DE INTERNET.....	32
2.4.5 MODELADO DE LA CAPA DE INTEGRACIÓN	34
2.4.6 MODELADO DE LA CAPA ONTOLOGICA	36
2.5 DISEÑO CONCEPTUAL DE BASE DE DATOS	38
2.6 DISEÑO FÍSICO DE BASE DE DATOS.....	38
CAPÍTULO 3. DISEÑO.....	39
3.1 MODELADO DE LA ARQUITECTURA.....	39
3.1.1 CAPA DE USUARIO.....	39
3.1.2 CAPA DE NAVEGACIÓN.....	39
3.1.3 CAPA ZONA DE INTERNET.....	40
3.1.4 CAPA DE INTEGRACIÓN	40

3.1.5 CAPA ONTOLOGICA	41
3.2 DISEÑO MODULAR	41
3.3 DISEÑO E/S	43
CAPÍTULO 4. DESARROLLO Y CODIFICACIÓN	44
4.1 CAPA DE USUARIO	44
4.2 CAPA DE NAVEGACIÓN	44
4.2.1 TÉCNOLOGIA EMPLEADA	44
4.2.2 CODIFICACIÓN	45
4.3 CAPA ZONA DE INTERNET	46
4.3.1 TÉCNOLOGIA EMPLEADA	46
4.3.2 CODIFICACIÓN	47
4.4 CAPA DE INTEGRACIÓN	49
4.4.1 TÉCNOLOGIA EMPLEADA	52
4.4.2 CODIFICACIÓN	54
4.5 CAPA ONTOLOGICA	88
4.5.1 TÉCNOLOGIA EMPLEADA	88
4.5.2 CODIFICACIÓN	90
CAPÍTULO 5. PRUEBAS, RESULTADOS, CONCLUSIONES Y TRABAJO A FUTURO	109
5.1 CAPA DE USUARIO	109
5.2 CAPA DE NAVEGACIÓN	109
5.3	113
CAPA ZONA DE INTERNET	114
5.4 CAPA DE INTEGRACIÓN	115
5.5 CAPA ONTOLOGICA	116
5.6 CONCLUSIONES	139
5.7 TRABAJO A FUTURO	140
GLOSARIO	141
REFERENCIAS	142

INDICE DE TABLAS

<i>Tabla 1. Soluciones existentes y Propuesta de solución</i>	11
<i>Tabla 2. Requerimientos Funcionales</i>	16
<i>Tabla 3. Requerimientos no Funcionales</i>	18
<i>Tabla 4. Comparativa Lenguajes</i>	20
<i>Tabla 5. Tipos de Proyectos Software</i>	20
<i>Tabla 6. Conductores de Coste</i>	20
<i>Tabla 7. Descripción de Actores en Diagrama de Casos de Uso</i>	24
<i>Tabla 8. Descripción de Casos de Uso de Arquitectura</i>	24
<i>Tabla 9. Descripción de Actividades de Arquitectura.....</i>	28
<i>Tabla 10. Descripción de Casos de Uso. Capa de Usuario</i>	29
<i>Tabla 11. Descripción de Secuencia. Capa Usuario.....</i>	30
<i>Tabla 12. Descripción de Actores. Capa de Navegación.....</i>	31
<i>Tabla 13. Descripción de Casos de Uso. Capa de Navegación</i>	31
<i>Tabla 14. Descripción diagrama de Secuencias. Capa de Navegación.....</i>	32
<i>Tabla 15. Descripción de Actores. Capa Zona de Internet.....</i>	33
<i>Tabla 16. Descripción de Casos de Uso. Capa Zona de Internet</i>	33
<i>Tabla 17. Descripción de diagramas de secuencia. Capa Zona de Internet.....</i>	34
<i>Tabla 18. Descripción de Casos de Uso. Capa de Integración</i>	35
<i>Tabla 19. Descripción de diagramas de secuencia. Capa de Integración.....</i>	36
<i>Tabla 20. Descripción de diagrama de secuencia. Capa Ontológica.....</i>	37
<i>Tabla 21. Librerías empleadas para subir archivos al servidor.....</i>	52

ÍNDICE DE FIGURAS

<i>Figura 1. Arquitectura de KIVOX</i>	12
<i>Figura 2. Escenario de Aplicación KIVOX</i>	13
<i>Figura 3. Etapas de identificación de personas mediante la voz TT0091</i>	14
<i>Figura 4. Capas de Arquitectura propuesta</i>	15
<i>Figura 6. Diagrama de Casos de Uso de Arquitectura</i>	23
<i>Figura 7 Diagrama de Clases</i>	25
<i>Figura 8. Diagrama de Actividades</i>	27
<i>Figura 9. Diagrama de Casos de Uso. Capa de Usuario</i>	28
<i>Figura 10. Diagrama de Secuencia. Capa de Usuario</i>	29
<i>Figura 11. Diagrama de Casos de Uso. Capa de Navegación</i>	30
<i>Figura 12. Diagrama de Secuencia. Capa de Navegación</i>	31
<i>Figura 13. Diagrama de Casos de Uso. Capa Zona de Internet</i>	32
<i>Figura 14. Diagrama de Secuencia. Capa Zona de Internet</i>	33
<i>Figura 15. Diagrama de Casos de Uso. Capa de Integración</i>	34
<i>Figura 16. Diagrama de Secuencia. Capa de Integración</i>	35
<i>Figura 17. Diagrama de Casos de Uso. Capa Ontológica</i>	36
<i>Figura 18. Diagrama de Secuencia. Capa Ontológica</i>	37
<i>Figura 19. Diseño conceptual de la base de datos</i>	38
<i>Figura 20. Diseño físico de la base de datos</i>	38
<i>Figura 21. Diagrama de Arquitectura</i>	39
<i>Figura 22. Diseño Modular</i>	41
<i>Figura 23. Funcionamiento del HTTPS</i>	46
<i>Figura 24. Configuración de certificado digital para Apache Tomcat 6.0</i>	47
<i>Figura 25. Algoritmo de autenticación de voz</i>	50
<i>Figura 26. Extracción de vectores mediante cuantificación vectorial</i>	51
<i>Figura 27. Login de usuarios</i>	109
<i>Figura 28. Registro de usuarios</i>	110
<i>Figura 29. Modificación de usuarios</i>	110
<i>Figura 30. Eliminar usuario</i>	111
<i>Figura 31. Asignación de usuarios</i>	111
<i>Figura 32. Consulta de usuarios</i>	112
<i>Figura 33. Registro de servicios</i>	112
<i>Figura 34. Modificar servicios</i>	113
<i>Figura 35. Eliminar servicios</i>	113
<i>Figura 36. Funcionamiento del servidor Apache Tomcat en HTTPS</i>	114
<i>Figura 37. Generación de archivo de vectores de voz</i>	115
<i>Figura 38. Generación del códec</i>	115
<i>Figura 39. Ventana inicial Protégé</i>	116
<i>Figura 40. Generación de esquema ontológico con Protégé</i>	116

OBJETIVOS

Objetivo General

Analizar, diseñar y desarrollar una arquitectura de autentificación basada en dos aspectos principales: ontologías y el reconocimiento de un biométrico (voz) para garantizar con esto un margen de error más reducido y un mínimo de falsos positivos.

Objetivos específicos

- Proporcionar mecanismo de identificación ergonómica basado en la utilización de biométricos para garantizar un medio único y confiable de acceso a los recursos y/o servicios solicitados.
- Adquirir biométrico del usuario en línea para dar portabilidad de acceso a los recursos y/o servicios requeridos.
- Garantizar que la información sea transmitida de manera fiable mediante el uso de canales cifrados en Internet.
- Dar soporte para una correcta gestión del biométrico utilizado para su vinculación con su esquema ontológico.
- Mediante el uso de una ontología garantizar la autentificación del usuario y brindar acceso seguro y confiable a los recursos y/o servicios requeridos.
- Brindar un repositorio de datos robusto para generar las respuestas adecuadas a la petición de usuario.
- Generar la implementación de un prototipo de la arquitectura propuesta, utilizando como biométrico la voz para la comprobación del funcionamiento correcto de la misma.

Además de los objetivos anteriormente mencionados, al finalizar el desarrollo de la arquitectura se obtendrá:

1. El código.
2. La documentación técnica.
3. El manual de usuario.

INTRODUCCIÓN

Sabemos que hoy en día la seguridad (tanto lógica como física) en las instituciones de cualquier índole es un factor primordial, y es por esta necesidad que demandan eficientes mecanismos de autenticación e identificación de usuarios a fin de proteger su activo más valioso (la información). El presente proyecto pretende brindar un medio fiable, robusto y efectivo para estos fines, al brindar una solución con estas características afrontamos un gran reto y es por ello que nos apoyamos en lo último en tecnologías para la autenticación e identificación de personas.

En la parte referente a la identificación usaremos un biométrico, en nuestro caso particular la voz, debido a que resulta un biométrico fácil de manejar, computacionalmente hablando, además de que economiza recursos en su captura, procesamiento y tratamiento; en la parte concerniente a la autenticación usaremos ontologías, ya que son un medio que garantiza una identificación muy cercana al 100% única y libre de duplicidades, además de que nos ayuda de manera inteligente al reconocimiento de individuos.

Aunado a la información brindada por la utilización de una ontología y de un biométrico, se generará un identificador único, tomando como base la información brindada por estos dos medios, con la utilización de un algoritmo de no retorno.

Debido al uso cooperativo y eficiente de esta inteligente combinación se tendrá un producto que sea una alternativa viable y que responda de manera contundente al problema que se desea atacar de fondo, que es la inseguridad en sistemas de información.

CAPÍTULO 1. ESTADO DEL ARTE

1.1 ANTECEDENTES

1.1.1 SISTEMAS DE IDENTIFICACIÓN, AUTENTIFICACIÓN Y CONTROL DE ACCESO

Uno de los requerimientos primordiales de los sistemas informáticos que desempeñan tareas importantes son los mecanismos de seguridad adecuados a la información que se intenta proteger; el conjunto de tales mecanismos ha de incluir al menos un sistema que permita identificar a las entidades (elementos activos del sistema, generalmente usuarios) que intentan acceder a los objetos (elementos pasivos, como ficheros o capacidad de cómputo), mediante procesos tan simples como una contraseña o tan complejos como un dispositivo analizador de patrones retinales.

Los sistemas que habitualmente utilizamos los humanos para identificar a una persona, como el aspecto físico o la forma de hablar, son demasiado complejos para una computadora; el objetivo de los sistemas de identificación de usuarios no suele ser **identificar** a una persona, sino **autenticar** que esa persona es quien dice ser realmente. Aunque como humanos seguramente ambos términos nos parecerán equivalentes, para un ordenador existe una gran diferencia entre ellos: imaginemos un potencial sistema de identificación estrictamente hablando, por ejemplo uno biométrico basado en el reconocimiento de la retina; una persona miraría a través del dispositivo lector, y el sistema sería capaz de decidir si es un usuario válido, y en ese caso decir de quién se trata; esto es identificación. Sin embargo, lo que habitualmente hace el usuario es introducir su identidad (un número, un nombre de usuario...) además de mostrar sus retinas ante el lector; el sistema en este caso no tiene que identificar a esa persona, sino autenticarlo: comprobar los parámetros de la retina que está leyendo con los guardados en una base de datos para el usuario que la persona dice ser: estamos reduciendo el problema de una población potencialmente muy elevada a un grupo de usuarios más reducido, el grupo de usuarios del sistema que necesita autenticarlos.

Los métodos de autenticación se suelen dividir en tres grandes categorías, en función de lo que utilizan para la verificación de identidad:

- a) Algo que el usuario sabe.
- b) Algo que éste posee.
- c) Una característica física del usuario o un acto involuntario del mismo. Conocida con el nombre de **autenticación biométrica**.

Cualquier sistema de identificación (aunque les llamemos así, recordemos que realmente son sistemas de autenticación) ha de poseer unas determinadas características para ser viable; obviamente, ha de ser fiable con una probabilidad muy elevada (podemos hablar de tasas de fallo de en los sistemas menos seguros), económicamente factible para la organización (si su precio es superior al valor de lo que se intenta proteger, tenemos un sistema incorrecto) y ha de soportar con éxito cierto tipo de ataques (por ejemplo, imaginemos que cualquier usuario puede descifrar el *password* utilizado en el sistema de autenticación de Unix en tiempo polinomial; esto sería

inaceptable). Aparte de estas características tenemos otra, no técnica sino humana, pero quizás la más importante: un sistema de autenticación ha de ser aceptable para los usuarios ([23]), que serán al fin y al cabo quienes lo utilicen.

1.1.2 SISTEMAS DE AUTENTIFICACIÓN BIOMÉTRICA

A pesar de la importancia de la criptología en cualquiera de los sistemas de identificación de usuarios vistos, existen otra clase de sistemas en los que no se aplica esta ciencia, o al menos su aplicación es secundaria. Es más, parece que en un futuro no muy lejano estos serán los sistemas que se van a imponer en la mayoría de situaciones en las que se haga necesario autenticar un usuario: son más amigables para el usuario (no va a necesitar recordar *passwords* o números de identificación complejos, y, como se suele decir, el usuario puede olvidar una tarjeta de identificación en casa, pero nunca se olvidará de su mano o su ojo) y son mucho más difíciles de falsificar que una simple contraseña o una tarjeta magnética; las principales razones por la que no se han impuesto ya en nuestros días es su elevado precio, fuera del alcance de muchas organizaciones, y su dificultad de mantenimiento ([24]).

Estos sistemas son los denominados **biométricos**, basados en características físicas del usuario a identificar. El reconocimiento de formas, la inteligencia artificial y el aprendizaje son las ramas de la informática que desempeñan el papel más importante en los sistemas de identificación biométricos; la criptología se limita aquí a un uso secundario, como el cifrado de una base de datos de patrones retinales, o la transmisión de una huella dactilar entre un dispositivo analizador y una base de datos. La autenticación basada en características físicas existe desde que existe el hombre y, sin darnos cuenta, es la que más utiliza cualquiera de nosotros en su vida cotidiana: a diario identificamos a personas por los rasgos de su cara o por su voz. Obviamente aquí el agente reconocedor lo tiene fácil porque es una persona, pero en el modelo aplicable a redes o sistemas Unix el agente ha de ser un dispositivo que, basándose en características del sujeto a identificar, le permita o deniegue acceso a un determinado recurso.

1.1.3 VERIFICACIÓN DE VOZ

En los sistemas de reconocimiento de voz no se intenta, como mucha gente piensa, reconocer lo que el usuario dice, sino identificar una serie de sonidos y sus características para decidir si el usuario es quien dice ser. Para autenticar a un usuario utilizando un reconocedor de voz se debe disponer de ciertas condiciones para el correcto registro de los datos, como ausencia de ruidos, reverberaciones o ecos; idealmente, estas condiciones han de ser las mismas siempre que se necesite la autenticación.

Cuando un usuario desea acceder al sistema pronunciará unas frases en las cuales reside gran parte de la seguridad del protocolo; en algunos modelos, los denominados de texto dependiente, el sistema tiene almacenadas un conjunto muy limitado de frases que es capaz de reconocer: por ejemplo, imaginemos que el usuario se limita a pronunciar su nombre, de forma que el reconocedor lo entienda y lo autentique. Como veremos a continuación, estos modelos proporcionan poca seguridad en comparación con los de texto independiente, donde el sistema va 'proponiendo' a la persona la pronunciación de ciertas palabras extraídas de un conjunto bastante grande. De cualquier forma, sea cual

sea el modelo, lo habitual es que las frases o palabras sean características para maximizar la cantidad de datos que se pueden analizar (por ejemplo, frases con una cierta entonación, pronunciación de los diptongos, palabras con muchas vocales...). Conforme va hablando el usuario, el sistema registra toda la información que le es útil; cuando termina la frase, ya ha de estar en disposición de facilitar o denegar el acceso, en función de la información analizada y contrastada con la de la base de datos.

El habla es una cuestión de comunicación e identificación tan “natural” para las personas, que en un futuro las organizaciones incrementarán el uso de tecnologías de voz para identificación y automatizar preguntas simples en los centros de contacto, dando como resultado sistemas que permitan realizar trámites de servicios de forma muy rápida y segura.

Otros canales de gran crecimiento y donde el apoyo de Autentificación de voz jugará un papel fundamental serán la telefonía móvil (SMS, 3G, 4G...) y la mensajería instantánea en Internet o desde terminales avanzados (PDAs, terminales 3G...) gracias a la utilización de tecnología de reconocimiento del lenguaje natural hablado y escrito se logrará optimizar y automatizar las interacciones generadas desde dichos terminales y reducir las consultas finales gestionadas por los actuales centros de contacto.

Se trata de personajes virtuales que, gracias a una apariencia gráfica y una personalidad específica, son capaces de mantener un diálogo interactivo con los usuarios, ayudándoles a navegar en la página Web en la que se sitúan, resolver consultas, llenar un formulario, comprar de forma proactiva servicios y productos utilizando fórmulas de cross-selling, aprender, conocer. En resumen, proporcionan el mismo nivel de atención y servicio que se podría conseguir a través de un sistema de administración más una correcta autentificación.

Actualmente existen un número importante de desarrollos de aplicaciones que usan la voz como interfaz para la recuperación de información [12][13], además existen ahora plataformas de desarrollo para aplicaciones vocales más conocidas que se pueden encontrar en [14][15], cuya característica principal es que se implementa una tecnología de reconocimiento del habla, independiente del hablante, lo común de éstas, es que usan el lenguaje VoiceXML; además existen algunas aproximaciones a la combinación de los dos técnicas como se puede ver en [10][11], en mayo del 2005 se llevó a cabo en Grecia una conferencia cuyo objetivo principal era organizar jornadas de trabajo internacionales sobre tecnologías de voz, lenguaje natural y Web semántica para mejorar el acceso de los ciudadanos a los servicios de la administración pública, se puede revisar en [20].

1.2 PROBLEMÁTICA

El principal problema del reconocimiento de voz es la inmunidad frente a *replay attacks*, un modelo de ataques de simulación en los que un atacante reproduce (por ejemplo, por medio de un magnetófono) las frases o palabras que el usuario legítimo pronuncia para acceder al sistema. Este problema es especialmente grave en los sistemas que se basan en textos preestablecidos: volviendo al ejemplo anterior, el del

nombre de cada usuario, un atacante no tendría más que grabar a una persona que pronuncia su nombre ante el autenticador y luego reproducir ese sonido para conseguir el acceso; casi la única solución consiste en utilizar otro sistema de autenticación junto al reconocimiento de voz. Por contra, en modelos de texto independiente, más interactivos, este ataque no es tan sencillo porque la autenticación se produce realmente por una especie de desafío-respuesta entre el usuario y la máquina, de forma que la cantidad de texto grabado habría de ser mucho mayor - y la velocidad para localizar la parte del texto que el sistema propone habría de ser elevada -. Otro grave problema de los sistemas basados en reconocimiento de voz es el tiempo que el usuario emplea hablando delante del analizador, al que se añade el que éste necesita para extraer la información y contrastarla con la de su base de datos; aunque actualmente en la mayoría de sistemas basta con una sola frase, es habitual que el usuario se vea obligado a repetirla porque el sistema le deniega el acceso (una simple congestión hace variar el tono de voz, aunque sea levemente, y el sistema no es capaz de decidir si el acceso ha de ser autorizado o no; incluso el estado anímico de una persona varía su timbre...). A su favor, el reconocimiento de voz posee la cualidad de una excelente acogida entre los usuarios, siempre y cuando su funcionamiento sea correcto y éstos no se vean obligados a repetir lo mismo varias veces, o se les niegue un acceso porque no se les reconoce correctamente.

Es importante aclarar que cuando se habla del reconocimiento de voz se hace referencia al reconocimiento de palabras aisladas no necesariamente es reconocimiento de lenguaje natural, lo cual significa que no hay extracción de significado. El problema que se presenta es garantizar que los sonidos emitidos sean comprendidos sin problema y que estos sonidos permitan identificar al emisor de los sonidos. En cuanto a la extracción de significado de manera general se puede presentar dos dificultades que son, la ambigüedad (léxica o estructural), o una entrada con errores. Uno de los problemas más serios es que una entrada de una sola palabra puede ser verbo o sustantivo.

Ahora bien el análisis sintáctico tiene el propósito de asignar una estructura a una sentencia o conjunto de ellas de acuerdo a una gramática donde las relaciones entre verbos y nombres son explícitas. Para hacer una aclaración el análisis sintáctico no da información acerca del significado sino de la estructura, el significado tiene que ver con el análisis semántico.

Actualmente se pueden distinguir cinco factores principales en la problemática del reconocimiento del habla, que son:

- El locutor
- La forma de hablar
- El vocabulario
- La gramática
- El entorno físico.

Ya que no es lo mismo un sistema que funciona en un ambiente poco ruidoso, que un sistema cuyo nivel de ruido está controlado. [23]

Principalmente el autenticar usuarios por medio de su voz es un trabajo complejo en el sentido de poder garantizar que la voz es de la persona autorizada para tener los servicios.

1.3 SOLUCIONES EXISTENTES

Actualmente en el mercado de la autentificación por medio de biométricos de voz, existen productos de alta calidad y que resuelven el problema de distintas maneras, a continuación analizaremos detalladamente estos productos dando sus principales características y describiendo la forma en que estos sistemas operan para poder llegar a su objetivo.

SOFTWARE	CARACTERÍSTICAS
KIVOX	Es un sistema que ofrece un método de autentificación remoto: el biométrico de voz, es decir la autenticación por medio de la voz, este sistema interactúa con sistemas existentes a través del intercambio de documentos XML, y por este procedimiento se hace enrolamiento, validación y autenticación de los usuarios del sistema. Sistemas operativos soportados: Windows XP, 2003, Vista, Linux, Solaris, HP-UX
TT0091 “Identificador de personas mediante la voz”	Este sistema permite identificar personas mediante la voz, para lo cual se adquiere la señal de voz de la persona, se procesa y se efectúa el reconocimiento que depende de que la persona esté dada de alta en la base de datos que maneja el sistema. Sistemas Operativos Soportados: Windows XP, 2003, NT.
<u>Propuesta de solución:</u> TT 2008012 “Arquitectura de Autentificación por biométrico de voz basada en Ontologías”	Arquitectura de autentificación basada en dos aspectos principales: por una parte las ontologías las cuales nos permiten autenticar a un individuo y por otra el reconocimiento de voz (biométrico) lo cual nos permite identificar para su posterior tratamiento en formato VoXML, garantizando esto con un margen de error más reducido. Sistemas Operativos Soportados: Múltiples (todo aquel que soporte java puesto que básicamente se desarrollara bajo este lenguaje).

Tabla 1.Soluciones existentes y Propuesta de solución

❖ KIVOX

KIVOX es un sistema automático de reconocimiento de locutor basado en la tecnología biométrica de voz desarrollada por AGNITIO. Esta compañía, como proveedor de tecnología, necesita la ayuda de un proveedor de soluciones o integrador para desplegar una solución completa y aplicación a los clientes finales. KIVOX se puede integrar fácilmente utilizando una petición XML - respuesta protocolo.

La comunicación con KIVOX se basa en la KIVOX interfaz XML que consiste en el intercambio de archivos XML, esta interfaz estándar XML hace posible integrar KIVOX con sistemas que ya están desplegados.

KIVOX interactúa con un módulo de integración que actúa como interfaz entre el sistema de respuesta de voz interactiva y KIVOX.

KIVOX recibe una petición y responde con los parámetros necesarios para cada acción. Lo que el módulo de integración debería hacer es construir las páginas de respuesta de voz interactiva (Vxml páginas u otros) añadiendo los parámetros enviados por KIVOX.

La página es enviada a la respuesta de voz interactiva, generando así el diálogo con el usuario. Por lo tanto, las tareas que la integración del módulo cumple son:

- Intercambio de mensajes XML
- Generación de peticiones XML (incluyendo la codificación de audios en formato de datos BASE64 para intercambio de archivos)
- Recepción de las respuestas XML.
- Extracción de los parámetros.
- La generación de cuadros de diálogo
- Creación de páginas VXML (o su equivalente, según el sistema de respuesta de voz interactiva) con el cuadro de diálogo para cada interacción con el usuario (inscripción y verificación).
- Inserción de los parámetros enviados desde KIVOX en las páginas para completar el cuadro de diálogo.
- Envío de las páginas a la respuesta de voz interactiva e interactuar con él.

El siguiente diagrama muestra el escenario de aplicación:

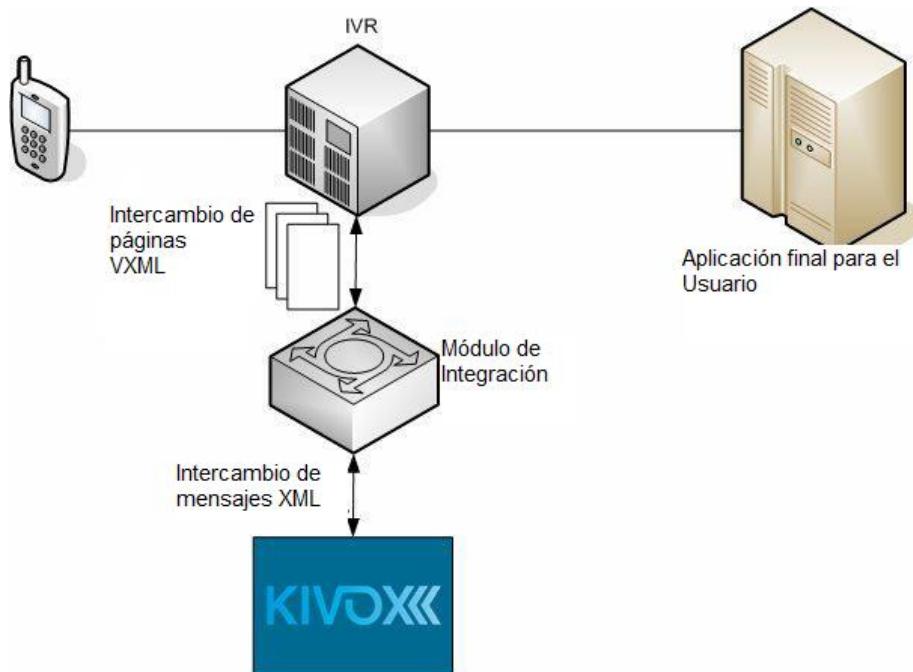


Figura 1. Arquitectura de KIVOX

El módulo de integración tiene dos partes diferenciadas: una que interactúa con el IVR (Interactive Voice Response) y otro que interactúa con KIVOX. El siguiente diagrama muestra el escenario de aplicación:

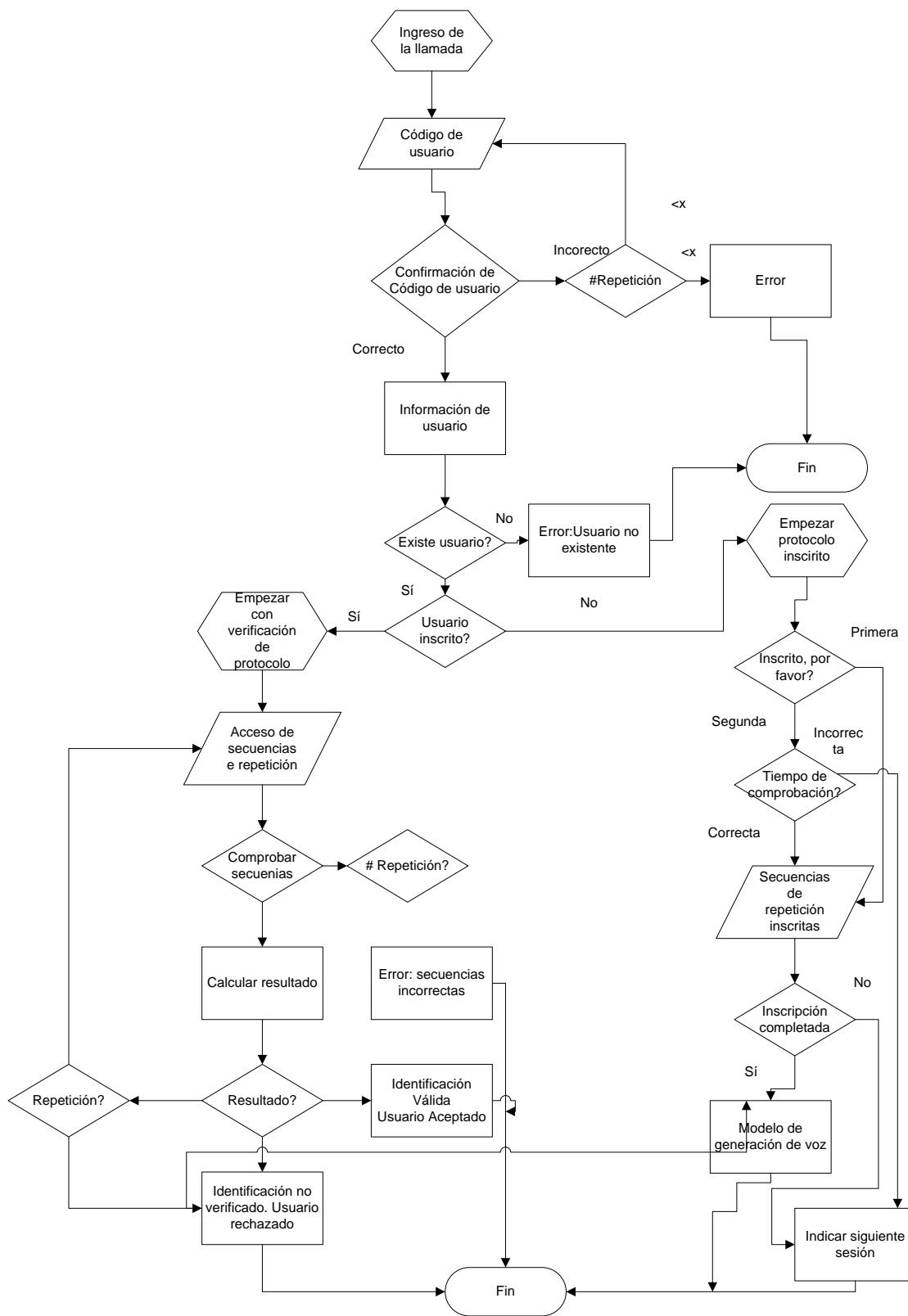


Figura 2. Escenario de Aplicación KIVOX

❖ TT0091 “Identificador de personas mediante la voz”

Este sistema permite identificar personas mediante la voz, para lo cual se adquiere la señal de voz de la persona, se procesa y se efectúa el reconocimiento que depende de que la persona esté dada de alta en la base de datos que maneja el sistema.

Utiliza algoritmos de procesamiento matemático para realizar la extracción de patrones característicos de cada persona, para lo cual se utiliza un procesador de señales digitales que se encarga de efectuar el procesamiento de la señal de voz. El reconocimiento se efectúa mediante Redes Neuronales.

Primeramente se elimina el ruido mediante un filtro analógico y se amplifica la señal hasta un nivel de voltaje identificable. Para poder manipular la señal de la voz, la cual es una señal continua, se transforma la misma señal en una señal discreta y se segmenta para su procesamiento y reconocimiento, el procesamiento está basado en los algoritmos denominados Cepstrum y Análisis Predictivo (LPC).

Una vez que se determinan los patrones característicos de cada persona la etapa de decisión se realiza mediante el uso de una red de Retro propagación o una red de Cohohen.

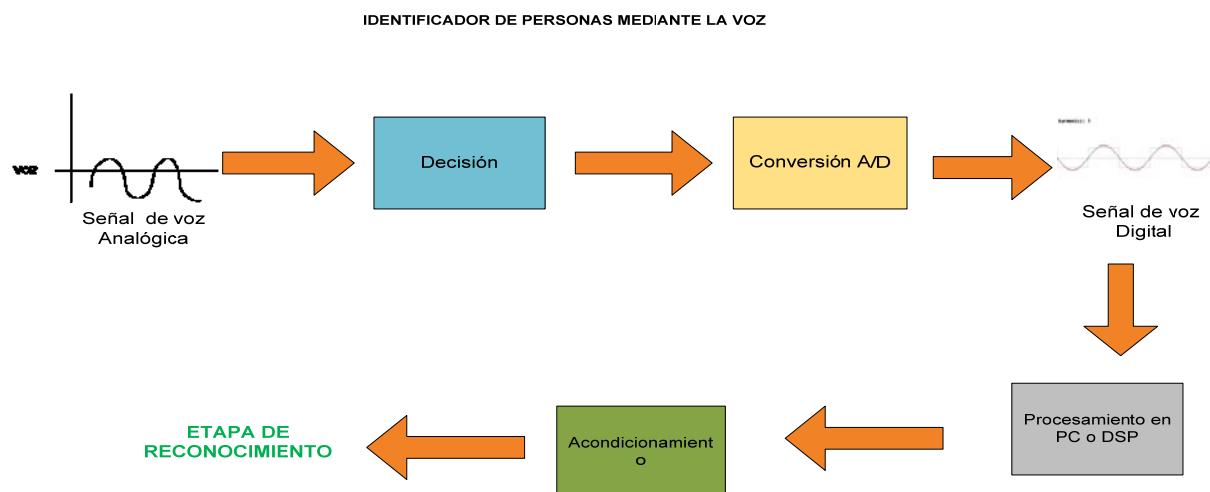


Figura 3. Etapas de identificación de personas mediante la voz TT0091

CAPÍTULO 2. ANÁLISIS

2.1 PROPUESTA DE SOLUCIÓN

Ya habiendo analizado las soluciones existentes al problema que deseamos atacar, y habiendo visto sus principales ventajas y debilidades, nosotros pretendemos realizar un producto que ofrezca una solución más portable, fiable, robusta y de mucho menor costo que las soluciones existentes.

La solución que ofrecemos en concreto es desarrollar una Arquitectura de Autentificación por Biométrico de voz basada en Ontologías, a continuación se presenta la arquitectura capa por capa con la intención de hacer claridad en la misma:

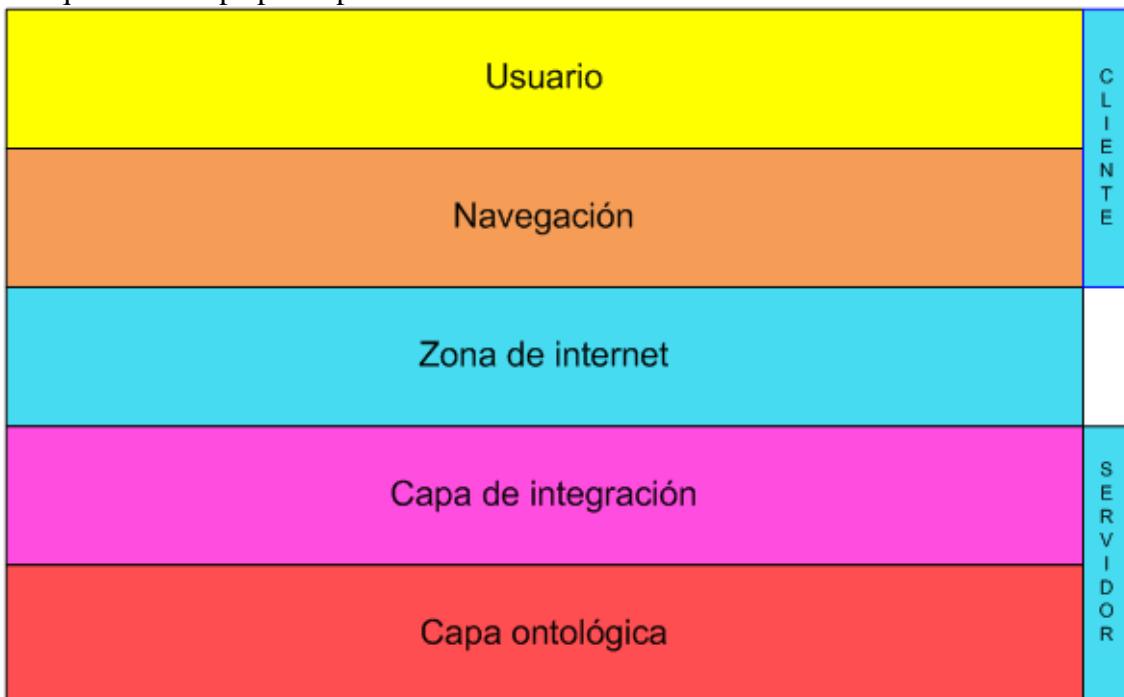


Figura 4. Capas de Arquitectura propuesta

A. Capa de Usuario

En esta capa el usuario es el actor numero uno que interactúa con toda la arquitectura propuesta.

B. Capa Navegador

Esta capa define que para la navegación con voz se utilizaría un navegador vocal como Opera.

C. Zona Internet

En esta capa funcionan los protocolos de comunicación vía Web, en la arquitectura se hace uso del protocolo HTTPS¹, éste utiliza un cifrado basado en las Secure Socket Layers (SSL) para crear un canal cifrado (cuyo nivel de cifrado depende del servidor remoto y del navegador utilizado por el cliente) más apropiado para el tráfico de información sensible que el protocolo HTTP.

¹ Versión segura del protocolo HTTP.

D. Capa de Integración

En esta capa se define un servidor Web y otro de voz. Para que el documento en formato VoXML, enviado por el navegador vocal, será traducido para su posterior tratamiento en la capa ontológica.

E. Capa Ontológica

En esta capa se hará uso de una ontología que autentificara de manera inequívoca al usuario. La información recibida del biométrico, será vinculada con su definición correspondiente en la ontología y se tendrá así acceso a la identidad del usuario solicitante y en caso de ser esta comprobada se dará acceso a los recursos solicitados.

2.2 ANÁLISIS DE REQUERIMIENTOS

2.2.1 REQUERIMIENTOS FUNCIONALES

REQUERIMIENTOS	ID	VERSIÓN
Capturar Biométrico en línea.	RF1	1
Registro de Usuarios en línea.	RF2	1
Transformar la voz al formato VoXML.	RF3	1
Enviar la información por un canal cifrado.	RF4	1
Traducir documento VoXML.	RF5	1
Vincular biométrico (voz) a su definición ontológica.	RF6	1
Autenticar usuario.	RF7	1
Proporcionar el Servicio Solicitado	RF8	1
Manual del usuario.	RF9	1
Manual del administrador.	RF10	1
Términos y Condiciones de uso.	RF11	1

Tabla 2. Requerimientos Funcionales

RF1. Capturar Biométrico en línea.

La captura del biométrico que se desee usar para la identificación de los usuarios deberá de poderse hacer en línea es decir vía Internet a fin de brindar la máxima portabilidad del servicio.

RF2. Registro de Usuarios en línea.

El registro de los usuarios con la finalidad de crear la base de conocimiento de las ontologías, se lleva a través de Internet.

RF3 Transformar la voz al formato VoXML.

A partir de la captura del biométrico de voz por medio del navegado, se generara un documento VoXML donde estará contenida la información de la voz capturada de manera íntegra.

RF4 Enviar la información por un canal cifrado.

Debido a la sensibilidad de la información que se maneja en la presente arquitectura y ya que esta hace uso de internet como medio de transmisión de la información, se usara un canal cifrado d para el envío y recepción de datos en la arquitectura.

RF5 Traducir documento VoXML.

El documento VoXML que se obtiene mediante la captura de la voz, será posteriormente traducido a un formato entendible al manejador ontológico a fin de que este pueda hacer la vinculación con su definición dentro de la ontología.

RF6 Vincular biométrico (voz) a su definición ontológica.

Cada uno de los biométricos habilitados para la identificación de los usuarios estarán contemplados en la definición de la ontología, así cada uno de ellos estará vinculado de manera directa con la definición ontológica de los usuarios.

RF7 Autentificar usuario.

El usuario será autenticado mediante su biométrico y la vinculación de este con su definición dentro de la ontología.

RF8 Proporcionar el Servicio Solicitado

Ya pasando durante las capas de la arquitectura y habiendo sido correctamente autenticado e identificado se dará acceso al usuario al recurso o servicio solicitado de manera inmediata.

RF9 Manual del usuario.

Tendrá un apartado de Instrucciones de uso para el usuario, en el cual se explicará el procedimiento a seguir para el correcto empleo de la arquitectura, su funcionamiento, el soporte que brinda para los distintos biométricos, resolverá las principales interrogantes que se presenten al hacer uso de la arquitectura y también al momento de hacer la integración con otros medios tanto de software como de hardware.

RF10 Manual del administrador.

Tendrá documentos de instrucciones de uso para el administrador del sistema, en donde se especificará las formas de actualización y adquisición de información de precios, clientes y pedidos. Este proporcionara la ayuda al administrador en sus interrogantes que presente.

RF11 Términos y Condiciones de uso.

Para la obtención de registro de usuario la arquitectura se especificará las condiciones y términos bajo las cuales se mantendrá la arquitectura así como también la forma de obtención de del registro de usuario.

2.2.2 REQUERIMIENTOS NO FUNCIONALES

REQUERIMIENTOS	ID	VERSIÓN
Transmitir la información de manera ágil.	RNF1	1
Fácil Navegación.	RNF2	1
Portabilidad en la captura del biométrico	RNF3	1
Estabilidad	RNF4	1
Compatibilidad con distintos dispositivos de entrada.	RNF5	1
Extensible a distintos biométricos	RNF6	1
Bajos costos en la elaboración	RNF7	1
Fácil mantenimiento	RNF8	1
Flexible	RNF9	1

Tabla 3. Requerimientos no Funcionales

RNF1 Transmitir la información de manera ágil.

El tiempo de adquisición de datos del sistema Web, será el menor posible, pensando en la comodidad de los usuarios del sistema.

Así mismo el tiempo de respuesta y procesamiento debe ser rápido, a fin de que sea práctico y funcional y no provoque descontentos en el usuario.

RNF2 Fácil Navegación.

Debe ser de fácil navegación ya que estará en acceso, además tendrá interfaces simples para su entendimiento y podrá ser hallado en Internet.

RNF3 Portabilidad en la captura del biométrico

La captura del biométrico debe de ser portable a fin de que la captura de este sea fácil al usuario no importando donde se encuentre, esta debe de ser flexible y adaptarse a las condiciones de los usuarios.

RNF4 Estabilidad

La arquitectura debe ser estable a fin de brindar seguridad, confianza y un medio atractivo tanto a los usuarios como a todos aquellos interesados en usarla para poder desarrollar sus sistemas sobre ella.

RNF5 Compatibilidad con distintos dispositivos de entrada.

Al ser una arquitectura, y ser un modelo sobre el cual se montaran distintos medios tanto físicos como lógicos, esta debe de brindar soporte y compatibilidad con distintos medio de captura para los distintos biométricos.

RNF6 Extensible a distintos biométricos

La arquitectura propuesta deberá ser capaz de extenderse y soportar la identificación y autenticación sobre distintos biométricos, solo cambiando las fases iniciales del proceso como la adquisición de los datos.

RNF7 Bajos costos en la elaboración

La arquitectura debe de tener un bajo costo en su elaboración a fin de poder ser una atractiva propuesta a todo aquel que quiera desarrollar un sistema de autenticación por medio de biométricos.

RNF8 Fácil mantenimiento

Debe de ser fácilmente de mantener a fin de adaptarse a los constantes cambios tecnológicos de manera rápida, lo que le dará vigencia durante un largo tiempo.

RNF9 Flexible

Debe ser flexible en su contenido ya que estará en constante mantenimiento. Para ello será necesaria una constante observación del funcionamiento del sistema Web por parte de técnicos especializados capaces de detectar cualquier anomalía.

2.3 ESTUDIO DE VIABILIDAD

Para estudiar la viabilidad del presente proyecto de software debemos de contemplar distintos aspectos que un proyecto de esta índole requiere para decidir si este es o no factible a ser realizado, dichos factores son los siguientes:

2.3.1 VIABILIDAD ECONÓMICA

Para comenzar con el estudio de la viabilidad económica del presente proyecto, debemos de hacer especial énfasis en la importancia de este rubro, ante todo cabe señalar que cuando se comienza el desarrollo de un proyecto software se busca que este resuelve de manera integra el problema para el que es realizado pero en gran medida se busca que el costo de realización del mismo resulte conveniente tanto para quien desea el producto como para quien desea desarrollarlo, es por ello que el presente estudio de viabilidad económica demostrará que el proyecto que se pretende realizar cumple con estas características.

Para hacer la estimación de los costos del desarrollo de este proyecto se empleo el modelo de estimación denominado COCOMO II, el cual corresponde al esfuerzo de desarrollo estimado una vez que se ha fijado la arquitectura del sistema.

Para comenzar con el uso de COCOMO definimos en primer lugar dentro de que clasificación del modelo cae el proyecto, para nuestro caso el modelo intermedio será el que usaremos, dado que realiza las estimaciones con bastante precisión.

Así pues las fórmulas serán las siguientes:

- **E = Esfuerzo = a KLDC^e * FAE (persona x mes)**
- **T = Tiempo de duración del desarrollo = c Esfuerzo^d (meses)**
- **P= Personal = E/T (personas)**

Para calcular el Esfuerzo, necesitaremos hallar la variable KDC (Kilo-líneas de código), donde los PF son 157.29 (dato conocido) y las líneas por cada PF equivalen a 53 según vemos en la tabla que se ilustra a continuación:



LENGUAJE	LDC/PF
Ensamblador	320
C	150
COBOL	105
Pascal	91
Prolog/LISP	64
Java	53

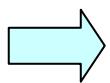
C++/PHP	64
Visual Basic	32
SQL	12

Tabla 4. Comparativa Lenguajes

Así pues tras saber que son 53 LDC por cada PF, por el hecho de ser Java el resultado de los KDLC será el siguiente:

$$KLDC = (PF * \text{Líneas de código por cada PF})/1000 = (157.29*53)/1000 = 8.368 \\ KDLC$$

Así pues, en nuestro caso el tipo orgánico será el más apropiado ya que el número de líneas de código no supera los 50 KLDC, y además el proyecto no es muy complejo, por consiguiente, los coeficientes que usaremos serán las siguientes:



Proyecto Software	a	E	c	d
Orgánico	3,2	1,05	2,5	0,38
Semi-acoplado	3,0	1,12	2,5	0,35
Empotrado	2,8	1,20	2,5	0,32

Tabla 5. Tipos de Proyectos Software

Y por otro lado también hemos de hallar la variable FAE, la cual se obtiene mediante la multiplicación de los valores evaluados en los diferentes 15 conductores de coste que se observan en la siguiente tabla:

Conductores de coste	VALORACIÓN					
	Muy bajo	Bajo	Nominal	Alto	Muy Alto	Extr. alto
Fiabilidad requerida del software	0,75	0,88	1.00	1,15	1,40	-
Tamaño de la base de datos	-	0,94	1,00	1,08	1,16	-
Complejidad del producto	0,70	0,85	1.00	1,15	1,30	1,65
Restricciones del tiempo de ejecución	-	-	1.00	1,11	1,30	1,66
Restricciones del almacenamiento principal	-	-	1,00	1,06	1,21	1,56
Volatilidad de la máquina virtual		0,87	1,00	1,15	1,30	-
	-					
Tiempo de respuesta del ordenador	-	0,87	1.00	1,07	1,15	-
Capacidad del analista	1,46	1,19	1.00	0,86	0,71	-
Experiencia en la aplicación	1,29	1,13	1.00	0,91	0,82	-
Capacidad de los programadores	1,42	1,17	1.00	0,86	0,70	-
Experiencia en S.O. utilizado	1,21	1,10	1,00	0,90	-	-
Experiencia en el lenguaje de programación	1,14	1,07	1.00	0,95	-	-
Prácticas de programación modernas	1,24	1,10	1,00	0,91	0,82	-
Utilización de herramientas software	1,24	1,10	1.00	0,91	0,83	-
Limitaciones de planificación del proyecto	1,23	1,08	1.00	1,04	1,10	-

Tabla 6. Conductores de Coste

$$FAE = 1.15 * 1.00 * 1.11 * 1.00 * 1.00 * 1.07 * 0.86 * 0.82 * 0.70 * 1.00 * 0.95 * 1.00 * 0.91 * 1.08 \\ = 0.944267305004478$$

Justificación de los valores:

Atributos de software

- Fiabilidad requerida del software: Si se produce un fallo por el pago de un pedido, o fallo en alguna reserva, etc... puede ocasionar grandes pérdidas a la empresa (Valoración Alta).
- Tamaño de la base de datos: La base de datos de nuestro producto será de tipo estándar (Valoración Nominal).
- Complejidad del producto: La aplicación realizará cálculos complejos (Valoración Alta).

Atributos de hardware

- Restricciones del tiempo de ejecución: En los requerimientos se exige alto rendimiento (Valoración Alta).
- Restricciones del almacenamiento principal: No hay restricciones al respecto (Valoración Nominal).
- Volatilidad de la máquina virtual: Se usarán sistemas de la “Familia Windows” (Valoración Nominal).
- Tiempo de respuesta del ordenador: Deberá ser interactivo con el usuario (Valoración Alta).

Atributos del personal

- Capacidad del analista: Capacidad alta relativamente, debido a la experiencia en análisis en proyecto similar (Valoración Alta)
- Experiencia en la aplicación: Se tiene cierta experiencia en aplicaciones de esta envergadura (Valoración muy alta).
- Capacidad de los programadores: Teóricamente deberá tenerse una capacidad muy alta por la experiencia en anteriores proyectos similares (Valoración muy alta).
- Experiencia en S.O. utilizado: Con Windows 2000 Professional la experiencia es a nivel usuario (Valoración Nominal).
- Experiencia en el lenguaje de programación: Es relativamente alta, dado que se controlan las nociones básicas y las propias del proyecto (Valoración Alta).

Atributos del proyecto

- Prácticas de programación modernas: Se usarán prácticas de programación mayormente convencional (Valoración Nominal).
- Utilización de herramientas software: Se usarán herramientas estándar que no exigirán apenas formación, de las cuales se tiene cierta experiencia (Valoración Alta).
- Limitaciones de planificación del proyecto: Existen pocos límites de planificación. (Valoración Baja).

Cálculo del esfuerzo del desarrollo:

$$E = a \cdot KLDC \cdot e^{\cdot FAE} = 3,2 \cdot (8.368)^{1.05} \cdot 0.9442673 = 27 \text{ personas /mes}$$

Cálculo tiempo de desarrollo:

$$T = c \cdot \text{Esfuerzo} \cdot d = 2,5 \cdot (27)^{0,38} = 8 \text{ meses}$$

Productividad:

$$PR = LDC/\text{Esfuerzo} = 8368/27 = 310 \text{ LDC/personas mes}$$

Personal promedio:

$$P = E/T = 27/8 = 3.3 \text{ personas}$$

Según estas cifras será necesario un equipo de 3 personas trabajando alrededor de 8 meses.

2.3.2 VIABILIDAD TÉCNICA

Un aspecto importante de todo proyecto Software es comprobar la viabilidad técnica para realizarlo, esto implica que: existe tecnología que posibilita el desarrollo del mismo o en su caso es factible desarrollarla para la elaboración del mismo. En el caso del proyecto se requiere del uso de distintas tecnologías que mencionamos a continuación:

- Navegador con soporte para VOXML: en el caso particular de la implementación de nuestra arquitectura necesitamos hacer uso de navegadores de internet que soporten VoXML, dentro de los cuales nos enfocaremos en uno en especial ya que dentro de sus características destaca el soporte de esta tecnología.
- Creación de Documentos que puedan ser trasferidos vía internet: en este caso nosotros usaremos documentos generados a partir de sonidos, en concreto documentos VoXML, que permiten almacenar en un documento la información íntegra del audio capturado por medio del navegador.
- Uso de un canal cifrado para la transferencia de información a través de internet, para este rubro se utilizará el protocolo HTTPS, éste utiliza un cifrado basado en las Secure Socket Layers (SSL) para crear un canal cifrado lo que hace a internet un medio más apropiado para el tráfico de información sensible que el protocolo HTTP.
- Uso de un servidor de aplicaciones capaz de procesar tanto el envío de información del cliente (voz y datos), para esto usaremos dos servidores, uno de aplicaciones que en nuestro caso será Tomcat, y otro de voz.
- Necesitamos de igual forma un manejar de ontologías, que como se expresó en la descripción de la propuesta tenemos contemplado el uso de Protégé, ésta plataforma soporta dos caminos principales para el modelado de ontologías, que son Protégé Frames y Protégé OWL. Además se hará uso del framework JENA que brinda un buen soporte para la persistencia de datos en forma de ontología.

Asimismo se necesitan de herramientas del lado del servidor, que permitan el tratamiento de la voz en formato VoXML y su posterior traducción a un lenguaje que soporten las ontologías, para estos dos procedimientos medulares, nos apoyaremos de APIs desarrolladas en lenguaje Java, además de esto usaremos como estándar de facto para el desarrollo de la arquitectura Java2EE, lo que hará a nuestra aplicación robusta, escalable y de fácil mantenimiento.

2.3.3 ALTERNATIVAS

Al hacer un proyecto software se deben de tener incluidas alternativas de desarrollo en caso de que la presente solución con el tiempo deje de ser posible de realizarse, en nuestro caso contemplamos dos posibles alternativas:

- Emplear ingeniería inversa en el proyecto denominado KIVOX para poder así desarrollar un producto de similares características que resolviera la problemática planteada.
- Optar por desarrollar el proyecto bajo el mismo modelo que se plantea solo cambiando las tecnologías de la implementación, es decir en gran medida el

presente proyecto es solo presentar un modelo no una implementación, así que solo podemos desarrollar en todo caso el modelo y no el implementación, asegurando así el desarrollo del proyecto por completo.

2.4 MODELADO UML

Para comenzar el modelado del sistema de manera formal, en primera instancia definiremos la herramienta que utilizaremos.

Para nuestro modelado emplearemos la versión 2.1 de UML, y en esta etapa de Análisis simplemente haremos uso de Diagramas de Casos de Uso, Diagramas de Clases y de Actividad.

Dado que nuestro proyecto consiste en la creación de una Arquitectura, la cual está compuesta de varias capas, en primera instancia haremos el modelado con los diagramas antes mencionados a un nivel general de la arquitectura, y posteriormente haremos el modelado con cada uno de los diagramas a nivel de capa.

Dando así un detalle a nuestro modelado, a fin de brindar una completa comprensión del funcionamiento interno y composición de nuestra arquitectura.

2.4.1 MODELADO GENERAL DE LA ARQUITECTURA

2.4.1.1 DIAGRAMA DE CASOS DE USO

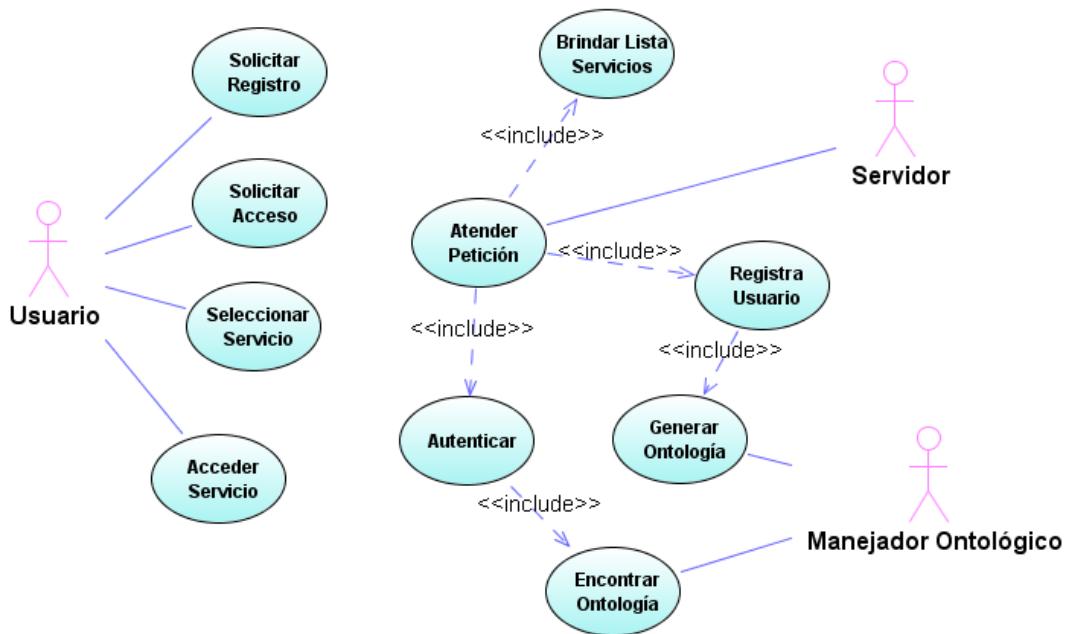


Figura 5. Diagrama de Casos de Uso de Arquitectura

Actores	Descripción
Usuario	Es el actor principal que interactúa con la arquitectura propuesta y es quien desata todos los procesos que se pueden realizar.
Administrador	Es una extensión de un Usuario. Es el encargado de dar de alta a los nuevos usuarios comunes con el fin de que la información introducida

	sea correcta y verdadera.
Servidor	Es el encargado de de atender las peticiones de los usuarios y dar o denegar el acceso a los servicios solicitados.
Manejador Ontológico	Es el encargado de transformar los datos introducidos por el usuario en una ontología o en caso de ya existir dichos datos en la base de conocimiento, asociar el biométrico introducido con el esquema ontológico correspondiente.

Tabla 7. Descripción de Actores en Diagrama de Casos de Uso

Caso de Uso	Descripción
Solicitar Registro	Caso de uso activado por el usuario que desea registrarse dentro del sistema. Solo se hará en presencia de un Administrador.
Solicitar Acceso	Caso de uso que permite al usuario identificarse para posteriormente hacer uso de los servicios disponibles para él.
Solicitar Servicio	En este, el usuario elige dentro de los diferentes servicios disponibles para él y que el servidor le proporciona.
Seleccionar Servicio	El usuario selecciona de una serie de servicios disponibles, aquel que satisfaga sus necesidades.
Atender Petición	El servidor recibe un petición por parte del usuario y dependiendo de esta petición decide que soporte brindar y q recursos utilizar para poder responder adecuadamente al usuario.
Brindar Lista de Servicios	El servidor proporciona al usuario autenticado una lista de servicios disponibles para dicho usuario.
Registrar Usuario	Caso de uso activado por el caso de uso de Atender Petición. En este el servidor registra un nuevo usuario y lo guarda en la base de datos.
Generar Ontología	Teniendo los datos del usuario, se genera la Ontología propia del usuario por el Manejador Ontológico.
Autenticar	El servidor recibe el biométrico del usuario y dependiendo de la respuesta que le dé el Manejador Ontológico a través del caso de Uso de Encontrar Ontología, brinda o no acceso a los servicios del usuario.
Encontrar Ontología	Caso de uso encargado de encontrar el biométrico dentro de las ontologías y asociarla a una en específico.

Tabla 8. Descripción de Casos de Uso de Arquitectura

2.4.1.2 DIAGRAMA DE CLASES

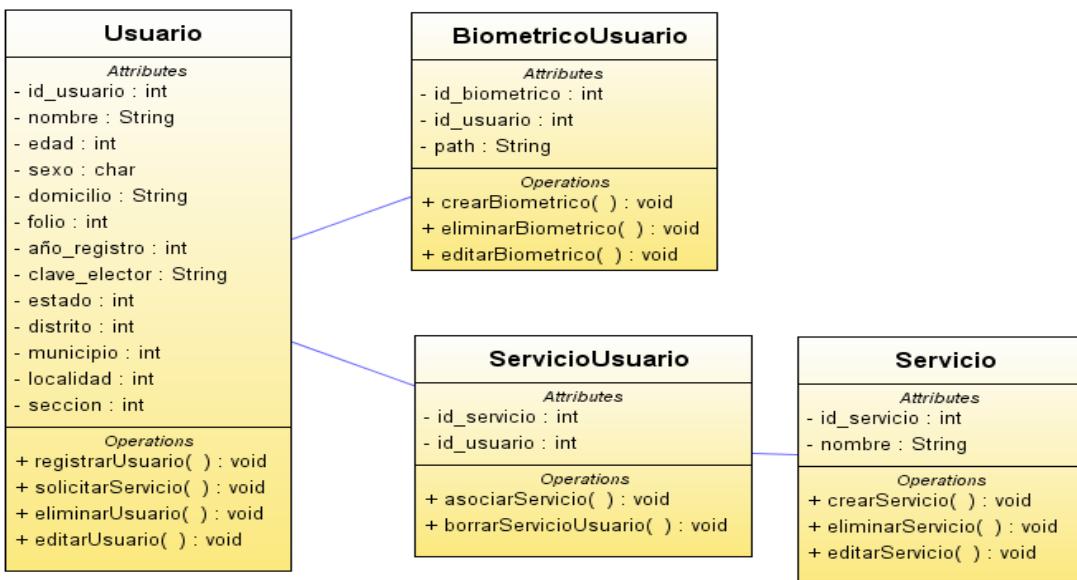


Figura 6 Diagrama de Clases

CLASE USUARIO. Clase que encapsula a los usuarios que interactúan con la arquitectura.

- Atributos:
 - id_usuario. Identificador único de usuario generado por algoritmos de no retorno con base a su esquema ontológico.
 - nombre, edad, sexo, domicilio, folio, año_registro, clave_electro, estado, distrito, municipio, localidad, seccion. Se refieren a los datos de identificación oficiales que están contemplados en el Registro Federal de Electores y que se encuentran en las credenciales para votar.
- Métodos:
 - registrarUsuario. Permite guardar un registro de usuario en la base de datos validando la información introducida.
 - solicitarSevicio. Se encarga de solicitar un servicio de los que se encuentran disponibles para el usuario.
 - eliminarUsuario. Borra el registro de un usuario en la base de datos.
 - editarUsuario. Permite cambiar los datos de un registro de usuario validando que la información siga siendo persistente.

CLASE BIOMETRICO. Se encarga de encapsular la información de los diferentes biométricos del usuario que puedan ser computables.

- Atributos:
 - id_biometrico. Identificador único de biométrico.
 - id_usuario. Se refiere al identificador del usuario al que pertenece el biométrico.
 - path. Contiene la ruta del archivo XML q contiene el biométrico.
- Métodos:
 - crearBiometrico. Permite guardar un registro de biométrico en la base de datos validando la información introducida.

- eliminarBiometrico. Borra el registro de un biométrico en la base de datos.
- editarBiometrico. Permite cambiar los datos de un registro de biométrico validando que la información siga siendo persistente.

CLASE SERVICIO. Contiene la definición de un servicio al cual el usuario puede acceder.

- Atributos:
 - id_servicio. Identificador único de servicio.
 - nombre. Nombre del servicio.
- Métodos:
 - crearServicio. Permite guardar un registro de servicio en la base de datos validando la información introducida.
 - eliminarServicio. Borra el registro de un servicio en la base de datos.
 - editarServicio. Permite cambiar los datos de un registro de servicio validando que la información siga siendo persistente.

CLASE SERVICIO USUARIO. Contiene la definición de que servicio puede utilizar un determinado usuario.

- Atributos:
 - id_servicio. Identificador único de servicio.
 - id_usuario. Identificador único de usuario.
- Métodos:
 - asociarServicio. Asocia un servicio con un determinado usuario.
 - borrarServicioUsuario. Borra la asociación existente entre un servicio y un usuario.

2.4.1.3 DIAGRAMA DE ACTIVIDADES

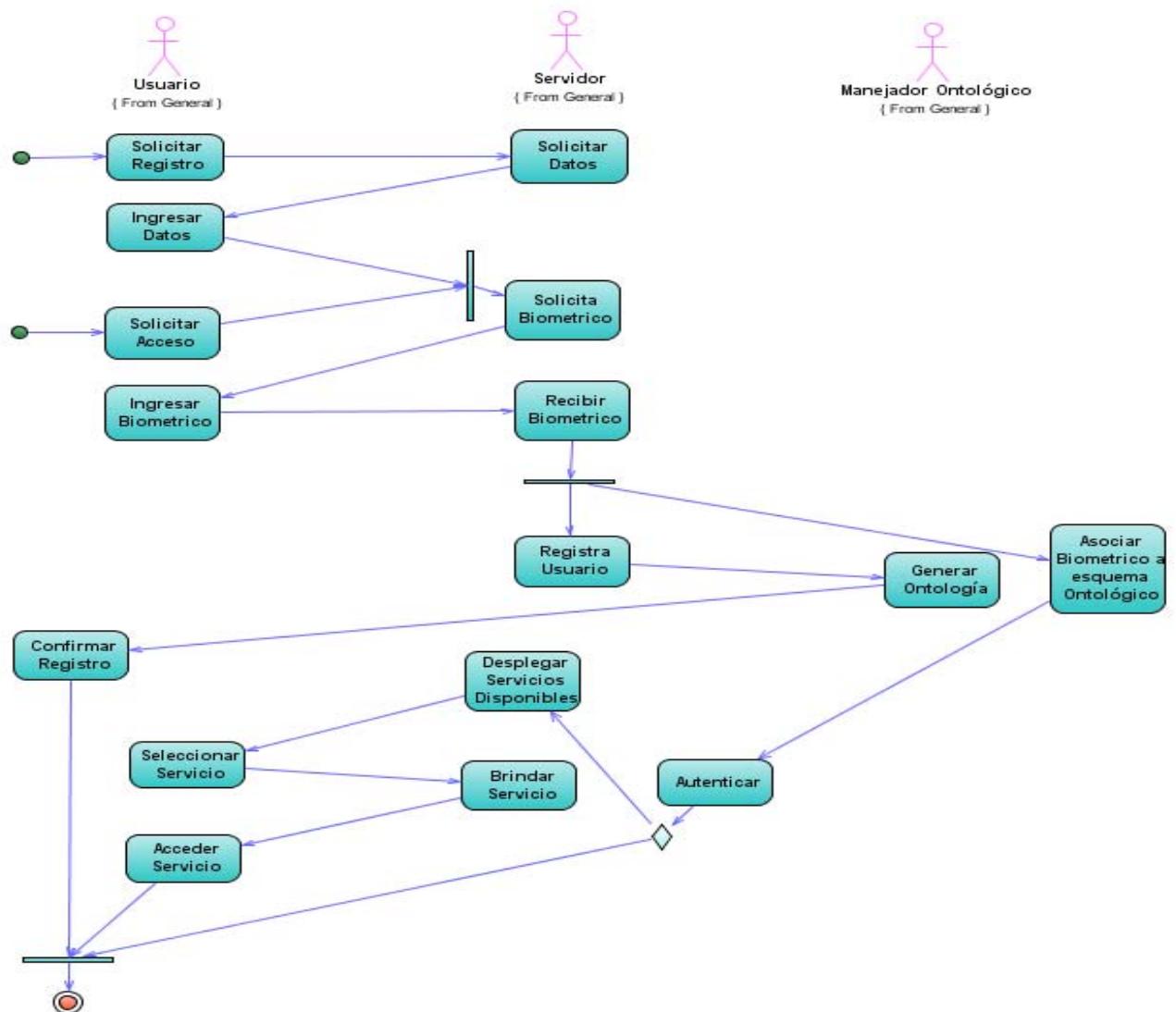


Figura 7. Diagrama de Actividades

Actividad	Descripción
Solicitar Registro	Un Administrador solicita registrar los datos de un usuario nuevo y genera una petición al servidor.
Solicitar Datos	El servidor recibe una petición de registro y primero pide los datos del usuario.
Ingresar Datos	El Administrador captura los datos del nuevo usuario y los manda al servidor.
Solicitar Acceso	Un usuario registrado pide ser autenticado al servidor..
Solicitar Biométrico	El servidor pide que se ingrese el biométrico del usuario y brinda el soporte para ello.
Ingresar Biométrico	El usuario introduce su voz por medio del navegador que transforma las frecuencias sonoras en un archivo VoXML.
Recibir Biométrico	El servidor recibe el biométrico. Si la petición inicial es de registro, entonces pasa a la actividad Registrar Usuario, de lo contrario pasa a la actividad Asociar biométrico a Esquema Ontológico.
Registrar	Con los datos del usuario y el biométrico se crea un registro de un

Usuario	nuevo usuario en la base de datos.
Generar Ontología	Teniendo los datos del usuario, se genera la Ontología propia del usuario por el Manejador Ontológico.
Confirmar Registro	El usuario es notificado del resultado de su registro y este se da por enterado de este resultado y se termina la petición.
Asociar Biométrico a esquema ontológico.	Se busca dentro de las definiciones ontológicas, aquella que sea la del biométrico introducido por el usuario.
Autenticar	El servidor, dependiendo de la respuesta que le de el Manejador Ontológico de la búsqueda del biométrico en los esquemas ontológicos, brinda o no acceso a los servicios del usuario.
Desplegar Servicios disponibles	El servidor proporciona al usuario autenticado una lista de servicios disponibles para dicho usuario.
Seleccionar Servicio	El usuario selecciona de una serie de servicios disponibles, aquel que satisfaga sus necesidades.
Brindar Servicio	El servidor despacha el servicio solicitado por el usuario.
Acceder Servicio	Una vez recibido el servicio solicitado, el usuario accede a él y lo utiliza para posteriormente finalizar la petición.

Tabla 9. Descripción de Actividades de Arquitectura.

2.4.2 MODELADO DE LA CAPA DE USUARIO

2.4.2.1 DIAGRAMA DE CASOS DE USO

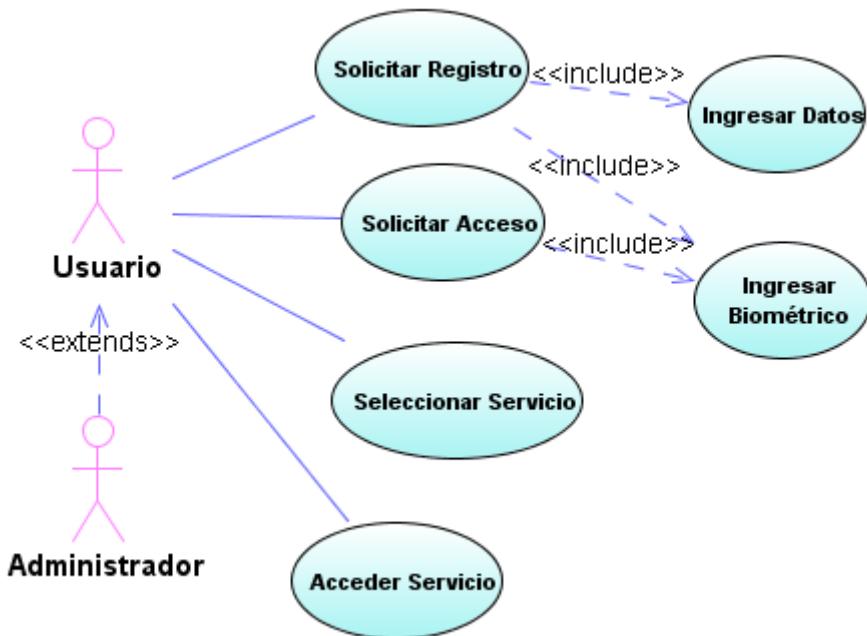


Figura 8. Diagrama de Casos de Uso. Capa de Usuario

<i>Caso de Uso</i>	<i>Descripción</i>
Ingresar Datos	El Administrador introduce los datos personales del usuario a registrar. Los datos son los de identificación oficial, contenidos en la credencial para votar.
Ingresar Biométrico	Se da soporte para la captura de biométrico y el usuario introduce dicho biométrico.

Tabla 10. Descripción de Casos de Uso. Capa de Usuario

2.4.2.2 DIAGRAMA DE SECUENCIA

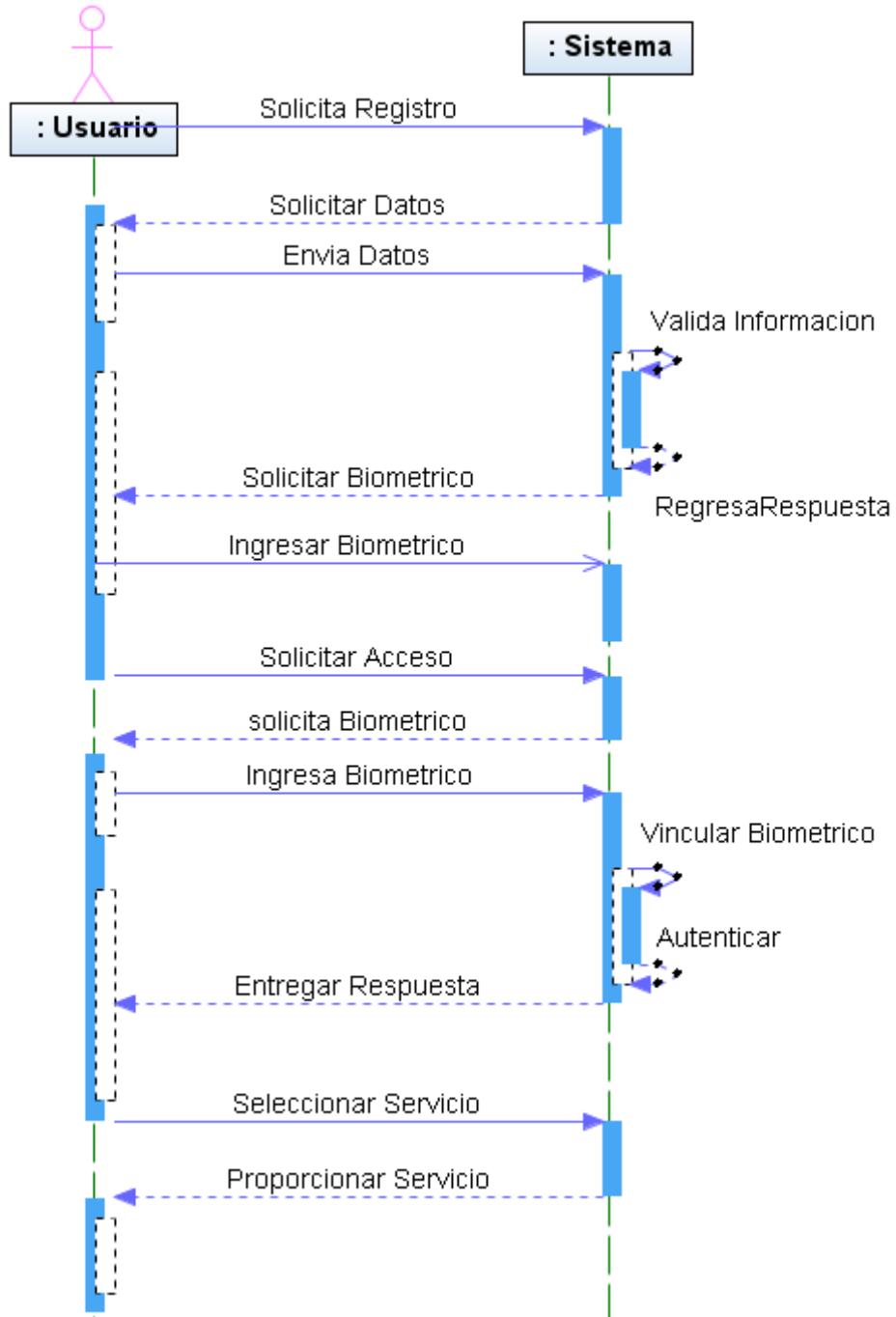


Figura 9. Diagrama de Secuencia. Capa de Usuario

Mensajes	Descripción
Solicita registro	El usuario manda una solicitud de registro al sistema
Solicitar Datos	El sistema pide la información personal del usuario
Envía Datos	Envía los datos personales del usuario
Valida información	Verifica que la información proporcionada por el usuario sea válida y que cumpla con los criterios especificados
Regresa respuesta	Regresa resultado de la validación de la información proporcionada por el usuario
Solicitar Biométrico	Pide al usuario el biométrico que será analizado
Ingresar Biométrico	Registra el biométrico introducido por el usuario
Solicitar Acceso	Realiza una petición de acceso al sistema
Solicita Biométrico	El sistema pide la inserción de un biométrico para acceder al sistema
Ingresa Biométrico	El usuario introduce biométrico para hacer petición
Vincula biométrico	Asocia el biométrico ingresado por el usuario a ontología
Autenticar	Realiza el proceso de verificar la existencia de ontología del usuario
Entregar respuesta	Devuelve el resultado de autenticar usuario para permitir o denegar acceso.
Seleccionar servicio	El usuario elige el servicio deseado
Proporcionar servicio	El sistema provee el servicio escogido por el usuario

Tabla 11. Descripción de Secuencia. Capa Usuario

2.4.3 MODELADO DE LA CAPA DE NAVEGACIÓN

2.4.3.1 DIAGRAMA DE CASOS DE USO

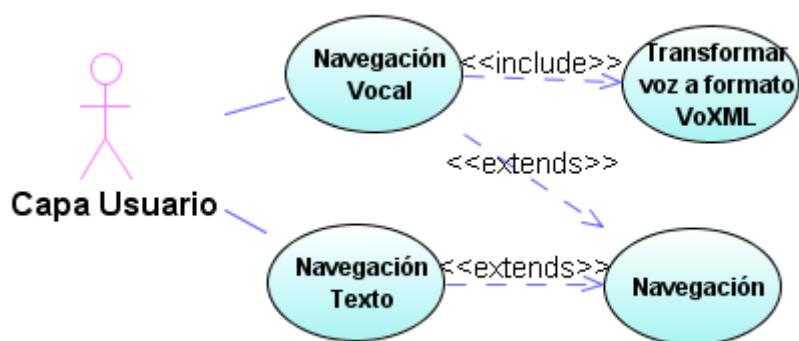


Figura 10. Diagrama de Casos de Uso. Capa de Navegación

<i>Actores</i>	<i>Descripción</i>
Capa Usuario	Se refiere al resultado de la interacción del actor principal Usuarios en la capa anterior.

Tabla 12. Descripción de Actores. Capa de Navegación

Caso de Uso	Descripción
Navegación	Caso de uso encargado de brindar el soporte necesario a través de un navegador Web para la navegación del usuario en el Internet.
Navegación Vocal	Es una extensión del caso de uso anterior. Este caso de uso tiene la particularidad de ser el encargado de dar soporte al manejo de la voz.
Navegación Texto	Es una extensión del caso de uso Navegación. Este caso de uso brinda soporte para la captura de datos textualmente.
Transformar voz a formato VoXML	Se encarga de trasformar la voz a un formato de transporte de datos por Internet, en este caso archivos VoXML.

Tabla 13. Descripción de Casos de Uso. Capa de Navegación

2.4.3.2 DIAGRAMA DE SECUNCIA

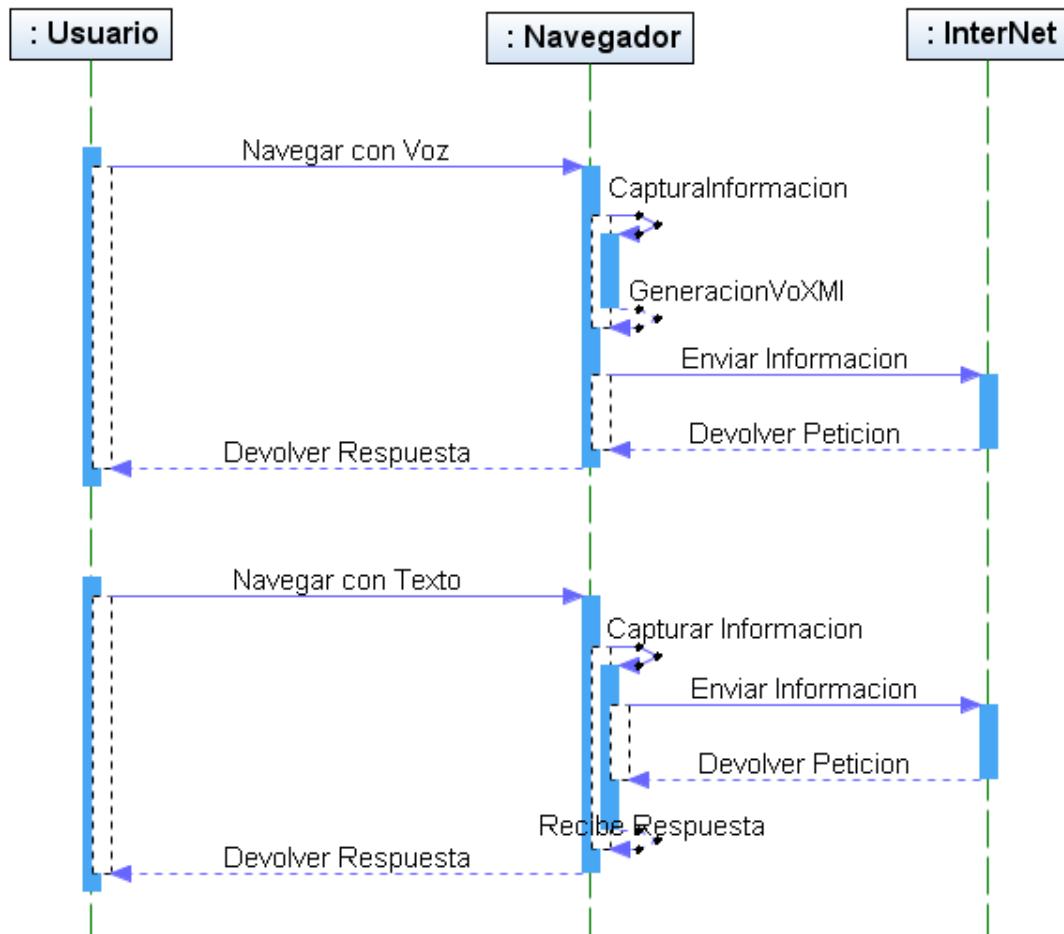


Figura 11. Diagrama de Secuencia. Capa de Navegación

Mensajes	Descripción
Navegar con voz	El usuario navega por medio de la voz
Captura información	El navegador recoge la información introducida por el usuario
Generación VoXML	La información (voz) es almacenada en un documento VoXML
Enviar información	El navegador envía el documento VoXML hacia la zona de internet
Devolver Petición	Regresa al navegador el resultado de la petición del usuario
Devolver respuesta	Regresa respuesta al usuario de la petición realizada
Navegar con texto	El usuario realiza una consulta o registro mediante un navegador
Capturar información	El navegador almacena la información proporcionada por el usuario
Enviar información	La información es enviada a la zona de internet.

Tabla 14. Descripción diagrama de Secuencias. Capa de Navegación

2.4.4 MODELADO DE LA CAPA ZONA DE INTERNET

2.4.4.1 DIAGRAMA DE CASOS DE USO

II.3.4.1 Diagrama de Casos de Uso.

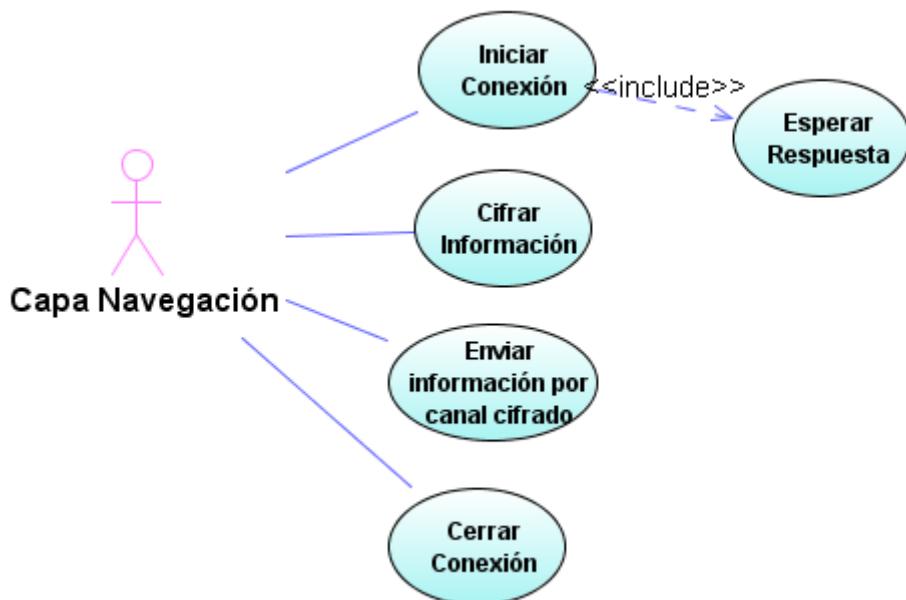


Figura 12. Diagrama de Casos de Uso. Capa Zona de Internet

<i>Actores</i>	<i>Descripción</i>
Capa Navegación	Actor que representa a la capa de anterior de la arquitectura y es quien actúa con esta nueva capa.

Tabla 15. Descripción de Actores. Capa Zona de Internet

Caso de Uso	Descripción
Iniciar Conexión	Se encarga de abrir una vía de comunicación entre el Cliente y el Servidor. Esto se hace abriendo una conexión de ambos lados del canal de comunicación.
Cifrar Información	Destinado al cifrado de la información enviada por el canal con el propósito de brindar mayor fidelidad en los datos transportados.
Enviar Información por canal cifrado	Destinado a brindar una canal de información cifrado para el envío de los datos y/o biométrico de usuarios.
Cerrar Conexión	Cierra ambos lados del canal de comunicación para asegurar que la transacción se termine y liberar así los recursos ocupados en el servidor y el usuario.
Esperar Respuesta	Se queda a la escucha de una posible petición. Mientras dicha petición no exista se permanece en un estado de espera.

Tabla 16. Descripción de Casos de Uso. Capa Zona de Internet

2.4.4.2 DIAGRAMA DE SECUNCIA

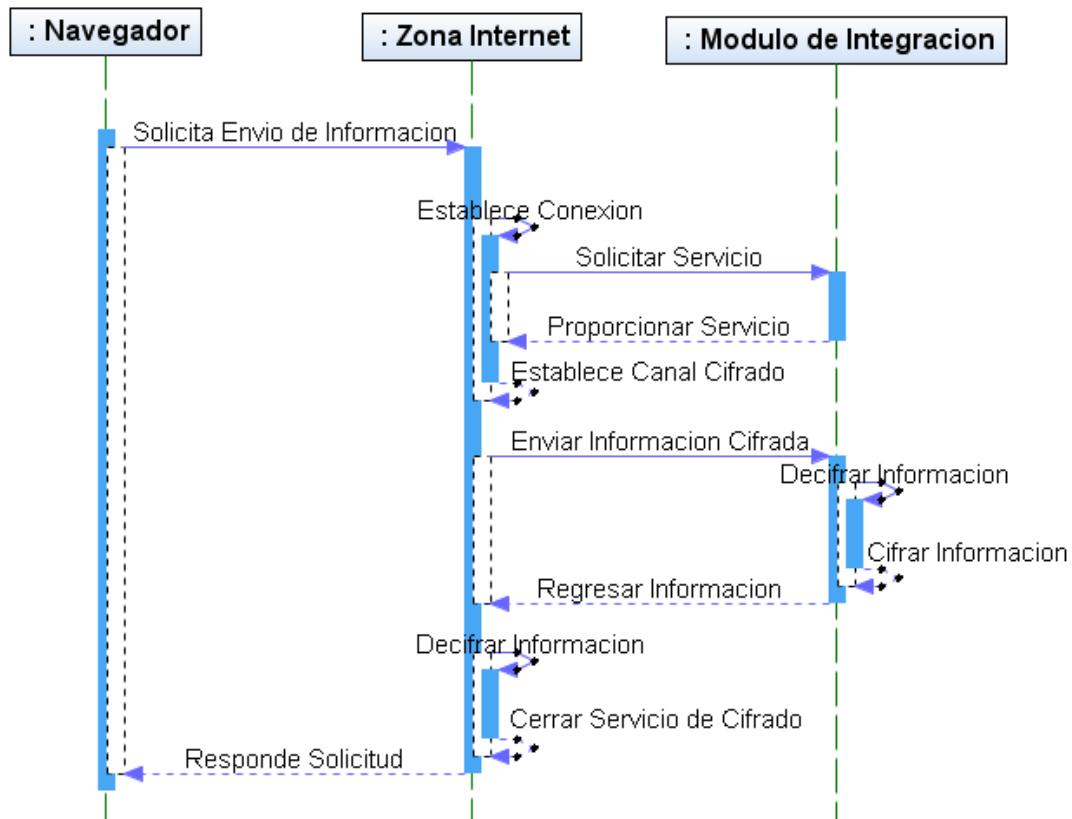


Figura 13. Diagrama de Secuencia. Capa Zona de Internet

Mensajes	Descripción
Solicita envío de Información	El navegador realiza una solicitud a la zona de internet para el envío de la información
Establece conexión	El navegador pide iniciar conexión para el envío de datos
Solicita servicio	Se solicita servicio para accesar al manejador ontológico
Proporciona servicio	El manejador atiende a la petición de la capa de internet
Establece Canal cifrado	Se cifra la información enviada por el navegador para proporcionar seguridad
Enviar información cifrada	El documento cifrado es enviado al manejador ontológico
Descifrar información	La información es traducida para su posterior tratamiento
Cifrar información	El documento es cifrado por el manejador ontológico para su transporte
Regresar información	Se regresa el documento hacia la zona de internet
Cerrar Servicio de cifrado	Una vez finalizada la traducción de la información se finaliza el servicio

Tabla 17. Descripción de diagramas de secuencia. Capa Zona de Internet

2.4.5 MODELADO DE LA CAPA DE INTEGRACIÓN

2.4.5.1 DIAGRAMA DE CASOS DE USO

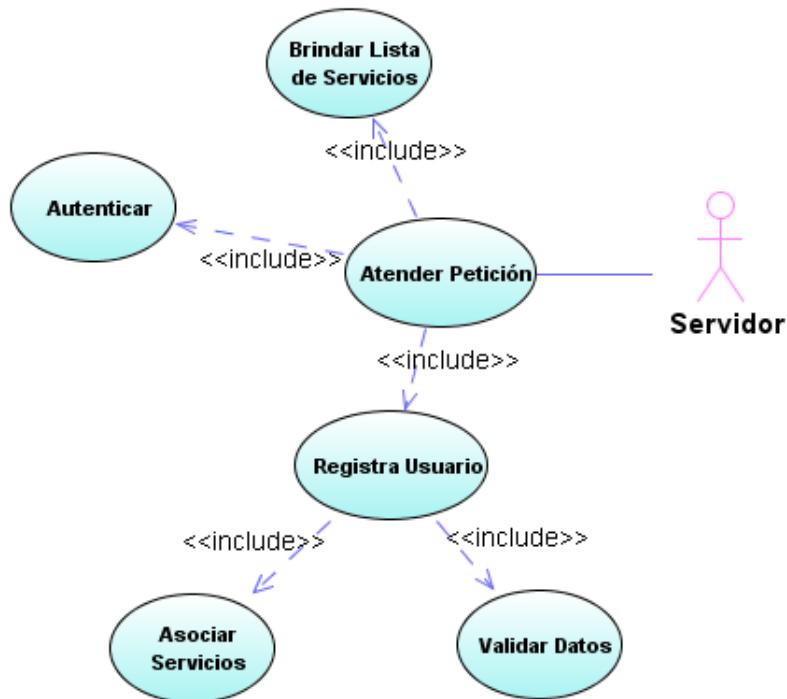


Figura 14. Diagrama de Casos de Uso. Capa de Integración

Caso de Uso	Descripción
Asociar Servicios	Depuse de registrado el usuario, se le asignan de manera predefinida un determinado número de servicios dependiendo del tipo de usuario que sea.
Validar Datos	Se encarga de verificar si los datos introducidos por el usuario cumplen con las características necesarias.

Tabla 18. Descripción de Casos de Uso. Capa de Integración

2.4.5.2 DIAGRAMA DE SECUNCIA

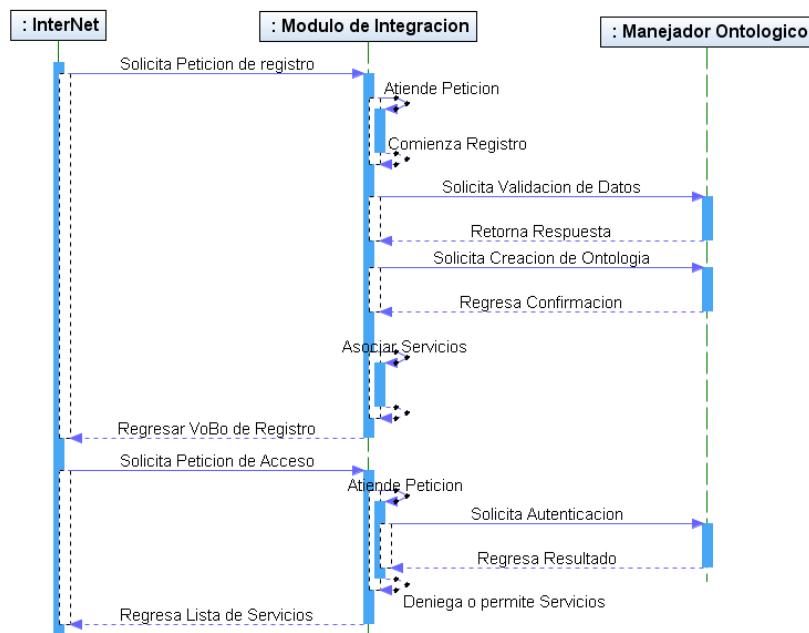


Figura 15. Diagrama de Secuencia. Capa de Integración

Mensajes	Descripción
Solicita Petición de registro	La capa de internet, lanza una petición para el registro de un usuario.
Atiende Petición	El modulo de integración atiende petición y la canaliza.
Comienza Registro	Dado que es una petición de registro el modulo comienza dicho proceso.
Solicita Validación de Datos	El modulo de integración solicita al manejador ontológico comprobar la validez de la información introducida
Retorna Respuesta	El manejador ontológico regresa una respuesta a la primera petición de validación de datos.
Solicita creación de ontología	El modulo de integración solicita al manejador ontológico la creación de la ontología ya con información consistente.
Regresa Confirmación	El manejador ontológico crea la ontología y manda al modulo de integración la confirmación de la alta.
Asocia Servicios	El modulo de integración asocia sus servicios al usuario recién ingresado.
Regresar VoBo de Registro	El modulo de integración regresa un VoBo a través de Internet para notificar el éxito del registro.
Solicita Petición de Acceso	De la zona de internet llega una petición de acceso por parte de un usuario ya registrado.

Solicita Autenticación	El modulo de integración lanza una solicitud de autenticación al manejador ontológico para permitir o denegar el acceso.
Regresa Resultado	Dependiendo del resultado de la petición se regresa un resultado al usuario.
Deniega o permite servicios	Si el resultado es positivo se permite el acceso a los Servicios solicitados de lo contrario se deniega el mismo.
Regresa lista de Servicios	Si los servicios fueron otorgados se regresa una lista de servicios a través de internet.

Tabla 19. Descripción de diagramas de secuencia. Capa de Integración

2.4.6 MODELADO DE LA CAPA ONTOLÓGICA

2.4.6.1 DIAGRAMA DE CASOS DE USO

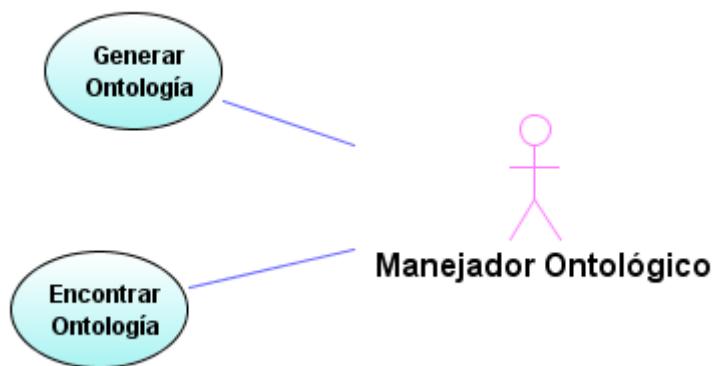


Figura 16. Diagrama de Casos de Uso. Capa Ontológica

Nota: Véase tabla 8.

2.4.6.2 DIAGRAMA DE SECUNCIA

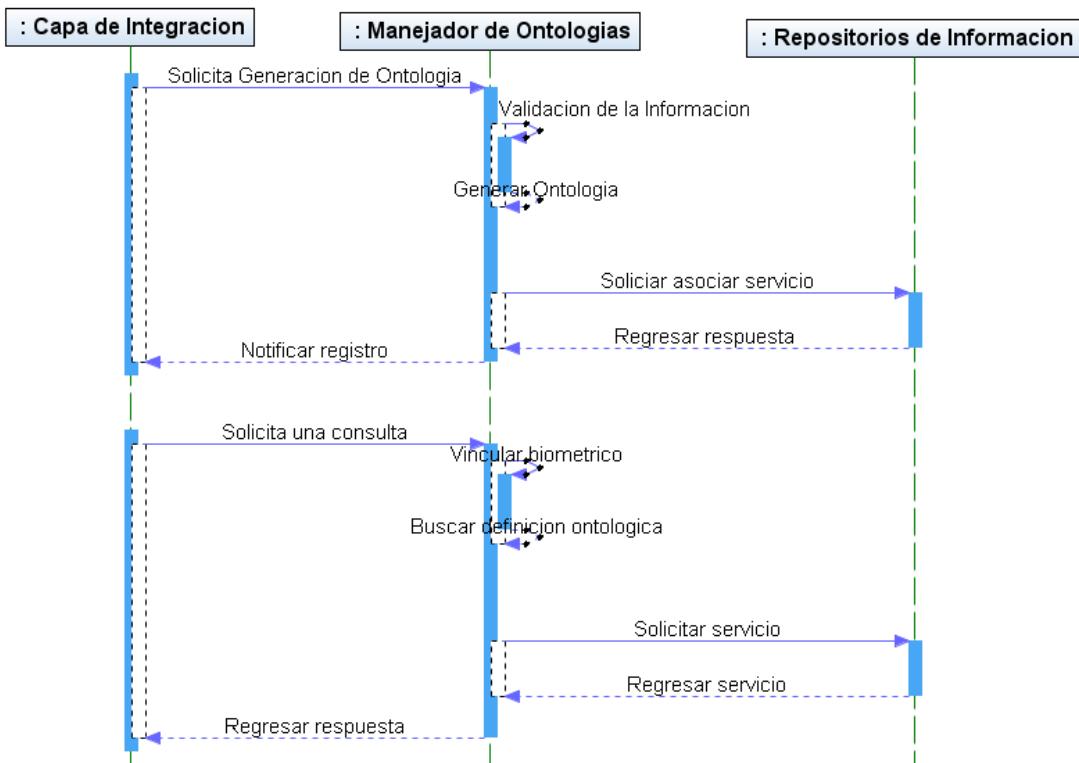


Figura 17. Diagrama de Secuencia. Capa Ontológica

Mensajes	Descripción
Solicita Generación de Ontología	La capa de integración solicita al manejador de ontologías la generación de una nueva ontología.
Validación de la información	El manejador ontológico valida la información
Generar Ontología	El manejador ontológico genera la ontología
Solicitar asociar Servicios	El manejador ontológico solicita al repositorio de información la asociación a uno de sus servicios.
Notificar registro	El manejador de ontologías notifica a la capa de integración la alta de la nueva ontología.
Solicita consulta	La capa de integración lanza una consulta al manejador ontológico con el fin de verificar la existencia del usuario.
Vincular biométrico	El manejador ontológico vincula el biométrico introducido con su definición, a fin de poder hacer la búsqueda.
Buscar definición ontológica	Ya que se hizo la vinculación se procede a la búsqueda de la definición ontológica dentro de las ontologías registradas.
Solicita servicio	El manejador ontológico solicita un servicio al repositorio de servicios para vincularlo con el usuario encontrado.
Regresar servicio	Los repositorios de información regresan los servicios vinculados.
Regresar Respuesta	El manejador ontológico regresa a la capa de integración toda la serie de servicios asociados que se encontraron.

Tabla 20. Descripción de diagrama de secuencia. Capa Ontológica

2.5 DISEÑO CONCEPTUAL DE BASE DE DATOS

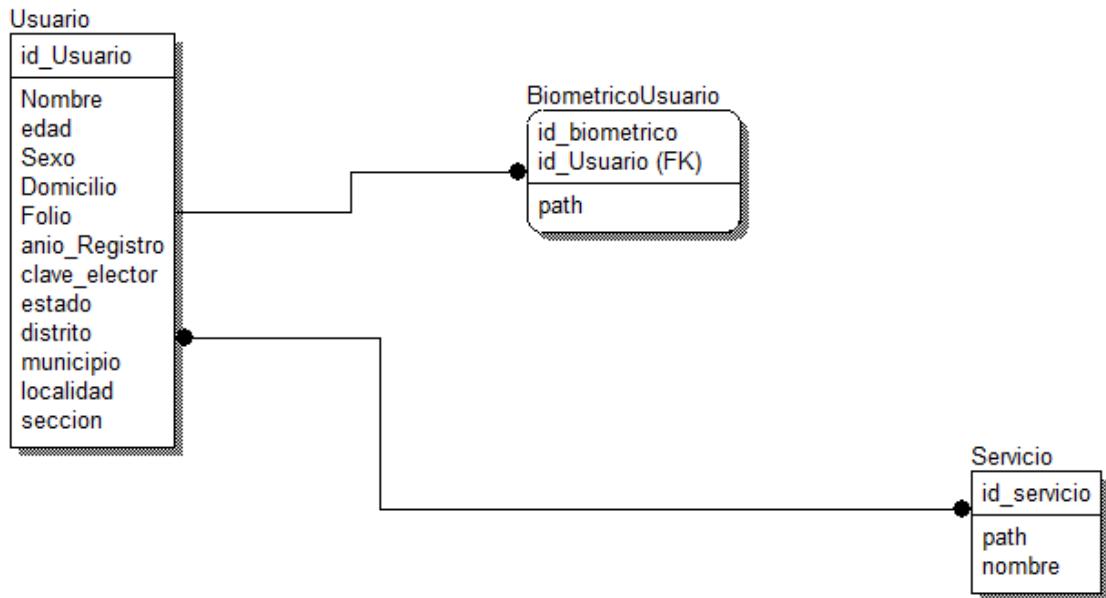


Figura 18. Diseño conceptual de la base de datos

Las relaciones empleadas para el modelado de nuestra base de datos son el mapeo directo de nuestro diagrama de clases, donde necesitamos la definición de nuestro usuario, y los biométricos y servicios asociados a este.

La descripción de cada uno de estas relaciones viene dada de manera explícita en la descripción del diagrama de clases (Véase Figura 6).

2.6 DISEÑO FÍSICO DE BASE DE DATOS

El diseño físico de la base de datos es obtenido de manera automática, quedando al final las siguientes relaciones.

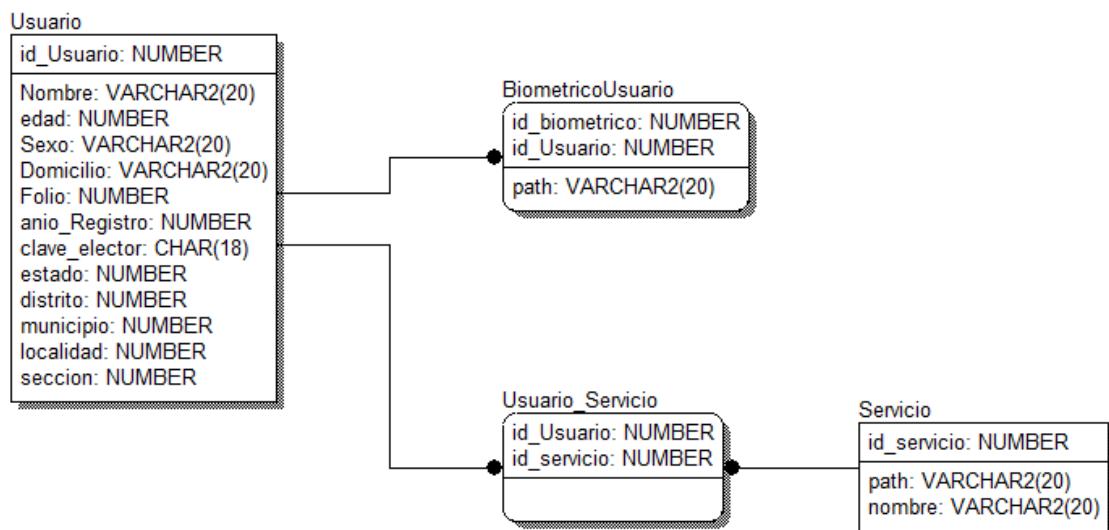


Figura 19. Diseño físico de la base de datos

CAPÍTULO 3. DISEÑO

Habiendo definido claramente los requerimientos en la tapa de análisis y definidas las necesidades de la arquitectura, pasamos a la fase de diseño dentro de la cual describiremos detalladamente los componentes que integran la arquitectura, las relaciones entre ellos, el intercambio de información, las interfaces y estructura de la misma.

3.1 MODELADO DE LA ARQUITECTURA

La arquitectura que se pretende desarrollar esta basada en un modelo multicapa, a continuación se describe el diseño de esta y las transiciones entre sus diferentes capas.

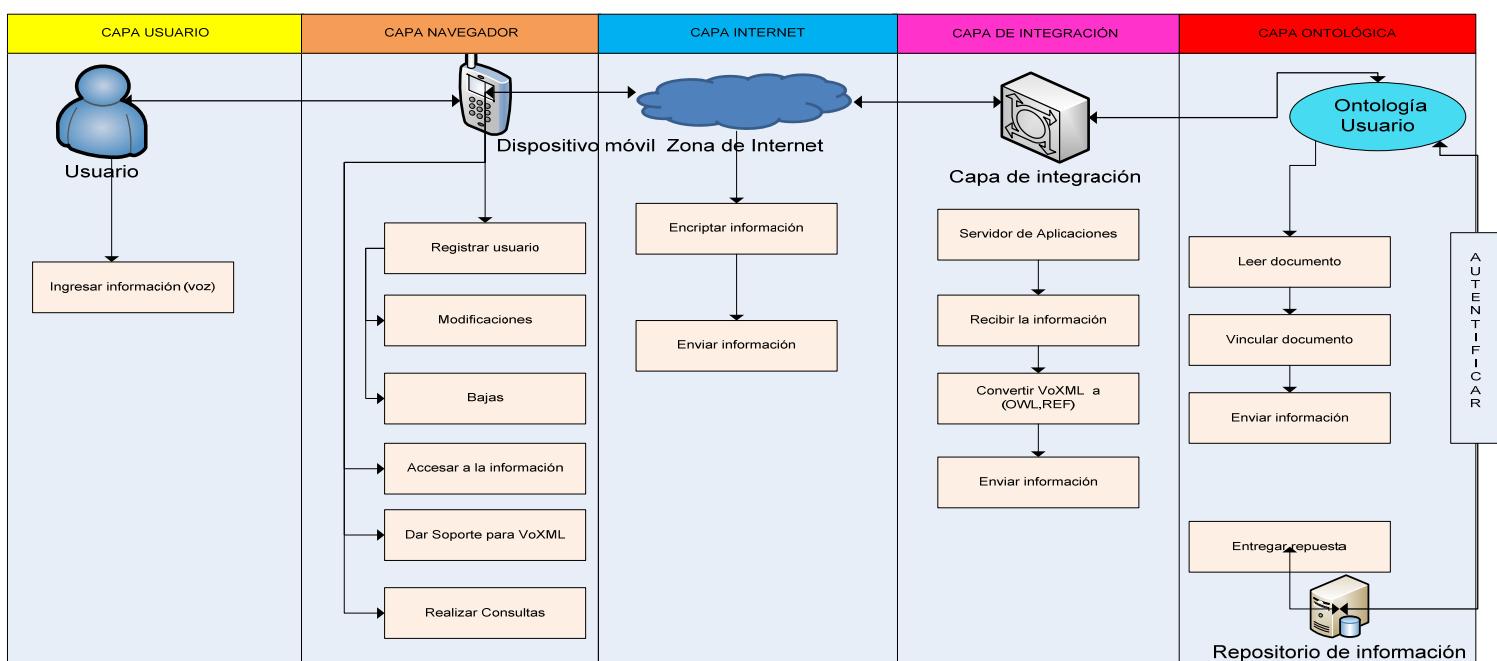


Figura 20. Diagrama de Arquitectura

La Arquitectura aquí propuesta se describe a continuación.

3.1.1 CAPA DE USUARIO

En la capa de usuario tenemos la interacción directa de cualquier usuario del sistema, el cual ingresa su información (tanto datos de texto como voz) a través de un navegador de Internet.

3.1.2 CAPA DE NAVEGACIÓN

En esta capa tenemos un navegador con características específicas como lo son:

- **Soporte para la captura de la voz del usuario.**
- **Generación de documentos VoXML a partir de la captura de la misma.**

Por medio de esto navegador se ofrecerán los siguientes servicios:

➤ **Acceso a la información:**

Este es el más importante de los Servicios que se proporcionara, ya que permitirá tener acceso a los recursos o servicios que posee el usuario a través de su identificación y autenticación en línea:

➤ **Registro de usuarios:**

Esta opción permitirá capturar los datos del usuario a fin de poder crear su definición ontológica del mismo, para brindar más seguridad al registro de usuarios este puede hacerse a través de internet, pero solo ciertos usuarios podrán capturar la información para que esto brinde más seguridad y fiabilidad a la información, o se podrá hacer de manera local por medio de un organismo calificado para poder hacer el registro correspondiente de los usuarios.

➤ **Modificación y edición de información de usuario:**

Esta opción permitirá a los administradores del sistema modificar la información ya capturada de un usuario, de esta forma por medio de esta autoridad calificada se podrá tener un control más estricto y seguro de la información almacenada.

➤ **Baja de registros de usuarios:**

Esta opción permitirá a los administradores del sistema eliminar registros de usuarios, de esta forma por medio de esta autoridad calificada se podrá tener un control más estricto y seguro de la información almacenada.

➤ **Consulta de información:**

Esta funcionalidad permitirá al usuario poder realizar consultas acerca de su información almacenada si como también de los servicios que se le proporcionan en línea por medio de su identificación e autentificación.

3.1.3 CAPA ZONA DE INTERNET

En esta capa se tienen dos que realizar dos funcionalidades y características en concreto:

Cifrar la información: este procedimiento es el encargado de crear un canal de comunicación cifrado para el envío recepción de la información que previamente se ha capturado por el navegador, a fin de brindar seguridad a los datos ya que estos son información sensible y de no proveer esta funcionalidad dentro de la arquitectura esta puede ser vulnerable a ataques informáticos.

Enviar información: ya cifrada la información esta se transmitirá al servidor para su posterior tratamiento.

3.1.4 CAPA DE INTEGRACIÓN

Esta capa es la más importante de toda la arquitectura ya que es dentro de esta que se harán los principales procesos de la misma, para lo cual necesitamos:

➤ **Servidor de aplicaciones:**

Sera el encargado de procesar las peticiones provenientes del cliente y de responder y procesar cada una de ellas de acuerdo a sus necesidades y requerimientos, los procesos principales que ejecutara el servidor de aplicaciones son los siguientes:

Recepción y des encriptación de la información:

Esta tarea en primer lugar recibe la información, posteriormente analiza su integridad, ya verificado esto, se procede a la de encriptación de la información.

Convertir la información:

A fin de convertir e un formato VoXML a OWL o equivalente para poder trabajar con el manejador de ontologías.

Enviar la información:

Ya convertido el documento, se enviara para que este pueda se procesado por el manejador ontológico

3.1.5 CAPA ONTOLOGICA

Esta capa tiene la responsabilidad de hacer el proceso de autenticación e identificación de los individuos para lo cual es auxilia de las siguientes tareas:

Lectura del documento:

Tiene como función leer el documento que se genera para poder interactuar de manera directa con el manejador ontológico

Vinculación de documento:

El manejador ontológico es el responsable de hacer este proceso que vincula el documento con su definición dentro de la ontología general del usuario.

Consulta de autentificación:

De manera directa esta tarea tiene como objetivo realizar la búsqueda dentro de nuestra base de conocimiento la definición de la ontología a fin de corroborar su existencia.

Entrega de respuesta:

Dependiendo del resultado de la consulta que se realice en esta etapa del proceso se permite o deniega el acceso al recurso o servicio que se desea.

3.2 DISEÑO MODULAR

Diseño Modular

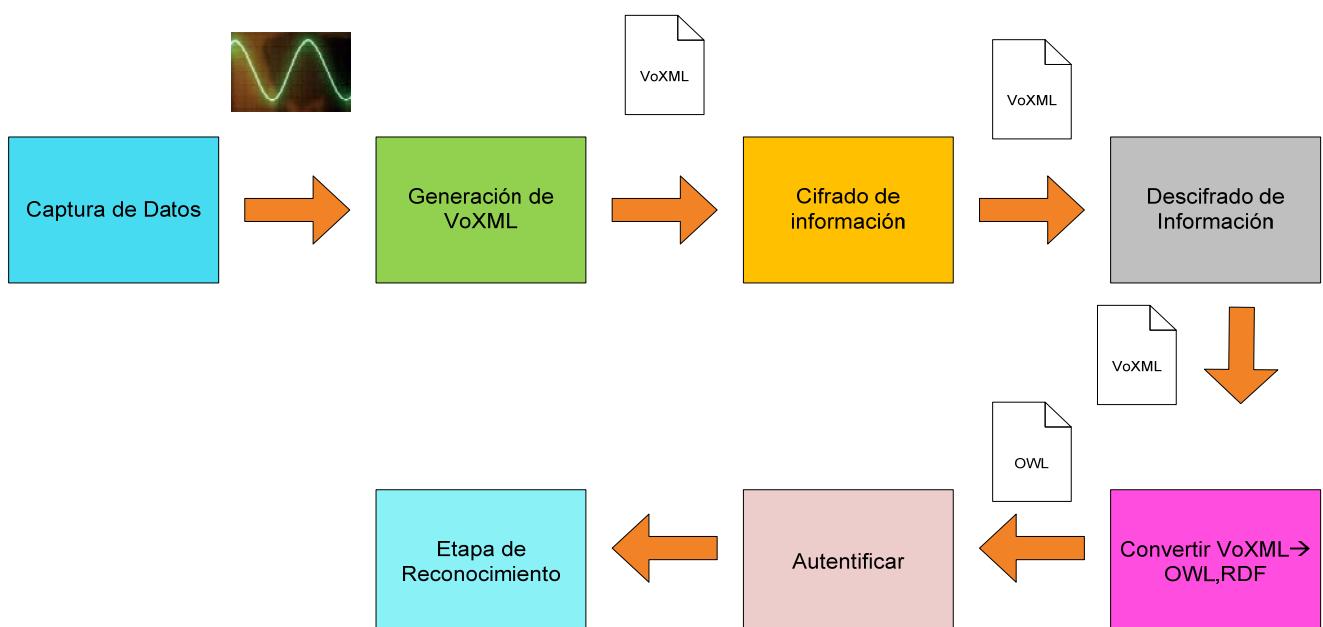


Figura 21. Diseño Modular.

CAPTURA DE DATOS

Este modulo será el encargado de recolectar la información del usuario, lo cual se hará a través de un interfaz grafica que será mostrada en un navegador vía Internet, la cual solicitará al usuario introducir datos con diferentes fines como lo son:

Registro de usuario.

Eliminación de usuario.

Modificación de información de usuarios.

Autentificar e identificar usuario.

GENERACIÓN DE VOXML

Este modulo tiene la función de a partir de la voz capturada generar un documento Vo XML que será tratado posteriormente.

CIFRADO DE LA INFORMACIÓN

La principal función de este proceso es el de cifrar la información y crear un canal de comunicación seguro para poder enviar de manera segura la información.

DESCIFRADO DE LA INFORMACIÓN

En este modulo la información tiene como objetivo descifrar la información que ha sido mandada por el canal seguro a fin de poder trabajar con la información que se ha mandado íntegramente y sin ningún ruido.

CONVERTIR VOXML A OWL

La funcionalidad de este modulo es la de a partir del documento VoXML que se ha recibido poder convertir ese documento que contiene la voz del usuario sobre la que se harán las pruebas para la autenticación, en un formato de definición de ontologías con la finalidad de poder por medio del manejador ontológico inferir la existencia de dicha definición dentro de la ontología del usuario.

AUTENTICAR

Este proceso es la medula espinal de todo ya que a partir del documento que se genera para poder consultar la definición de la voz del usuario dentro de su definición ontológica, para esto el manejador ontológico buscara dentro de la definición ontológica del usuario la especificación de voz del usuario para ver si concuerda con la que se está recibiendo.

ETAPA DE RESPUESTA

Dentro de esta etapa solo se entrega la respuesta que se genera a partir de la ejecución del modulo anterior, si se encuentra la definición se entrega el acceso al recurso solicitado de lo contrario se manda un mensaje de notificación indicando que el usuario no pudo ser identificado.

3.3 DISEÑO E/S

Dado que tenemos una arquitectura tenemos documentos que se van generando y a lo largo del recorrido por las distintas capas estos van cambiando y convirtiéndose en salidas pero también en entradas al hacerse la transición entre las diferentes etapas, es por ello que definimos las estradas y salidas de manera general y podemos Ver que rol fungen (entrada o salida) observado el esquema del diseño modular de la arquitectura.

ONDAS DE SONIDO

Esta es la primera entrada del sistema, ya que son las que genera el usuario al interactuar con el sistema, el cual captura en primera instancia este tipo de información para posteriormente tratarla.

VoXML

Este documento es generado por el navegador compatible con esta tecnología a partir de la captura de la voz esta estará contenida de manera integra dentro de este documento.

VoXML CIFRADO

Este documento se genera al cifrar la información para transmitirla por un medio cifrado, con la finalidad de encapsular el documento y protegerlo de posibles ataques en contra de la integridad del documento.

OWL

Este documento es propiamente el lenguaje que entiende un manejador ontológico para poder hacer la vinculación y búsqueda de la información contenida dentro de la misma.

XML RESPUESTA

Es el documento que se genera para dar respuesta de la petición de autentificación que se genero indicando si se pudo autenticar e identificar el individuo y en caso de que si se haya podido hacer también contiene el acceso al recurso o servicio que se pretendía acceso.

CAPÍTULO 4. DESARROLLO Y CODIFICACIÓN

4.1 CAPA DE USUARIO

En esta capa se definen los roles de usuarios. La capa es flexible al tipo de organización en la que se deseé implementar. En nuestro caso, definimos que habrá dos roles principales:

- Administrador.- Se encarga de dar de alta a los usuarios.
- Usuario.- Accede al sistema mediante la autenticación de voz.

El prototipo implementa un proceso de enrolamiento de usuarios, el cuál se deberá realizar 3 veces, esto a fin de obtener un mejor rango en los coeficientes cepstrales de Mel, ya que nos permitirá obtener la voz en diferentes entornos o condiciones y evaluar diferencias en el habla.

4.2 CAPA DE NAVEGACIÓN

La capa de navegación debía estar soportada por un navegador para aplicaciones Multimodal (visual y vocal), para esto se eligió Opera.

Opera es un navegador web y suite de Internet. La aplicación es gratuita desde su versión 8.50. Es reconocido por su gran velocidad, seguridad, soporte de estándares (especialmente CSS), tamaño reducido, internacionalidad y constante innovación.

Una de sus principales características por la cual fue elegido es el soporte de voz, permite el control del navegador hablándole y también generar fragmentos hablados a partir del texto de las páginas Web.

Para implementar esta capa se recurrió a varias herramientas que posibilitaran una interacción rica y atractiva entre el usuario y la aplicación es por ello que empleamos tecnologías web como XML, XHTML+Voice y en esta capa en particular una parte de un framework de java llamado JSF, dado de que JSF implementa el patrón MVC, en esta capa solo se implementa la vista, el modelo se aplica en la capa de integración y se precisara su aplicación mas adelante, como ya se mencionó se aplicó la parte de visual que nos ofrece el modelo de nuestras páginas con JSP y con la librería de componentes visuales web conocida como CoreJSF.

El usuario debe introducir su contraseña por medio del habla interactuando con el navegador, este a su vez hará la recepción de voz en un buffer, para posteriormente enviarla a un servlet y que el archivo de audio sea guardado.

4.2.1 TÉCNOLOGIA EMPLEADA

Opera utiliza XHTML+Voice para las aplicaciones Multimodal, así como soporte para CSS.

Para el diseño de la interfaz del login se uso XHTML+Voice que es un lenguaje de marcado que permite crear las llamadas Multimodal Application, consiste en un

proceso en el cual diversos dispositivos y personas son capaces de llevar a cabo una interacción (auditiva, visual, táctil y gestual) conjunta desde cualquier sitio, en cualquier momento, utilizando cualquier dispositivo y de forma accesible. Con XHTML+Voice, se pueden crear páginas Web que permiten a los usuarios finales el uso de voz como entrada y salida y al mismo tiempo tener la tradicional interfaz visual. Este lenguaje está estandarizado por la W3C.

Está basado en:

- **XHTML** eXtensible Hypertext Markup Language (lenguaje extensible de marcado de hipertexto), es el lenguaje de marcado pensado para sustituir a HTML como estándar para las páginas web. En su versión 1.0, XHTML es solamente la versión XML de HTML, por lo que tiene, básicamente, las mismas funcionalidades, pero cumple las especificaciones, más estrictas, de XML.
- **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).
- **CSS:** lenguaje de marcado para dar propiedades visuales a la aplicación
- **HTML:** lenguaje de marcado para generar páginas web.
- **JSP:** lenguaje que adiciona tag específicas y de posibles librerías para poder insertar código java en las páginas html.
- **JavaScrip:** lenguaje interpretado que da funcionalidades extras a las páginas web ejecutándose del lado del cliente.

4.2.2 CODIFICACIÓN

La implementación de la presente capa se realizó con el uso de JSF, y dos librerías de etiquetas que nos sirvieron para poder dar la funcionalidad optima a la aplicación que se desarrolló:

❖ *Librerías empleadas*

- **JSFCore:** brinda etiquetas para dar funcionalidades extras a las páginas JSP.
- **JSFHTML:** serie de etiquetas con componentes visuales de la página que se pueden ligar a datos (beans) y desde ahí poder implementar política de negocios.

4.3 CAPA ZONA DE INTERNET

El tipo de aplicación que desarrollamos requería de transportar los datos de manera segura por medio del protocolo HTTPS (protocolo seguro de transferencia de hipertexto).

El sistema HTTPS utiliza un cifrado basado en las Secure Socket Layers (SSL) para crear un canal cifrado (cuyo nivel de cifrado depende del servidor remoto y del navegador utilizado por el cliente) más apropiado para el tráfico de información sensible que el protocolo HTTP. De este modo se consigue que la información sensible (usuario y claves de paso normalmente) no puede ser usada por un atacante que haya conseguido interceptar la transferencia de datos de la conexión, ya que lo único que obtendrá será un flujo de datos cifrados que le resultará imposible de descifrar.

Los datos que circulan en un sentido y otro entre el cliente y el servidor se cifra mediante un algoritmo simétrico como DES o RC4.

El puerto estándar para este protocolo es el 443.

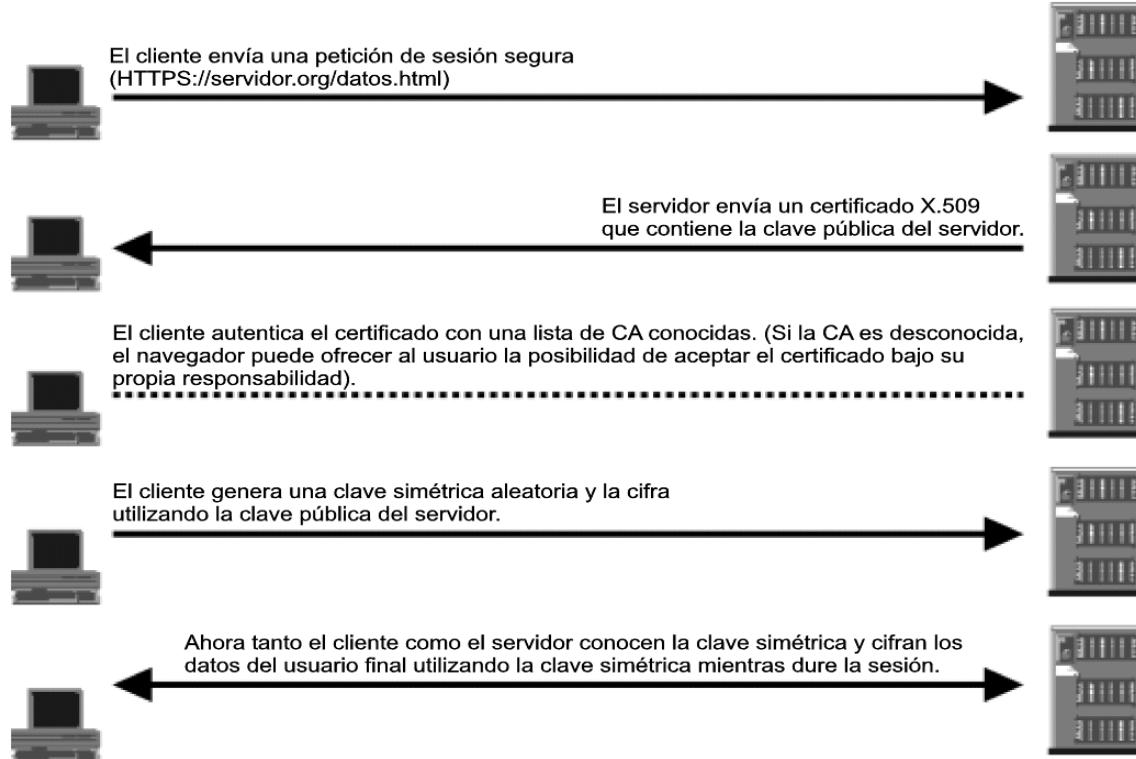


Figura 22. Funcionamiento del HTTPS

4.3.1 TÉCNOLOGIA EMPLEADA

El servidor que se usa es el **Apache Tomcat 6.0**, el cual deberá estar configurado. Apache Tomcat funciona como un contenedor de servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Tomcat implementa las especificaciones de los servlets y de JavaServer Pages (JSP) de Sun Microsystems

4.3.2 CODIFICACIÓN

Para lograr un canal cifrado se siguieron los siguientes pasos:

Conseguir nuestro **certificado digital**. Para generararlo utilizamos la herramienta "**keytool**" que nos proporciona **Java**. Esta utilidad se encuentra en el directorio *bin* de la JDK y se utilizó de la siguiente forma desde línea de comandos:
keytool -genkey -alias tomcat -keyalg RSA

Una vez iniciado el proceso, Nos pidió una serie de datos para configurar el **certificado digital**.



```
[DIANA] C:\$ cd Java
El sistema no puede hallar la ruta especificada.

[DIANA] C:\$ C:\Archivos de programa\Java\jdk1.6.0_12\bin
"C:\Archivos" no se reconoce como un comando interno o externo,
programa o archivo por lotes ejecutable.

[DIANA] C:\$ cd C:\Archivos de programa\Java\jdk1.6.0_12\bin
[DIANA] C:\Archivos de programa\Java\jdk1.6.0_12\bin$ keytool -genkey -alias to
mcat -keyalg RSA
error de keytool: java.lang.RuntimeException: Error de uso, -genkey no es un com
ando legal

[DIANA] C:\Archivos de programa\Java\jdk1.6.0_12\bin$
[DIANA] C:\Archivos de programa\Java\jdk1.6.0_12\bin$
[DIANA] C:\Archivos de programa\Java\jdk1.6.0_12\bin$ keytool -genkey -alias to
mcat -keyalg RSA
Escriba la contraseña del almacén de claves:
Volver a escribir la contraseña nueva:
¿Cuáles son su nombre y su apellido?
[Unknown]: diana
¿Cuál es el nombre de su unidad de organización?
[Unknown]: escom
¿Cuál es el nombre de su organización?
[Unknown]: escom
¿Cuál es el nombre de su ciudad o localidad?
[Unknown]: mexico
¿Cuál es el nombre de su estado o provincia?
[Unknown]: df
¿Cuál es el código de país de dos letras de la unidad?
[Unknown]: mx
¿Es correcto CN=diana, OU=escom, O=escom, L=mexico, ST=df, C=mx?
[no]: si

Escriba la contraseña clave para <tomcat>
(INTRO si es la misma contraseña que la del almacén de claves):
La contraseña clave es demasiado corta; debe tener al menos 6 caracteres
Escriba la contraseña clave para <tomcat>
(INTRO si es la misma contraseña que la del almacén de claves):
Volver a escribir la contraseña nueva:

[DIANA] C:\Archivos de programa\Java\jdk1.6.0_12\bin$
```

Figura 23. Configuración de certificado digital para Apache Tomcat 6.0

Se generó un fichero de claves llamado ".keystore"

Después se activó el conector para soporte SSL en el fichero "server.xml" que encontramos en el directorio "conf" de Tomcat. Para ello se descomentaron las líneas que se encuentran cuando buscamos "**Define a SSL HTTP/1.1 Connector on port 8443**".

```

<!-- A "Connector" using the shared thread pool-->
<!--
<Connector executor="tomcatThreadPool"
    port="8080" protocol="HTTP/1.1"
    connectionTimeout="20000"
    redirectPort="8443" />
-->
<!-- Define a SSL HTTP/1.1 Connector on port 8443
    This connector uses the JSSE configuration, when using APR, the
    connector should be using the OpenSSL style configuration
    described in the APR documentation -->

<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
    maxThreads="150" scheme="https" secure="true"
    clientAuth="false" sslProtocol="TLS" keystoreFile="C:\Archivos de programa\Apache Software Foundation\Tomcat
6.0\.keystore"/>

<!-- Define an AJP 1.3 Connector on port 8009 -->
<Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />

```

Donde keystoreFile="C:\Archivos de programa\Apache Software Foundation\Tomcat 6.0\.keystore" nos indica la ruta de la llave del certificado.

4.4 CAPA DE INTEGRACIÓN

El desarrollo de los componentes que integraban esta capa en principio fue fácil dado que empleamos el API Servlet para el tratamiento de archivos y asimismo nos ayudo el Framework JSF en la parte del modelado de datos para implementar la política de negocio de nuestra aplicación a través de los ManagedBeans.

Todo iba bien pero al ir avanzando nos enfrentamos a varios problemas, dado que la primera recomendación de implementación planteada en el modelo y diseño de la arquitectura resulto no ser la más adecuada, esto se debió a que en principio la arquitectura nos señalaba como primer alternativa el uso de VoiceXML para parte de autentificación de los usuarios puesto que al hacer nuestro trabajo de investigación encontramos que esta tecnología tenia implementada dentro de si esta parte de la arquitectura (autentificación), lo que en la práctica no se pudo hacer ya en el sitio oficial de VoiceXML se publico que esa parte de su proyecto aun no pasaba de ser un borrador, el cual tienen planeado liberar a finales de este año.

Ya aclarado el punto queremos dar a conocer como fue solucionado este problema: En primer lugar se analizo el problema de autentificación por voz y los métodos que actualmente se emplean para abordar este tema, sabemos de antemano que este problema se reduce a extraer características únicas de la voz de un individuo a fin de poder autenticarlo por medio de estos elementos.

Después de una exhaustiva búsqueda acerca de los algoritmos y métodos de extracción de características de la voz humana, optamos por trabajar en la extracción de los Coeficientes Cepstrales en escala Mel (MFCC).

Ahora bien la extracción características es el primer paso en la autentificación, después de esto el siguiente paso, en el caso particular de la extracción de los coeficientes es el siguiente: al extraer los MFCC estos guardan la información extraída en Vectores acústicos, dado que este conjunto de vectores es demasiado extenso, se tiene que compactar, este procedimiento se hace mediante la implementación de un algoritmo de cuantificación vectorial, en nuestro caso empleamos un algoritmo llamado LGB.

Los puntos antes mencionados son una descripción general, pero se mostraran a detalle los algoritmos y la implementación en las secciones posteriores.

❖ ***Algoritmos y procedimientos empleados***

➤ ***Algoritmo empleado para el cálculo de los MFFC***

El procedimiento pasa por varios pasos los cuales detallamos a continuación:

1. Captura de la señal de audio y digitalización y muestreo de la misma

El primer paso es transparente en este punto de la arquitectura puesto que los archivos de audio digitalización provienen de las capas superiores, ya que en esas capas se realiza la captura digitalización y envío de los archivos, con el objetivo de únicamente ser posteriormente tratados.

2. División de frames y bloques

El primer paso para la extracción de características es dividir las muestras de voz en marcos de aproximadamente 30 milisegundos [30 milisegundos a una frecuencia de muestreo (FS)=8kHz nos da como resultado 240 muestras por marco (frame)]

3. Proceso de paso de ventanas

El siguiente paso en el procesamiento es pasar por una ventana cada uno de los marcos creados en el paso anterior con el objetivo de minimizar la discontinuidad de la señal al principio y fin de cada uno de los marcos.

4. Transformada Rápida de Fourier (FFT)

El siguiente paso es aplicar la transformada rápida de Fourier en el marco al que le hemos pasado la ventana. Convertiendo cada marco del dominio del tiempo al de la frecuencia. La transformada rápida de Fourier es un algoritmo rápido para implementar la transformada de Fourier discreta.

5. Envolvimiento en la frecuencia de MEL

La escala de frecuencias mel es una frecuencia lineal debajo de los 1000 hz y logarítmica por arriba de los 1000 hz. Como punto de referencia, un tono con una frecuencia de 1kHz, 40 decibeles arriba del umbral de la percepción auditiva, es definido como 1000 MELs.

Nuestra aproximación para simular la forma en que los oídos extraen el poder de una señal de habla es aplicando un filtro de bandas al espectro de poder computado en el punto anterior. Este filtro es uniformemente espaciado en la escala de Mel, tiene un pasa bandas triangular que responde a las frecuencias

6. Coeficientes Cepstrales

Después de esto, el poder de la señal triangular es contabilizado, y el valor de este total es tomado. El siguiente paso es convertir el espectro de Mel logarítmico de vuelta al dominio del tiempo. El resultado es llamado Mel Frequency Cepstral Coefficients (coeficientes Cepstrales en las frecuencias de Mel).

Al aplicar el proceso mencionado arriba para cada marco de habla, un conjunto de coeficientes Cepstrales en las frecuencias de Mel serán calculados. Este conjunto de coeficientes es llamado un vector acústico. Estos datos ahora necesitan ser comprimidos. Para comprimir un vector acústico, necesitamos optimizar el proceso de verificación.

Algoritmo general mostrado a bloques



Figura 24. Algoritmo de autenticación de voz

➤ **Algoritmo empleado para la cuantificación Vectorial**

El proceso de cuantificación consiste en representar un vector N-dimensional que puede tomar cualquier valor continuo en cada una de sus N componentes mediante otro vector seleccionado de entre un conjunto finito de vectores N-dimensionales.

La cuantificación vectorial transforma el vector N-dimensional \mathbf{x} , en otro vector N-dimensional \mathbf{y} , mediante una operación de cuantificación:

$$y = q(x)$$

Decimos que \mathbf{x} se cuantifica como \mathbf{y} , y que el operador q es el **operador de cuantificación**.

El vector \mathbf{y} no puede tomar cualquier valor, sino que sólo puede tomar un conjunto finito de valores N-dimensionales \vec{y}_i , $1 \leq i \leq L$

Al conjunto $\mathbf{Y} = \vec{y}_i$, $1 \leq i \leq L$, compuesto por los posibles vectores N-dimensionales se le conoce como **librería de códigos**, ó **codebook**.

Al número de elementos que componen la librería de vectores, que ha de ser un número finito, (en nuestro caso, L), se le llama **tamaño** de la librería.

Al proceso de encontrar los L vectores que forman el conjunto \mathbf{Y} de forma óptima en función del universo de posibles valores de \mathbf{x} se le conoce con el nombre de **entrenamiento**, o **diseño** de la librería.

Básicamente podemos esquematizar el proceso de cuantificación vectorial de la siguiente manera:

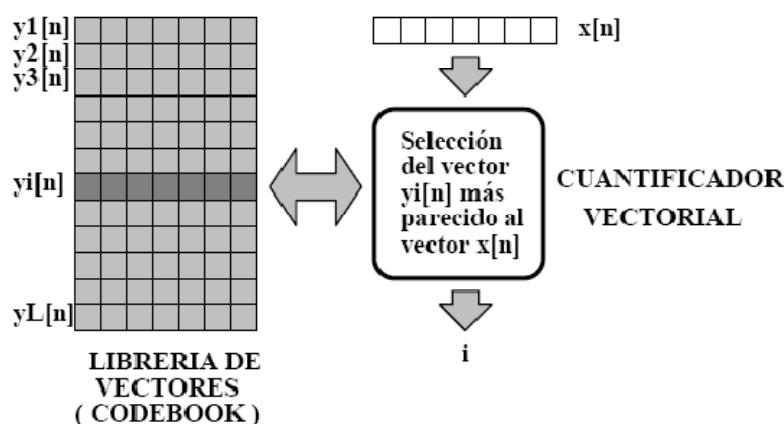


Figura 25. Extracción de vectores mediante cuantificación vectorial.

Existen algoritmos de diseño de la librería de códigos, que si bien no encuentran el cuantificador óptimo para un determinado universo de vectores de entrada, obtienen **cuantificadores sub-óptimos** que corresponden a mínimos locales en la función de distorsión media, pero que en la mayoría de los casos pueden resolver nuestro problema. Para ello se precisa de un conjunto suficientemente amplio de valores de vectores de datos \mathbf{x} , y haber definido una medida de distorsión adecuada.

Básicamente existen dos métodos tradicionales de creación de una librería formada por L vectores:

- **Algoritmo de Linde-Buzo-Gray (LBG):** este sistema no necesita de ninguna inicialización especial, aunque solo permite crear librerías con un número de códigos que sea potencia de 2, o sea, 1, 2, 4, 8, ... [LBG 80]. Los pasos a seguir son los siguientes:

1. Inicialización

Crear un grupo inicial que englobe a todos los vectores de entrenamiento, y calcular su centroide. Sea y_1 este centroide.

i=numero de centroides=1.

2. Duplicación de centroides

Transformar cada centroide y_1 en dos, modificándolo ligeramente. Sean los dos nuevos centroides $y_1 + d$, $y_1 - d$

3. Clasificación

Clasificar el conjunto de vectores de entrenamiento en $2*i$ subgrupos.

4. Recálculo del diccionario

Calcular para cada grupo de vectores su "nuevo" representante.

5. Terminación

Si el número de centroides es menor al deseado, volver al paso 2. En caso contrario, terminar.

4.4.1 TÉCNOLOGIA EMPLEADA

La implementación de los presentes algoritmos se realizo en java puesto que es un lenguaje libre y también nos ofrece librerías de código abierto además de ser multiplataforma, aunado a esto todo el desarrollo de la implementación se pensó en el uso de este lenguaje como pilar, se uso la versión del JSE 1.5 y de J2EE 1.5.

❖ *Librerías empleadas*

➤ *Para subir archivos de audio al servidor:*

Fichero	Producto + Versión	Descripción
commons-fileupload-1.2.1.jar	Commons fileupload 1.2.1	Este Jar es para subir ficheros al Servidor.
commons-io-1.4.jar	commons io 1.4	Lo utiliza el fileUpload

Tabla 21. Librerías empleadas para subir archivos al servidor

➤ *API Servlet:*

Se empleo el API que proporciona java 2EE llamada Serlvet que permite crear Servlets, aplicaciones del lado del servidor que pueden recuperar y procesar datos, en concreto se uso esta api para recuperar los valores mandados por los formularios de captura de voz (acceso por medio de voz y enrolamiento), es decir la recuperación de los archivos de audio y su posterior proceso del lado del servidor

➤ *JSF*

Dado que este framework se basa en el patrón MVC se empleo en esta capa la parte del modelo a traves de dos componentes principales: el manejo de toda la política de negocio de nuestras páginas definidas en la capa de usuario por medio de ManagedBeans, asimismo se implemento el controlador a través del servlet que implementa este framework, de igual forma se usaron los archivos de configuración con los que cuenta el framework (faces-config.xml) para implementar de manera adecuada el manejo de la interfaz de usuario y la política de las mismo.

➤ ***comirva-0.2.N3.jar***

Librería de código abierto, proyecto llamado CoMIRVA: (Collection of Music Information Retrieval and Visualization Applications). Tiene como objetivo crear un framework para la implementación en Java de varios algoritmos concernientes a música, multimedia, recuperación de información, visualización de la información, y minería de datos. Hasta el momento solo hay disponible una versión de CoMIRVA.

La licencia de esta librería esta bajo la denominación GNU y puede ser modificada y redistribuida bajo los términos el GLP.

Esta librería se empleo para el cálculo de los MFCC, dado que brinda una clase llamada así, para el cálculo de estos coeficientes a partir de un pre procesador de audio.

➤ ***Sautrela.jar***

Sautrela es un frameowrk de código abierto altamente modularizada y fácilmente usable centrada en el reconocimiento del habla. El objetivo de Sautrela es unificar un solo Framework que reúna casi en su totalidad tareas relacionadas con: reconocimiento de patrones así como el procesamiento de señales, el modelo de entrenamiento y la decodificación, sus características principales son.

- ✓ Desarrollada en Java.
- ✓ Arquitectura basada en Java Beans.
- ✓ Diferentes técnicas de modelos estocásticos se integran en un modelo generalizado de la arquitectura.
- ✓ Cuenta con un modulo de entrenamiento.

La licencia de esta librería esta bajo la denominación GNU lo que significa que su código es distribuido de manera gratuita y disponible a todo público en general.

Dado que esta librería tiene un paquete donde implementa la cuantificación vectorial, en nuestro caso se empleo para la implementación de la creación de las librerías de códigos puesto que aplica el algoritmo LGB citado arriba para este fin, es decir sirvió para ayudar a generar los codebooks de los usuarios a partir de sus vectores fonéticos.

Además ayudo al cálculo de distancias entre vectores y a calcular el error de distorsión de los codebooks generados, así como también las distancias entre vectores.

4.4.2 CODIFICACIÓN

GESTIÓN DE ARCHIVOS

Clase SubirArchivos:

Esta clase se encarga de implementar las funciones relacionadas con la recuperación de archivos del lado del cliente y subirlos al servidor para después tratarlos:

prospectivearch.java

```
package view;

import java.io.File;
import java.io.IOException;
import java.io.PrintWriter;

import java.util.Iterator;
import java.util.List;

import javax.servlet.*;
import javax.servlet.http.*;
import org.apache.commons.fileupload.FileItem;
import org.apache.commons.fileupload.disk.DiskFileItemFactory;
import org.apache.commons.fileupload.servlet.ServletFileUpload;

public class prospectivearch extends HttpServlet {
    private static final String CONTENT_TYPE = "text/html; charset=windows-1252";

    public void init(ServletConfig config) throws ServletException {
        super.init(config);
    }

    public void doPost(HttpServletRequest request,
                       HttpServletResponse response) throws ServletException,
                                                       IOException {
        response.setContentType(CONTENT_TYPE);
        //procesando informacion del cliente

        try {
            //primero creamos el objeto que es responsable de despachar la peticion
            DiskFileItemFactory fabrica = new DiskFileItemFactory();
            // Fijando restricciones sobre los archivos
            //tamano maximo de los archivos guardado en memoria
            fabrica.setSizeThreshold(1024 * 512);
            // si se excede el tamano anterior, se ira guardando temporalmente, en la sgte
            direccion
            fabrica.setRepository(new File("D:\\temporal\\"));

            // Despues crear un manejador de archivos
            ServletFileUpload upload = new ServletFileUpload(fabrica);
```

```

// Fijando el tamano maximo de la peticion
upload.setSizeMax(1024 * 20000000);
//Obteniendo los elementos de la peticion
List elementos = upload.parseRequest(request);

//procesando los elementos
Iterator iterador = elementos.iterator();
while (iterador.hasNext()) {
    FileItem elemento = (FileItem)iterador.next();

    if (elemento.isFormField()) {
        String nombre = elemento.getFieldName();
        String valor = elemento.getString();
        System.out.println(nombre+" : "+valor);
    } else {
        ServletContext contexto = this.getServletContext();
        String ruta = contexto.getRealPath("/archivos");
        System.out.println("Ruta del doc: "+ruta+"\\"+elemento.getName());
        File fichero = new File(ruta+"\\"+elemento.getName()+".wav");
        elemento.write(fichero);
    }
}
} catch (Exception e) {
    e.printStackTrace();
}

PrintWriter out = response.getWriter();
out.println("<html>");
out.println("<head><title>ProsPeticionesArch</title></head>");
out.println("<body>");
out.println("<p>The servlet has received a POST. This is the reply.</p>");
out.println("</body></html>");
out.close();
}
}

```

Uso del framework JSF en esta capa:

En la parte concerniente al manejo de JSF se empleo el siguiente archivo de configuración para la aplicación WEB:

WEB.xml:

Brinda la configuración requerida para que la aplicación Web java pueda ser desplegada en un contenedor web.

```

<?xml version = '1.0' encoding = 'windows-1252'?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
          http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"

```

```

version="2.5" xmlns="http://java.sun.com/xml/ns/javaee">
<description>Empty web.xml file for Web Application</description>
<context-param>
    <param-name>javax.faces.STATE_SAVING_METHOD</param-name>
    <param-value>client</param-value>
</context-param>
<context-param>
    <description>.</description>
    <param-
name>org.apache.myfaces.trinidad.CHECK_FILE_MODIFICATION</param-
name>
    <param-value>false</param-value>
</context-param>
<filter>
    <filter-name>trinidad</filter-name>
    <filter-class>org.apache.myfaces.trinidad.webapp.TrinidadFilter</filter-
class>
</filter>
<filter-mapping>
    <filter-name>trinidad</filter-name>
    <servlet-name>Faces Servlet</servlet-name>
    <dispatcher>FORWARD</dispatcher>
    <dispatcher>REQUEST</dispatcher>
</filter-mapping>
<servlet>
    <servlet-name>Faces Servlet</servlet-name>
    <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet>
    <servlet-name>resources</servlet-name>
    <servlet-
class>org.apache.myfaces.trinidad.webapp.ResourceServlet</servlet-class>
</servlet>
<servlet>
    <servlet-name>prospectiveSearch</servlet-name>
    <servlet-class>view.prospectiveSearch</servlet-class>
</servlet>
<filter-mapping>
    <filter-name>Faces Servlet</filter-name>
    <url-pattern>/faces/*</url-pattern>
</filter-mapping>
<filter-mapping>
    <filter-name>resources</filter-name>
    <url-pattern>/adf/*</url-pattern>
</filter-mapping>
<filter-mapping>
    <filter-name>resources</filter-name>
    <url-pattern>/afr/*</url-pattern>
</filter-mapping>

```

```

<servlet-mapping>
    <servlet-name>prospectiveSearch</servlet-name>
    <url-pattern>/prospectiveSearch</url-pattern>
</servlet-mapping>
<session-config>
    <session-timeout>35</session-timeout>
</session-config>
<mime-mapping>
    <extension>html</extension>
    <mime-type>text/html</mime-type>
</mime-mapping>
<mime-mapping>
    <extension>txt</extension>
    <mime-type>text/plain</mime-type>
</mime-mapping>
<context-param>
    <!-- Memoria maxima auxiliar por peticion (en bytes) -->
    <param-
name>org.apache.myfaces.trinidad.UPLOAD_MAX_MEMORY</param-
name>
    <!-- Use 500K -->
    <param-value>1000000000</param-value>
</context-param>
<context-param>
    <!-- Espacio maximo en disco por peticion (en bytes) -->
    <param-
name>org.apache.myfaces.trinidad.UPLOAD_MAX_DISK_SPACE</param-
name>
    <!-- Usar 5,000K -->
    <param-value>1000000000</param-value>
</context-param>
<context-param>
    <!-- directory to store temporary files -->
    <param-
name>org.apache.myfaces.trinidad.UPLOAD_TEMP_DIR</param-name>
    <!-- Use a TrinidadUploads subdirectory of /tmp -->
    <param-value>/tmp/TrinidadUploads</param-value>
</context-param>
</web-app>

```

Archivo de configuración de JSF Faces-config.xml

Configura los beans usados, la relación entre páginas JSP usadas para el desarrollo así mismo también configura el uso de librerías externas a la aplicación.

```

<?xml version="1.0" encoding="windows-1252"?>
<faces-config version="1.2" xmlns="http://java.sun.com/xml/ns/javaee">
<application>

```

```

<default-render-kit-id>org.apache.myfaces.trinidad.core</default-render-kit-
id>
</application>
<managed-bean>
<managed-bean-name>Conexion</managed-bean-name>
<managed-bean-class>com.Conexion</managed-bean-class>
<managed-bean-scope>session</managed-bean-scope>
</managed-bean>
<navigation-rule>
<from-view-id>/Index.jsp</from-view-id>
<navigation-case>
<from-outcome>MenuUsuario</from-outcome>
<to-view-id>/MenuAdministrador.jsp</to-view-id>
</navigation-case>
<navigation-case>
<from-outcome>MenuAdministrador</from-outcome>
<to-view-id>/MenuUsuario.jsp</to-view-id>
</navigation-case>
</navigation-rule>
<navigation-rule>
<from-view-id>/MenuAdministrador.jsp</from-view-id>
<navigation-case>
<from-outcome>AdminServicio</from-outcome>
<to-view-id>/MenuAdministracionSevicio.jsp</to-view-id>
</navigation-case>
<navigation-case>
<from-outcome>AdminUsuario</from-outcome>
<to-view-id>/MenuAdministracionUsuario.jsp</to-view-id>
</navigation-case>
</navigation-rule>
<navigation-rule>
<from-view-id>/MenuAdministracionUsuario.jsp</from-view-id>
<navigation-case>
<from-outcome>RegUsuario</from-outcome>
<to-view-id>/RegistroDeUsuario.jsp</to-view-id>
</navigation-case>
<navigation-case>
<from-outcome>ModUsuario</from-outcome>
<to-view-id>/ModificacionUsuario.jsp</to-view-id>
</navigation-case>
<navigation-case>
<from-outcome>ElimUsuario</from-outcome>
<to-view-id>/EliminacionDeUsuario.jsp</to-view-id>
</navigation-case>
<navigation-case>
<from-outcome>AsignServicio</from-outcome>
<to-view-id>/AsignarServicioAUsuario.jsp</to-view-id>
</navigation-case>
<navigation-case>
<from-outcome>ConsultaUsuario</from-outcome>

```

```

<to-view-id>/ConsultasDeUsuarios.jsp</to-view-id>
</navigation-case>
</navigation-rule>
<navigation-rule>
<from-view-id>/MenuAdministracionServicio.jsp</from-view-id>
<navigation-case>
<from-outcome>RegServicio</from-outcome>
<to-view-id>/RegistroDeServicios.jsp</to-view-id>
</navigation-case>
<navigation-case>
<from-outcome>ModServicio</from-outcome>
<to-view-id>/ModificacionDeServicios.jsp</to-view-id>
</navigation-case>
<navigation-case>
<from-outcome>ElimServicio</from-outcome>
<to-view-id>/EliminacionDeServicio.jsp</to-view-id>
</navigation-case>
</navigation-rule>
<navigation-rule>
<from-view-id>/MenuUsuario.jsp</from-view-id>
<navigation-case>
<from-outcome>Menusservicios</from-outcome>
<to-view-id>/MenuDeUsuario.jsp</to-view-id>
</navigation-case>
</navigation-rule>
<managed-bean>
<managed-bean-name>Usuario</managed-bean-name>
<managed-bean-class>com.Usuario</managed-bean-class>
<managed-bean-scope>session</managed-bean-scope>
</managed-bean>
<managed-bean>
<managed-bean-name>Servicio</managed-bean-name>
<managed-bean-class>com.Servicio</managed-bean-class>
<managed-bean-scope>session</managed-bean-scope>
</managed-bean>
<managed-bean>
<managed-bean-name>UsuarioServicio</managed-bean-name>
<managed-bean-class>com.UsuarioServicio</managed-bean-class>
<managed-bean-scope>session</managed-bean-scope>
</managed-bean>
<managed-bean>
<managed-bean-name>ProcesarVoz</managed-bean-name>
<managed-bean-class>com.ProcesarVoz</managed-bean-class>
<managed-bean-scope>session</managed-bean-scope>
</managed-bean>
</faces-config>

```

Clases (ManagedBeans Empleados)

Conexión.java : Implementa toda la política de negocio referente al manejo de conexión a base de datos.

```
package com;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class Conexion {
    //Variables para las consultas y manipulaciona de datos
    Connection jdbcConnection = null;
    Statement sentencia;
    ResultSet resultado;

    public void conectar() throws Exception {
        Class.forName("com.mysql.jdbc.Driver").newInstance();
        jdbcConnection =
            DriverManager.getConnection("jdbc:mysql://localhost/tt2008023?user=root&pa
ssword=genosaurer");

    }

    public ResultSet ejecutarConsulta(String consulta) {
        try {
            conectar();
            sentencia =
                jdbcConnection.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
                    ResultSet.CONCUR_UPDATABLE);
            resultado = sentencia.executeQuery(consulta);
            System.out.println("\n Ejecutando la consulta: " + consulta);
        } catch (Exception e) {
            // TODO
            e.printStackTrace();
        }
        return resultado;
    }

    public int ejecutarModificacion(String sqlstring) {
        int resultados = 0;
        System.out.println("\n Ejecutando la consulta: " + sqlstring);
        try {
            conectar();
            sentencia =
                jdbcConnection.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
```

```
        ResultSet.CONCUR_UPDATABLE);
resultados = sentencia.executeUpdate(sqlstring);

} catch (Exception e) {
    // TODO
    e.printStackTrace();
}

return resultados;
}

public void closeAll() {
    if (sentencia != null) {
        try {
            sentencia.close();
        } catch (Exception ex) {
            ex.printStackTrace();
        }
        sentencia = null;
    }
    if (jdbcConnection != null) {
        try {
            jdbcConnection.close();
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }
}
```

Servicio.java: Implementa toda la política de negocio de la tabla servicio residente en la base de datos.

```
package com;

import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;

import java.sql.ResultSet;

import java.text.SimpleDateFormat;

import java.util.Map;

import java.util.StringTokenizer;

import javax.faces.context.ExternalContext;
import javax.faces.context.FacesContext;

import javax.servlet.ServletContext;
```

```
import javax.servlet.jsp.jstl.sql.Result;
import javax.servlet.jsp.jstl.sql.ResultSupport;
import org.apache.myfaces.trinidad.modelUploadedFile;

public class Servicio {
    private int idServicio;
    private String path;
    private String nombre;
    private Result Servicios;
    private UploadedFile archivo;
    private boolean edicion;
    private Result todos;
    private String crit;
    private int opc;
    private String rutaVieja;

    public Servicio() {
    }

    public void setIdServicio(int idServicio) {
        this.idServicio = idServicio;
    }

    public int getIdServicio() {
        return idServicio;
    }

    public void setPath(String path) {
        this.path = path;
    }

    public String getPath() {
        return path;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getNombre() {
        return nombre;
    }

    public void setServicios(Result Servicios) {
        this.Servicios = Servicios;
    }

    public Result getServicios() {
```

```

        return Servicios;
    }

    public void setArchivo(UploadedFile archivo) {
        this.archivo = archivo;
    }

    public UploadedFile getArchivo() {
        return archivo;
    }

    public void setEdicion(boolean edicion) {
        this.edicion = edicion;
    }

    public boolean isEdicion() {
        return edicion;
    }

    public void setCrit(String crit) {
        this.crit = crit;
    }

    public String getCrit() {
        return crit;
    }

    public void setOpc(int opc) {
        this.opc = opc;
    }

    public int getOpc() {
        return opc;
    }

    public void setTodos(Result todos) {
        this.todos = todos;
    }

    public Result getTodos() {
        return todos;
    }

    public String Alta() throws Exception {
        //Procesando el Archivo
        FacesContext context = FacesContext.getCurrentInstance();
        ExternalContext external = context.getExternalContext();
        ServletContext servletContext = (ServletContext)external.getContext();
        System.out.println("Ruta:" +
                           servletContext.getRealPath("/index.html"));
    }

```

```

String rutadocto = servletContext.getRealPath("/index.html");
rutadocto = rutadocto.replaceAll("index.html", "");
rutadocto = rutadocto.replace("\\", "/");
String rutabase = rutadocto + "FileUploads/";

InputStream entradabytes = archivo.getInputStream();
String nombreach = archivo.getFilename();
System.out.println("El archivo subido es:" + nombreach);
byte[] bufer = new byte[1024];
int nbytes = 1024;
int contadorActual = 1;
File archivoSalida;

path = rutabase + nombreach;

archivoSalida = new File(rutabase + nombreach);
while (archivoSalida.exists()) {
    path = rutabase + contadorActual + nombreach;
    archivoSalida = new File(path);
    contadorActual++;
}

FileOutputStream fsalida = new FileOutputStream(archivoSalida);
do {
    nbytes = entradabytes.read(bufer, 0, 1024);
    System.out.println("Recibiendo: " + nbytes + " bytes");
    if (nbytes != -1) {
        fsalida.write(bufer, 0, nbytes);
        fsalida.flush();
    }
} while (nbytes != -1);
entradabytes.close();
fsalida.close();

//Procesando los datos de la base de Datos
Conexion pconexion = new Conexion();
int resultado;
String sentencia =
    "INSERT INTO SERVICIO (path,nombre)values ( '" +
archivoSalida.getName() +
    "','" + nombre + "')";
resultado = pconexion.ejecutarModificacion(sentencia);

if (resultado > 0) {
    return "RegresarAcuerdo";
}
pconexion.closeAll();
return "fallido";
}

```

```

public Object eliminar() {
    FacesContext context = FacesContext.getCurrentInstance();
    Map map = context.getExternalContext().getRequestParameterMap();
    if (map != null) {
        idServicio = Integer.parseInt(map.get("idSer").toString());
    }
    Conexion pconexion = new Conexion();
    int resultado;
    String sentencia =
        "DELETE FROM SERVICIO WHERE idservicio=" + idServicio;
    resultado = pconexion.ejecutarModificacion(sentencia);
    if (resultado > 0) {
        pconexion.closeAll();
        return "exito";
    }
    pconexion.closeAll();
    return "fallido";
}

public Object consultar() {
    Conexion pconexion = new Conexion();
    ResultSet resultado;
    String sentencia = "";
    if (!crit.equals("")) {
        switch (opc) {
            case 1: //identificador
            {
                sentencia =
                    "SELECT DISTINCT idServicio,path,nombre FROM servicio
WHERE idServicio = " +
                    crit;
            }
            break;
            case 2: //nombre del servicios
            {
                sentencia =
                    "SELECT DISTINCT idServicio,path,nombre FROM servicio
WHERE nombre = " +
                    crit + "'";
            }
            break;
            case 3: //nombre del archivo
            {
                sentencia =
                    "SELECT DISTINCT idServicio,path,nombre FROM servicio
WHERE path = " +
                    crit + "'";
            }
            break;
        }
    }
}

```

```

    } else {
        sentencia = "SELECT idServicio,path,nombre FROM servicio";
    }

    System.out.println(sentencia);
    resultado = pconexion.ejecutarConsulta(sentencia);
    todos = ResultSupport.toResult(resultado);
    return null;
}

public Object prepararServicio() {
    // Add event code here...

    FacesContext context = FacesContext.getCurrentInstance();
    ExternalContext external = context.getExternalContext();
    ServletContext servletContext = (ServletContext)external.getServletContext();
    System.out.println("Ruta:" +
        servletContext.getRealPath("/index.html"));
    String rutadocto = servletContext.getRealPath("/index.html");
    rutadocto = rutadocto.replaceAll("index.html", "");
    rutadocto = rutadocto.replace("\\", "/");
    String rutabase = rutadocto + "FileUploads/";

    //recuperando parametros
    Map map = context.getExternalContext().getRequestParameterMap();

    idServicio = Integer.parseInt(map.get("idServ").toString());
    nombre = map.get("nombreServ").toString();
    rutaVieja = rutabase + map.get("pathServ").toString();
    return null;
}

public Object cambio() throws Exception {
    //Procesando el Archivo
    FacesContext context = FacesContext.getCurrentInstance();
    ExternalContext external = context.getExternalContext();
    ServletContext servletContext = (ServletContext)external.getServletContext();
    System.out.println("Ruta:" +
        servletContext.getRealPath("/index.html"));
    String rutadocto = servletContext.getRealPath("/index.html");
    rutadocto = rutadocto.replaceAll("index.html", "");
    rutadocto = rutadocto.replace("\\", "/");
    String rutabase = rutadocto + "FileUploads/";

    InputStream entradabytes = archivo.getInputStream();
    String nombreach = archivo.getFilename();
    System.out.println("El archivo subido es:" + nombreach);
    byte[] bufer = new byte[1024];
    int nbytes = 1024;
    int contadorActual = 1;
}

```

```

File archivoSalida;
path = rutabase + nombrearch;

archivoSalida = new File(rutabase + nombrearch);
while (archivoSalida.exists()) {
    path = rutabase + contadorActual + nombrearch;
    archivoSalida = new File(path);
    contadorActual++;
}

FileOutputStream fsalida = new FileOutputStream(archivoSalida);
do {
    nbytes = entradabytes.read(bufer, 0, 1024);
    System.out.println("Recibiendo: " + nbytes + " bytes");
    if (nbytes != -1) {
        fsalida.write(bufer, 0, nbytes);
        fsalida.flush();
    }
} while (nbytes != -1);
entradaBytes.close();
fsalida.close();

//Procesando los datos de la base de Datos
Conexion pconexion = new Conexion();
int resultado;
String sentencia =
    "UPDATE SERVICIO SET nombre='" + nombre + "',path='" +
    archivoSalida.getName() + "' WHERE idServicio=" + idServicio;
resultado = pconexion.ejecutarModificacion(sentencia);

if (resultado == 1) {
    pconexion.closeAll();
    File borrar = new File(rutaVieja);
    if (borrar.delete())
        System.out.println("El fichero ha sido borrado satisfactoriamente");
    else
        System.out.println("El fichero no puede ser borrado");
    return "exito";
}
pconexion.closeAll();
return "fracaso";
}
}

```

Usuario.java: Implementa toda la política de negocio de la tabla usuario residente en la base de datos

```
package com;
```

```
import java.io.File;

import java.sql.ResultSet;
import java.sql.SQLException;

import java.util.Map;

import javax.faces.context.ExternalContext;
import javax.faces.context.FacesContext;

import javax.servlet.ServletContext;
import javax.servlet.jsp.jstl.sql.Result;
import javax.servlet.jsp.jstl.sql.ResultSupport;

public class Usuario {
    private int idUsuario;
    private String nombre;
    private String appPaterno;
    private String appMaterno;
    private int edad;
    private String domicilio;
    private int folio;
    private int anioRegistro;
    private int claveElector;
    private int estado;
    private int distrito;
    private int municipio;
    private int localidad;
    private int seccion;
    private String perfil;
    private Result todos;
    private boolean administrador;
    private String sexo;
    private String crit;
    private int opc;

    public Usuario() {
    }

    public void setIdUsuario(int idUsuario) {
        this.idUsuario = idUsuario;
    }

    public int getIdUsuario() {
        return idUsuario;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
}
```

```
public String getNombre() {
    return nombre;
}

public void setAppPaterno(String appPaterno) {
    this.appPaterno = appPaterno;
}

public String getAppPaterno() {
    return appPaterno;
}

public void setAppMaterno(String appMaterno) {
    this.appMaterno = appMaterno;
}

public String getAppMaterno() {
    return appMaterno;
}

public void setEdad(int edad) {
    this.edad = edad;
}

public int getEdad() {
    return edad;
}

public void setDomicilio(String domicilio) {
    this.domicilio = domicilio;
}

public String getDomicilio() {
    return domicilio;
}

public void setFolio(int folio) {
    this.folio = folio;
}

public int getFolio() {
    return folio;
}

public void setAnioRegistro(int anioRegistro) {
    this.anioRegistro = anioRegistro;
}

public int getAnioRegistro() {
```

```
        return anioRegistro;
    }

    public void setClaveElector(int claveElector) {
        this.claveElector = claveElector;
    }

    public int getClaveElector() {
        return claveElector;
    }

    public void setEstado(int estado) {
        this.estado = estado;
    }

    public int getEstado() {
        return estado;
    }

    public void setDistrito(int distrito) {
        this.distrito = distrito;
    }

    public int getDistrito() {
        return distrito;
    }

    public void setMunicipio(int municipio) {
        this.municipio = municipio;
    }

    public int getMunicipio() {
        return municipio;
    }

    public void setLocalidad(int localidad) {
        this.localidad = localidad;
    }

    public int getLocalidad() {
        return localidad;
    }

    public void setSeccion(int seccion) {
        this.seccion = seccion;
    }

    public int getSeccion() {
        return seccion;
    }
```

```
public void setPerfil(String perfil) {
    this.perfil = perfil;
}

public String getPerfil() {
    return perfil;
}

public void setTodos(Result todos) {
    this.todos = todos;
}

public Result getTodos() {
    return todos;
}

public void setAdministrador(boolean administrador) {
    this.administrador = administrador;
}

public boolean isAdministrador() {
    return administrador;
}

public void setSexo(String sexo) {
    this.sexo = sexo;
}

public String getSexo() {
    return sexo;
}

public void setCrit(String crit) {
    this.crit = crit;
}

public String getCrit() {
    return crit;
}

public void setOpc(int opc) {
    this.opc = opc;
}

public int getOpc() {
    return opc;
}

public Object Alta()
```

```

Conexion pconexion = new Conexion();
int resultado;
String sentencia =
    "INSERT INTO USUARIO (nombre, edad, sexo, domicilio,
folio, anio_Registro, clave_Elector," +
"estado, distrito, municipio, localidad, seccion, appaterno, appmaterno, perfil, fase)va
lues (" +
    nombre + "", "" + edad + "", "" + sexo + "", "" + domicilio +
    "", "" + folio + "", "" + anioRegistro + "", "" + claveElector +
    "", "" + estado + "", "" + distrito + "", "" + municipio + "", "" +
    localidad + "", "" + seccion + "", "" + appPaterno + "", "" +
    appMaterno + "", "" + perfil + "", 0);
resultado = pconexion.ejecutarModificacion(sentencia);
if (resultado > 0) {
    pconexion.closeAll();
    //creando carpeta de enrolamiento
    FacesContext context = FacesContext.getCurrentInstance();
    ExternalContext external = context.getExternalContext();
    ServletContext servletContext = (ServletContext)external.getServletContext();
    String rutadocto = servletContext.getRealPath("/index.html");
    rutadocto = rutadocto.replaceAll("index.html", "");
    rutadocto = rutadocto.replace("\\", "/");
    String rutabase = rutadocto + "FileUploads/" + claveElector;
    File dir = new File(rutabase);
    //el directorio de enrolamiento se crea en FileUploads/claveElector
    if (dir.mkdirs()) {
        System.out.println("Directorio de enrolamiento crado");
    }
}

return "rexitoso";
}
pconexion.closeAll();
return "fallido";
}

public Object consultar() {
    Conexion pconexion = new Conexion();
    ResultSet resultado;
    String sentencia = "";
    if (!crit.equals("")) {
        switch (opc) {
        case 1: //clave elector
        {
            sentencia =
                "SELECT DISTINCT idUsuario, clave_Elector, nombre,
appaterno, appmaterno FROM usuario WHERE clave_Elector = '" +
                crit + "'";
        }
        break;
    }
}

```

```

        case 2: //nombre
        {
            sentencia =
                "SELECT DISTINCT idUsuario,clave_Elector, nombre,
appaterno,appmaterno FROM usuario WHERE nombre = '" +
                crit + "'";
        }
        break;
    case 3: //appPaterno
    {
        sentencia =
            "SELECT DISTINCT idUsuario,clave_Elector, nombre,
appaterno,appmaterno FROM usuario WHERE appaterno = '" +
            crit + "'";
    }
    break;
    case 4: //appMaterno
    {
        sentencia =
            "SELECT DISTINCT idUsuario,clave_Elector, nombre,
appaterno,appmaterno FROM usuario WHERE appmaterno= '" +
            crit + "'";
    }
    break;
}
} else {
    sentencia =
        "SELECT idUsuario,clave_Elector, nombre, appaterno,appmaterno
FROM usuario";
}
System.out.println(sentencia);
resultado = pconexion.ejecutarConsulta(sentencia);
todos = ResultSupport.toResult(resultado);
return null;
}

public Object eliminar() {
    FacesContext context = FacesContext.getCurrentInstance();
    Map map = context.getExternalContext().getRequestParameterMap();
    if (map != null) {
        idUsuario = Integer.parseInt(map.get("idUsr").toString());
    }
    Conexion pconexion = new Conexion();
    int resultado;
    String sentencia =
        "DELETE FROM SERVICIO WHERE idservicio=" + idUsuario;
    resultado = pconexion.ejecutarModificacion(sentencia);
    if (resultado > 0) {
        pconexion.closeAll();
        return "exito";
    }
}

```

```

        }
        pconexion.closeAll();
        return "fallido";
    }

public Object prepararUsuario() throws SQLException {
    FacesContext context = FacesContext.getCurrentInstance();
    Map map = context.getExternalContext().getRequestParameterMap();
    if (map != null) {
        idUsuario = Integer.parseInt(map.get("idUsr").toString());
    }
    Conexion pconexion = new Conexion();
    ResultSet resultado;
    String sentencia = "SELECT * FROM usuario WHERE idUsuario = "
    "+idUsuario;
    resultado = pconexion.ejecutarConsulta(sentencia);
    if(resultado.next()){
        idUsuario=resultado.getInt("idUsuario");
        nombre=resultado.getString("nombre");
        edad=resultado.getInt("edad");
        sexo=resultado.getString("sexo");
        domicilio=resultado.getString("domicilio");
        folio= resultado.getInt("folio");
        anioRegistro= resultado.getInt("anio_Registro");
        claveElector= resultado.getInt("clave_Elector");
        estado= resultado.getInt("estado");
        distrito= resultado.getInt("distrito");
        municipio=resultado.getInt("municipio");
        localidad= resultado.getInt("localidad");
        seccion=resultado.getInt("seccion");
        appPaterno= resultado.getString("appaterno");
        appMaterno= resultado.getString("appmaterno");
        perfil= resultado.getString("perfil");
    }else{
        System.out.println("no carga valores de base de datos");
    }
    return null;
}

public Object actualizar() {
    //Procesando los datos de la base de Datos
    Conexion pconexion = new Conexion();
    int resultado;
    String sentencia =
        "UPDATE USUARIO SET nombre='" + nombre + "',edad=" + edad +
    ",sexo ='" +
        sexo+"', domicilio ='" +domicilio+"', folio = " +folio+ " , anio_Registro =
    "+ anioRegistro+ " , clave_Elector =" +claveElector+
        " , estado=" +estado+ " , distrito=" +distrito+ " , municipio =" +municipio+",
    localidad =" +localidad+ " , seccion =" +seccion+

```

```

        ", appaterno='"+appPaterno+"', appmaterno='"+appMaterno+"', perfil=
"+perfil+" WHERE idUsuario= " + idUsuario;
        resultado = pconexion.ejecutarModificacion(sentencia);

        if (resultado == 1) {
            pconexion.closeAll();
            return "exito";
        }
        pconexion.closeAll();
        return "fracaso";
    }
}

```

UsuarioServicio.java: Implementa la política de negocio de la tabla UsuarioServicio de la base de datos.

```

package com;

import java.sql.ResultSet;

import java.text.SimpleDateFormat;

import java.util.Map;

import javax.faces.context.FacesContext;

import javax.servlet.jsp.jstl.sql.Result;
import javax.servlet.jsp.jstl.sql.ResultSupport;

public class UsuarioServicio {
    private int idUsuario;
    private int idServicio;
    private String strUsr;
    private String strServ;
    private String crit1;
    private String crit2;
    private int opc1;
    private int opc2;
    private Result usuarios;
    private Result documentos;
    private int opcion;

    public UsuarioServicio() {
    }

    public void setIdUsuario(int idUsuario) {
        this.idUsuario = idUsuario;
    }
}

```

```
public int getIdUsuario() {
    return idUsuario;
}

public void setIdServicio(int idServicio) {
    this.idServicio = idServicio;
}

public int getIdServicio() {
    return idServicio;
}

public void setUsuarios(Result usuarios) {
    this.usuarios = usuarios;
}

public Result getUsuarios() {
    return usuarios;
}

public void setDocumentos(Result documentos) {
    this.documentos = documentos;
}

public Result getDocumentos() {
    return documentos;
}

public void setOpcion(int opcion) {
    this.opcion = opcion;
}

public int getOpcion() {
    return opcion;
}

public void setStrUsr(String strUsr) {
    this.strUsr = strUsr;
}

public String getStrUsr() {
    return strUsr;
}

public void setStrServ(String strServ) {
    this.strServ = strServ;
}

public String getStrServ() {
    return strServ;
```

```

}

public void setCrit1(String crit1) {
    this.crit1 = crit1;
}

public String getCrit1() {
    return crit1;
}

public void setCrit2(String crit2) {
    this.crit2 = crit2;
}

public String getCrit2() {
    return crit2;
}

public void setOpc1(int opc1) {
    this.opc1 = opc1;
}

public int getOpc1() {
    return opc1;
}

public void setOpc2(int opc2) {
    this.opc2 = opc2;
}

public int getOpc2() {
    return opc2;
}

public Object asignarUsuario() {
    FacesContext context = FacesContext.getCurrentInstance();
    Map map = context.getExternalContext().getRequestParameterMap();
    if (map != null) {
        idUsuario = Integer.parseInt(map.get("idUsr").toString());
        strUsr = map.get("nombreUsr").toString();
    }
    return null;
}

public Object filtrarUsuarios() {
    Conexion pconexion = new Conexion();
    ResultSet resultado;
    String sentencia = "";
    if (!crit1.equals("")) {
        switch (opc1) {

```

```

case 1: //clave elector
{
    sentencia =
        "SELECT DISTINCT idUsuario,clave_Elector, nombre,
appaterno,appmaterno FROM usuario WHERE clave_Elector = '" +
        crit1 + "'";
}
break;
case 2: //nombre
{
    sentencia =
        "SELECT DISTINCT idUsuario,clave_Elector, nombre,
appaterno,appmaterno FROM usuario WHERE nombre = '" +
        crit1 + "'";
}
break;
case 3: //appPaterno
{
    sentencia =
        "SELECT DISTINCT idUsuario,clave_Elector, nombre,
appaterno,appmaterno FROM usuario WHERE appaterno = '" +
        crit1 + "'";
}
break;
case 4: //appMaterno
{
    sentencia =
        "SELECT DISTINCT idUsuario,clave_Elector, nombre,
appaterno,appmaterno FROM usuario WHERE appmaterno= '" +
        crit1 + "'";
}
break;
}
}else{
    sentencia =
        "SELECT idUsuario,clave_Elector, nombre, appaterno,appmaterno
FROM usuario";
}
System.out.println(sentencia);
resultado = pconexion.ejecutarConsulta(sentencia);
usuarios = ResultSupport.toResult(resultado);
return null;
}

public Object filtrarServicios() {
    Conexion pconexion = new Conexion();
    ResultSet resultado;
    String sentencia = "";
    if (!crit2.equals("")) {
        switch (opc1) {

```

```

        case 1: //identificador
        {
            sentencia =
                "SELECT DISTINCT idServicio,path,nombre FROM servicio
WHERE idServicio = " +
                crit2;
            }
            break;
        case 2: //nombre del servicios
        {
            sentencia =
                "SELECT DISTINCT idServicio,path,nombre FROM servicio
WHERE nombre = " +
                crit2 + "';";
            }
            break;
        case 3: //nombre del archivo
        {
            sentencia =
                "SELECT DISTINCT idServicio,path,nombre FROM servicio
WHERE path = " +
                crit2 + "';";
            }
            break;
        }
    }else{
        sentencia ="SELECT idServicio,path,nombre FROM servicio";
    }

    System.out.println(sentencia);
    resultado = pconexion.ejecutarConsulta(sentencia);
    documentos = ResultSupport.toResult(resultado);
    return null;
}

public Object asignarServicio() {
    FacesContext context = FacesContext.getCurrentInstance();
    Map map = context.getExternalContext().getRequestParameterMap();
    if (map != null) {
        idServicio = Integer.parseInt(map.get("idSer").toString());
        strServ = map.get("nombreSer").toString();
    }
    return null;
}

public String Alta() {
    Conexion pconexion = new Conexion();
    int resultado;
    String sentencia =

```

```

    "INSERT INTO usuarioservicio (idUsuario,idServicio)values (" +
    idUsuario + "," + idServicio + ")";
    resultado = pconexion.ejecutarModificacion(sentencia);

    if (resultado > 0) {
        return "exito";
    }
    pconexion.closeAll();
    return "fallido";
}
}

```

TRATAMIENTO DE LA VOZ

Clase TratamientoDeMFCC

Esta clase es encargada de implementar todas las funcionalidades que se refieren a la extracción de los coeficientes en la escala Mel, tiene los siguientes atributos:

- Atributos con que cuenta la clase:
 - CoeficientesMel: es un objeto donde se almacenan y procesan los coeficientes.
 - fMuestreo: es la frecuencia de muestreo a la que se trabaja el archivo de audio muestreado.
 - rutaArchivoVoz: es la ruta del documento de voz del que se extraerán los patrones.
 - vectorAcustico: es el vector donde se almacenaran los coeficientes como vectores fonéticos.
- Métodos con que cuenta la clase:
 - generarVectoresAcusticos: este método realiza propiamente todo el proceso y almacena el resultado en el atributo vector acústico.
 - guardarVectoresAcusticos: este método recibe un objeto tipo File donde se guardaran los vectores acústicos.
 - recuperarVectoresAcusticos: este método recibe un objeto tipo File donde de donde se recuperaran los vectores acústicos y se cargaran al atributo vectorAcustico.
 - Adicional a cada uno de estos métodos cada uno de los atributos cuenta con sus métodos getters y setters.

○ Código:

```
import comirva.audio.util.AudioPreProcessor;
```

```
import comirva.audio.util.MFCC;
```

```
import java.io.BufferedReader;
```

```
import java.io.File;
```

```
import java.io.FileReader;
```

```
import java.io.FileWriter;
```

```
import java.io.IOException;
```

```
import java.util.Iterator;
```

```
import java.util.StringTokenizer;
```

```
import java.util.Vector;
```

```

import javax.sound.sampled.AudioInputStream;
import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.UnsupportedAudioFileException;

public class TratamientoDeMFCC {
    private MFCC coefientesMel;
    private String rutaArchivoVoz;
    private Vector vectorAcustico;
    private float fMuestreo;

    public TratamientoDeMFCC(String ruta) {
        rutaArchivoVoz = ruta;
    }

    public void generarVectoresAcusticos() {
        File archivoAudio;
        AudioInputStream flujoAudio;
        AudioPreProcessor procesadorAudio;
        try {

            archivoAudio = new File(rutaArchivoVoz);
            flujoAudio = AudioSystem.getAudioInputStream(archivoAudio);
            procesadorAudio = new AudioPreProcessor(flujoAudio);
            fMuestreo = procesadorAudio.getSampleRate();
            coefientesMel = new MFCC(fMuestreo);
            vectorAcustico = coefientesMel.process(procesadorAudio);
            flujoAudio.close();
            flujoAudio = null;
            archivoAudio = null;
        } catch (UnsupportedAudioFileException e) {
            System.out.println("Arvhivo no soportado");
        } catch (IOException e) {
            System.out.println("Error de entrada-salida");
        }
    }

    public boolean guardarVectoresAcusticos(File archivoGuardar) {
        Iterator recorrer = vectorAcustico.iterator();
        String straux;
        FileWriter grabador;
        try {
            grabador = new FileWriter(archivoGuardar, true);

            while (recorrer.hasNext()) {
                double[] dblaux = (double[])recorrer.next();
                straux = "";
                for (double auxdd : dblaux) {
                    straux += Double.toString(auxdd) + ",";
                }
            }
        }
    }
}

```

```

        grabador.write(straux + "\n");
        grabador.flush();
    }
    grabador.close();
    System.out.println("Archivo creado");
    return true;
} catch (IOException e) {
    e.printStackTrace();
    return false;
}
}

public void recuperarVectoresAcusticos(File archivoLeer) throws
Exception {
    vectorAcustico = new Vector();
    double dblaux[];
    String strAux;
    FileReader lector = new FileReader(archivoLeer);
    BufferedReader bufer = new BufferedReader(lector);
    while ((strAux = bufer.readLine()) != null) {
        if (!strAux.equals("")) {
            StringTokenizer stkSeparador =
                new StringTokenizer(strAux, ",");
            int ntok = stkSeparador.countTokens();
            int conter = 0;
            dblaux = new double[ntok];
            while (stkSeparador.hasMoreTokens()) {
                dblaux[conter] =
                    Double.parseDouble(stkSeparador.nextToken());
                conter++;
            }
            vectorAcustico.add(dblaux);
        }
    }
}

public void setCoeficientesMel(MFCC coefientesMel) {
    this.coefientesMel = coefientesMel;
}

public MFCC getCoeficientesMel() {
    return coefientesMel;
}

public void setRutaArchivoVoz(String rutaArchivoVoz) {
    this.rutaArchivoVoz = rutaArchivoVoz;
}

public String getRutaArchivoVoz() {
    return rutaArchivoVoz;
}

```

```

    }

    public void setVectorAcustico(Vector vectorAcustico) {
        this.vectorAcustico = vectorAcustico;
    }

    public Vector getVectorAcustico() {
        return vectorAcustico;
    }

    public void setFMuestreo(float fMuestreo) {
        this.fMuestreo = fMuestreo;
    }

    public float getFMuestreo() {
        return fMuestreo;
    }
}

```

Clase Librería

Esta clase implementa todas las tareas relacionadas con la generación de la librería de códigos a partir de los vectores de entrenamiento así mismo es encargada de proveer el método de verificación para comprobar si posibles vectores pueden pertenecer o pueden ser cuantificados dentro de esta misma, cuenta también además con funciones para calcular la distancia entre vectores, para generar el libro de códigos y además para cuantificar vectores posibles dentro de la misma, a continuación una descripción detallada de la misma:

- Atributos con que cuenta la clase:
 - FDatosLeidos: es un atributo de tipo buffer donde se van almacenando todas las entradas que conforman el vector de entrenamiento de la librería, a partir del cual se generara el libro de códigos de la misma.
 - LibroCódigos atributo de tipo vector donde se almacenara el libro de códigos de la misma.
 - distorcion: atributo donde se almacena el factor de distorsión del libro de códigos que se genera.
 - entrenadorDeCódigos: atributo que permite por medio de sus métodos realizar el entrenamiento de la librería y generar el libro de códigos de la misma, así mismo calcula el factor de istorcion de la misma.
 - rutaGuardarLibro: atrobuto de tipo cadena donde se almacena la ruta donde se guardara el libro de códigos que se genere.
 - rutaVENTrenamiento: ruta donde se almacena y de donde se leerá el vector de entrenamiento de la librería.
 - Salida: atributo de tipo buffer donde se almacenara la salida del proceso de calcular la librería de códigos.
- Métodos con que cuenta la clase:
 - leerVectoresEntrenamiento: método que lee los vectores de entrenamiento del archvivo especificado en la ruta rutaVENTrenamiento, y los deposita en el buffer de entrada FDatosLeidos.
 - generarLibroDeCódigos: método que realiza la acción de generar el libro de códigos a partir del vector de entrenamiento dado.

- recuperarLibroDeCódigos: método que recibe de parámetro un objeto de tipo File de donde se leerán valores del libro de códigos y serán cargados en el atributo LibroDeCódigos.
 - cuantificar: función que recibe un vector, calcula y regresa el índice del libro de códigos del posible vector que lo cuantifica.
 - distanciaEuclídeana: función que calcula la distancia euclíadiana entre dos vectores.
 - validarUsuario: función que recibe como parámetro el vector fonético que clama ser verificado para ver si puede ser cuantificado dentro de la librería de códigos actual, de corroborarse que si pertenece a la librería se regresa verdadero de lo contrario falso.
- Código:
- ```

import edu.gtts.sautrela.engine.data.DoubleData;
import edu.gtts.sautrela.engine.data.CloseData;
import edu.gtts.sautrela.engine.Buffer;
import edu.gtts.sautrela.engine.DataProcessorException;
import edu.gtts.sautrela.engine.data.Data;
import edu.gtts.sautrela.engine.util.StreamReader;
import edu.gtts.sautrela.vq.CodebookTrainer;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;

import java.net.URL;

import java.util.ArrayList;
import java.util.StringTokenizer;
import java.util.Vector;

public class Libreria {
 CodebookTrainer entrenadorDeCódigos;
 Buffer salida;
 Buffer FDatosLeidos;
 String rutaGuardarLibro;
 String rutaVEntrenamiento;
 Vector LibroCódigos;
 double distorcion;

 public Libreria(String rutaVe, String rutaguarL) {
 rutaVEntrenamiento = rutaVe;
 rutaGuardarLibro = rutaguarL;
 entrenadorDeCódigos.setMaxIter(2);
 entrenadorDeCódigos.setCodebookSize(32);
 entrenadorDeCódigos.setELBG(true);
 entrenadorDeCódigos.setIncremental(true);
 entrenadorDeCódigos.setFinalCodebook(new File(rutaGuardarLibro));
 }

 public void leerVectoresEntrenamiento() throws Exception {

```

```

salida = new Buffer();
FDatosLeidos = new Buffer();
FileReader lector = new FileReader(new File(rutaVENTrenamiento));
BufferedReader bufer = new BufferedReader(lector);
Vector vector = new Vector();
DoubleData datoDbl;
double dblaux[];
String strAux;

while ((strAux = bufer.readLine()) != null) {
 if (!strAux.equals("")) {
 StringTokenizer stkSeparador =
 new StringTokenizer(strAux, ",");
 int ntok = stkSeparador.countTokens();
 int conter = 0;
 dblaux = new double[ntok];
 while (stkSeparador.hasMoreTokens()) {
 dblaux[conter] =
 Double.parseDouble(stkSeparador.nextToken());
 conter++;
 }
 datoDbl = new DoubleData(dblaux);
 FDatosLeidos.push(datoDbl);
 vector.add(dblaux);
 }
}
bufer.close();
lector.close();
}

public void generarLibroDeCódigos() throws DataProcessorException {
 CloseData dfinal = new CloseData();
 FDatosLeidos.push(dfinal);
 entrenadorDeCódigos.process(FDatosLeidos, salida);
}

public void recuperarLibroDeCódigos(File archivoLeer) throws Exception {
 ArrayList<Data> codeBook;
 codeBook = new ArrayList();
 LibroCódigos = new Vector();
 double[] arreglodbl;
 URL urlarchivo = archivoLeer.toURI().toURL();
 codeBook = StreamReader.readStream(urlarchivo, false);

 for (Data arrdbl : codeBook) {
 arreglodbl =
 AutentificadorDeVoz.cadenaAarreglo(arrdbl.toString(), 20);
 if (arreglodbl != null) {
 LibroCódigos.add(arreglodbl);
 }
 }
}

```

```

 }
 }

public boolean validarUsuario(Vector<double[]> vectorAevaluar,
 int lvector) {
 double arrDistancias[] = new double[lvector];
 int cont = 0;
 for (double[] arrdbl : vectorAevaluar) {
 int ivcuant = cuantificar(arrdbl);
 arrDistancias[cont] =
 Libreria.distanciaEuclidea((double[])LibroCódigos.get(ivcuant),
 arrdbl);
 cont++;
 }
 for (double dbldist : arrDistancias) {
 if (dbldist > this.distorsion) {
 return false;
 }
 }
 return true;
}

public static double distanciaEuclidea(double[] va, double[] vb) {
 double sumas = 0;
 double dblaux;
 for (int i = 0; i < va.length; i++) {
 dblaux = Math.pow((va[i] - vb[i]), 2);
 sumas += dblaux;
 }
 dblaux = Math.sqrt(sumas);
 return dblaux;
}

public int cuantificar(double[] vectorAcuantifcar) {
 double dist;
 int indice = 0;
 int conter = 0;
 double vDeLibCod[] = (double[])LibroCódigos.get(0);
 dist = Math.abs(Libreria.distanciaEuclidea(vDeLibCod,
 vectorAcuantifcar));
 for (double[] vLibCod : LibroCódigos) {
 double distaux =
 Math.abs(Libreria.distanciaEuclidea(vLibCod, vectorAcuantifcar));
 if (distaux <= dist) {
 dist = distaux;
 indice = conter;
 }
 conter++;
 }
 return indice;
}

```

```

}

public void setEntrenadorDeCódigos(CodebookTrainer
entrenadorDeCódigos) {
 this.entrenadorDeCódigos = entrenadorDeCódigos;
}

public CodebookTrainer getEntrenadorDeCódigos() {
 return entrenadorDeCódigos;
}

public void setSalida(Buffer salida) {
 this.salida = salida;
}

public Buffer getSalida() {
 return salida;
}

public void setFDatosLeídos(Buffer FDatosLeídos) {
 this.FDatosLeídos = FDatosLeídos;
}

public Buffer getFDatosLeídos() {
 return FDatosLeídos;
}

public void setRutaGuardarLibro(String rutaGuardarLibro) {
 this.rutaGuardarLibro = rutaGuardarLibro;
}

public String getRutaGuardarLibro() {
 return rutaGuardarLibro;
}

public void setRutaVEntrenamiento(String rutaVEntrenamiento) {
 this.rutaVEntrenamiento = rutaVEntrenamiento;
}

public String getRutaVEntrenamiento() {
 return rutaVEntrenamiento;
}

public void setLibroCódigos(Vector LibroCódigos) {
 this.LibroCódigos = LibroCódigos;
}

public Vector getLibroCódigos() {
 return LibroCódigos;
}

```

```

public void setDistorción(double distorción) {
 this.distorción = distorción;
}

public double getDistorción() {
 return distorción;
}
}

```

## 4.5 CAPA ONTOLÓGICA

---

En la capa ontológica, se pretende hacer una correcta estructuración de la información de los usuarios de los diversos sistemas de autentificación que implementen la presente arquitectura con el propósito de hacer la organización, búsqueda y compartición de la información a través de Internet de manera inteligente, previendo una futura reestructuración de esta red mundial a lo que se conoce como Web Semántica.

Para que esto ocurra, es necesario que la información de las páginas web se codifique mediante ontologías. Las ontologías representarán el conocimiento de Internet, definiendo formalmente los conceptos de los diferentes dominios y sus relaciones, con capacidad para realizar deducciones con este conocimiento.

La idea es que los datos puedan ser utilizados y “comprendidos” por los ordenadores sin necesidad de supervisión humana, de forma que los agentes web puedan ser diseñados para tratar la información situada en las páginas web de manera semiautomática.

Se trata de convertir la información en conocimiento, referenciando datos dentro de las páginas web a metadatos con un esquema común consensuado sobre algún dominio. Los metadatos no sólo especificarán el esquema de datos que debe aparecer en cada instancia, sino que además podrán tener información adicional de cómo hacer deducciones con ellos, es decir, axiomas que podrán aplicarse en los diferentes dominios que trate el conocimiento almacenado.

---

### 4.5.1 TÉCNOLOGIA EMPLEADA

---

#### ONTOLOGÍA

Para que esto pueda llevarse a cabo, se necesita que el conocimiento de la web esté representado de forma que sea legible por las computadoras, esté consensuado, y sea reutilizable. Las ontologías proporcionan la vía para representar este conocimiento.

El término ontología proviene de la filosofía; pero en IA, tiene diferentes connotaciones. La definición declarativa más consolidada es la propuesta por Gruber [25] y extendida por Studer y colegas [26] que la describe como “una especificación explícita y formal sobre una conceptualización compartida”. La interpretación de esta definición es que las ontologías definen conceptos y relaciones de algún dominio, de forma compartida y

consensuada; y que esta conceptualización debe ser representada de una manera formal, legible y utilizable por las computadoras.

Las ontologías tienen los siguientes componentes que servirán para representar el conocimiento de algún dominio:

- ❖ **Conceptos:** son las ideas básicas que se intentan formalizar. Los conceptos pueden ser clases de objetos, métodos, planes, estrategias, procesos de razonamiento, etc.
- ❖ **Relaciones:** representan la interacción y enlace entre los conceptos del dominio. Suelen formar la taxonomía del dominio. Por ejemplo: *subclase-de*, *parte-de*, *parte-exhaustiva-de*, *conectado-a*, etc.
- ❖ **Funciones:** son un tipo concreto de relación donde se identifica un elemento mediante el cálculo de una función que considera varios elementos de la ontología. Por ejemplo, pueden aparecer funciones como *categorizar-clase*, *asignarfecha*, etc.
- ❖ **Instancias:** se utilizan para representar objetos determinados de un concepto.
- ❖ **Axiomas:** son teoremas que se declaran sobre relaciones que deben cumplir los elementos de la ontología. Por ejemplo: “*Si A y B son de la clase C, entonces A no es subclase de B*”, “*Para todo A que cumpla la condición C1, A es B*”, etc.

Estos últimos componentes, los axiomas, permiten junto con la herencia de conceptos, inferir conocimiento que no esté indicado explícitamente en la taxonomía de conceptos.

## LENGUAJE ONTOLÓGICO.

Actualmente se cuenta con el lenguaje para ontologías OWL (Ontology Web Language) el cual es un lenguaje de marcado para publicar y compartir datos usando ontologías en la WWW. OWL tiene como objetivo facilitar un modelo de marcado construido sobre RDF y codificado en XML.

OWL permite crear ontologías las cuales tienen como características principales las siguientes:

- Capacidad de ser distribuidas a través de varios sistemas
- Escalable a las necesidades de la Web
- Compatible con los estándares Web de accesibilidad e internacionalización
- Abierto y extensible

## ENTORNO DE DESARROLLO

Protégé es una herramienta para el desarrollo de Ontologías y Sistemas basados en el conocimiento creada en la Universidad de Stanford. Protégé está desarrollada en JAVA y puede funcionar perfectamente bajo WINDOWS.

Las aplicaciones desarrolladas con Protégé son empleadas en resolución de problemas y toma de decisiones en dominios particulares. La herramienta Protégé emplea una interfaz de usuario que facilita la creación de una estructura de *frames* con clases, slots e instancias de una forma integrada.

La interfaz grafica de Protégé, permite al desarrollador crear ontologías sin necesidad de escribir una línea de código, dando como resultado un archivo OWL. Sin embargo, también se cuenta con un API Java que permite la creación de Ontologías de manera dinámica, en tiempo de ejecución.

## MARCO DE DESARROLLO

A pesar de las ventajas del entorno de desarrollo Protégé, si se desea dar un almacenamiento persistente a los datos, es decir guardarlos en una base de datos relacional, existe para ello el Framework JENA, desarrollado por HP Labs para manipular metadata desde una aplicación Java.

JENA cuenta con un API de Ontologías con soporte para OWL. Jena permite crear modelos persistentes:

- Son persistidos de forma transparente en una base de datos relacional.

JENA 2 soporta:

- MySQL
- Oracle
- PostgreSQL

---

### 4.5.2 CODIFICACIÓN

---

El código para la generación del esquema ontológico es:

```
//ARCHIVO ConexionBD.java
package BaseDatos;

import com.hp.hpl.jena.db.DBConnection;
import com.hp.hpl.jena.db.IDBConnection;
import com.hp.hpl.jena.db.ModelRDB;
import com.hp.hpl.jena.ontology.OntClass;
import com.hp.hpl.jena.ontology.OntModel;
import com.hp.hpl.jena.ontology.OntModelSpec;
import com.hp.hpl.jena.rdf.model.ModelFactory;
import com.hp.hpl.jena.rdf.model.ModelMaker;
import com.hp.hpl.jena.util.FileManager;
import com.hp.hpl.jena.util.iterator.ExtendedIterator;
import java.io.InputStream;
import java.util.logging.Level;
import java.util.logging.Logger;

public class ConexionBD {
 //Objetos para conexion y manejo de Ontologias
 private OntModel ontologia;
 private IDBConnection con;
 private ModelMaker maker;
 private ModelRDB modeloR;

 //Parametros para conectar a bd
```

```

private String bd_nombre="sistemaautentificacion";
private String bd_usuario="root";
private String bd_pass="password";
private String bd_manejador="MySQL";
private String nombre_modelo="autentificacion";

public ConexionBD(String bd_nombre, String bd_usuario, String bd_pass, String
bd_manejador, String nombre_modelo){
 //Cargar Driver para conexion a base de datos
 try {
 Class.forName("com.mysql.jdbc.Driver");
 } catch (ClassNotFoundException ex) {
 Logger.getLogger(ConexionBD.class.getName()).log(Level.SEVERE, null, ex);
 }

 //Inicilizacion de parametros
 this.bd_nombre=bd_nombre;
 this.bd_usuario=bd_usuario;
 this.bd_pass=bd_pass;
 this.bd_manejador=bd_manejador;
 this.nombre_modelo=nombre_modelo;
}

//Método para cargar el esquema ontologico que estaremos utilizando
//Este esquema ha sido credo con PROTEGE anteriormente
public void cargarOntologia(){
 //Crear el modelo para contener la ontología. La especificación es la siguiente:
 //*Lenguaje OWL-FULL
 //*Almacenaje en Memoria
 //*Inferencia basada en RDF – Schema
 setOntologia(ModelFactory.createOntologyModel());

 //Abrir el archivo con la ontologia
 InputStream in=(FileManager.get()).open("AutentificacionUsuarios.owl");
 if (in == null) {
 throw new IllegalArgumentException("Archivo no encontrado");
 }

 //Leer el archivo RDF/XML
 getOntologia().read(in,"");

 //
 // El siguiente código muestra un listado con las clases e instancias existentes en el
 modelo
 //
 System.out.println("Clases/Instancias");
 System.out.println("=====");
 ExtendedIterator iteratorClasses = getOntologia().listClasses();
 while (iteratorClasses.hasNext()){

```

```

 OntClass ontClass = (OntClass) iteratorClasses.next();
 System.out.println(ontClass);

 ExtendedIterator iteratorInstances = ontClass.listInstances();
 while (iteratorInstances.hasNext()){
 System.out.println("t"+iteratorInstances.next());
 }
 System.out.println("");
 }

}

//Crear la base de datos tomando como modelo un esquema ontologico previamente
hecho con PROTEGE
public boolean crearEsquemadeBD(){
 boolean r=false;

 setCon(new DBConnection("jdbc:mysql://localhost:3306/" + getBd_nombre(),
getBd_usuario(), getBd_pass(), getBd_manejador()));

 try{
 //El esquema se crea a partir de la bd vacia

 getCon().cleanDB();

 //Modelo JENA para el almacenamiento persistente de la bd relacional
 ModelRDB modelo_relacional=null;

 //Creacion de las tablas necesarias para almacenar el modelo 'nombre_modelo'
 //El proceso utilizará la coneccion existente
 modelo_relacional=ModelRDB.createModel(getCon(), getNombre_modelo());
 System.out.println("Base de Datos "+getBd_nombre()+" preparada!!!");

 //Cierre del modelo y de la conexion
 getCon().close();
 modelo_relacional.close();

 //La base de datos esta lista
 r=true;
 }

 catch(Exception e){
 System.out.println(e.getMessage());
 e.printStackTrace();
 }

 return r;
}

//Metodo para cargar esquema otologico en la base de datos
public boolean guardarModeloBD(){

}

```

```

boolean r=false;
setCon(new DBConnection("jdbc:mysql://localhost:3306/" + getBd_nombre(),
getBd_usuario(), getBd_pass(), getBd_manejador()));

//maker se utilizará más adelante en la apertura del modelo del tipo ModelRDB
//Se indica la conexión que se utilizará
setMaker(ModelFactory.createModelRDBMaker(getCon()));

//Existe el modelo 'nombre_modelo' en la BD??
if(!maker.hasModel(nombre_modelo)){
 System.out.println("El modelo "+getNombre_modelo()+" no existe!!!");
}
else{
 //Apertura del modelo llamado 'nombre_modelo' almacenado en la BD
 setModeloR((ModelRDB) getMaker().openModel(getNombre_modelo()));
 /*Descripción de los componentes del modelo de la ontología, esquema de
almacenamiento, razonador.
 * OWL_MEM_RULE_INF: Modelo OWL almacenado en memoria, con
inferencias
 * OWL_MEM: Modelo OWL almacenado en memoria, no se ejecuta el
razonador
 * NOTA: OWL_MEM_RULE_INF--->"Java heap space"
 * OWL_MEM --->Ejecución normal
 */
}

//Creación del modelo OntModel utilizando el modelo persistente modelR
OntModel
ont=ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM,getModeloR());

//Inicialmente, el modelo modeloR no contiene datos, obviamente ont tampoco
//Se añade el modelo que se desea almacenar y que ha sido previamente creado
con PROTEGE
cargarOntología();

//La siguiente acción implica el almacenamiento inmediato de la ontología en la
base de datos
//a través de modeloR con la conexión creada al inicio del método
ont.add(getOntología());

System.out.println("Se ha guardado una instancia OntModel en la BD");
r=true;
try{
}catch(Exception e){
 System.out.println(e.getMessage());
 e.printStackTrace();
}

}

return r;
}

```

```

public void cargarModeloBD(){
 //Conexion a la base de datos local
 setCon(new DBConnection("jdbc:mysql://localhost:3306/" + getBd_nombre(),
 getBd_usuario(), getBd_pass(), getBd_manejador()));
 //Apertura del modelo existente en la base de datos
 setMaker(ModelFactory.createModelRDBMaker(getCon()));
 setModeloR((ModelRDB) getMaker().openModel(getNombre_modelo()));

 //Carga y generacion de la Ontologia ontologia
 setOntologia(ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM,
 getModeloR()));
 /*A partir de este punto, todos los cambios realizados sobre 'ontologia' se reflejaran
 * igualmente en la base de datos (incluso si no se ha creado la conexion)*/
}

//Método para que una vez realizadas las operaciones sobre el modelo ontológico,
//cerrar conexión a él.
public void cerrarModeloOntologico(){
 try{//Cierre del modelo y conexiones
 getOntologia().close();
 getModeloR().close();
 getCon().close();

 }catch(Exception e){
 System.out.println("ERROR: "+e.getMessage());
 }
}

public static void main(String args[]){
 ConexionBD c=new
 ConexionBD("sistemaautentificacion","root","password","MySQL","autentificacion");

 //Esto solo se hace una vez para prepara la base de datos
 //Se prepara la base de datos
 c.crearEsquemadeBD();
 //Se carga a la BD el esquema ontologico generado con PROTEGE
 c.guardarModeloBD();

}

public OntModel getOntologia() {
 return ontologia;
}

public void setOntologia(OntModel ontologia) {
 this.ontologia = ontologia;
}

```

```
public IDBConnection getCon() {
 return con;
}

public void setCon(IDBConnection con) {
 this.con = con;
}

public ModelMaker getMaker() {
 return maker;
}

public void setMaker(ModelMaker maker) {
 this.maker = maker;
}

public ModelRDB getModeloR() {
 return modeloR;
}

public void setModeloR(ModelRDB modeloR) {
 this.modeloR = modeloR;
}

public String getBd_nombre() {
 return bd_nombre;
}

public void setBd_nombre(String bd_nombre) {
 this.bd_nombre = bd_nombre;
}

public String getBd_usuario() {
 return bd_usuario;
}

public void setBd_usuario(String bd_usuario) {
 this.bd_usuario = bd_usuario;
}

public String getBd_pass() {
 return bd_pass;
}

public void setBd_pass(String bd_pass) {
 this.bd_pass = bd_pass;
}

public String getBd_manejador() {
 return bd_manejador;
}
```

```

}

public void setBd_manejador(String bd_manejador) {
 this.bd_manejador = bd_manejador;
}

public String getNombre_modelo() {
 return nombre_modelo;
}

public void setNombre_modelo(String nombre_modelo) {
 this.nombre_modelo = nombre_modelo;
}
}

//ARCHIVO Usuario.java
package com;

public class Usuario {
 //Datos de identificación oficial de Usuario
 private int id_usuario;
 private String nombre;
 private String appat;
 private String apmat;
 private String domicilio;
 private int folio;
 private int edad;
 private String sexo;
 private int anio_registro;
 private String clave;
 private int estado;
 private int distrito;
 private int municipio;
 private int localidad;
 private int seccion;

 private int fase;
 //

 public Usuario(){}
 public Usuario(int id_usuario, String nombre, String appat, String apmat ,String
domicilio, String clave, String sexo,
 int edad, int folio, int anio_registro, int estado, int distrito, int municipio, int
localidad, int seccion){
 this.id_usuario=id_usuario;
 this.nombre=nombre;
 this.appat=appat;
 this.apmat=apmat;
 }
}

```

```
 this.domicilio=domicilio;
 this.folio=folio;
 this.anio_registro=anio_registro;
 this.clave=clave;
 this.estado=estado;
 this.distrito=distrito;
 this.municipio=municipio;
 this.localidad=localidad;
 this.seccion=seccion;
 this.sexo=sexo;
 this.edad=edad;
 this.fase=1;
}

public int getId_usuario() {
 return id_usuario;
}

public void setId_usuario(int id_usuario) {
 this.id_usuario = id_usuario;
}

public String getNombre() {
 return nombre;
}

public void setNombre(String nombre) {
 this.nombre = nombre;
}

public String getAppat() {
 return appat;
}

public void setAppat(String appat) {
 this.appat = appat;
}

public String getApmat() {
 return apmat;
}

public void setApmat(String apmat) {
 this.apmat = apmat;
}

public String getDomicilio() {
 return domicilio;
}
```

```
public void setDomicilio(String domicilio) {
 this.domicilio = domicilio;
}

public int getFolio() {
 return folio;
}

public void setFolio(int folio) {
 this.folio = folio;
}

public int getAnio_registro() {
 return anio_registro;
}

public void setAnio_registro(int anio_registro) {
 this.anio_registro = anio_registro;
}

public String getClave() {
 return clave;
}

public void setClave(String clave) {
 this.clave = clave;
}

public int getEstado() {
 return estado;
}

public void setEstado(int estado) {
 this.estado = estado;
}

public int getDistrito() {
 return distrito;
}

public void setDistrito(int distrito) {
 this.distrito = distrito;
}

public int getMunicipio() {
 return municipio;
}

public void setMunicipio(int municipio) {
```

```

 this.municipio = municipio;
 }

public int getLocalidad() {
 return localidad;
}

public void setLocalidad(int localidad) {
 this.localidad = localidad;
}

public int getSeccion() {
 return seccion;
}

public void setSeccion(int seccion) {
 this.seccion = seccion;
}

public int getEdad() {
 return edad;
}

public void setEdad(int edad) {
 this.edad = edad;
}

public String getSexo() {
 return sexo;
}

public void setSexo(String sexo) {
 this.sexto = sexo;
}

public int getFase() {
 return fase;
}

public void setFase(int fase) {
 this.fase = fase;
}
}

//ARCHIVO Biometrico.java
package com;

public class Biometrico {
 private String id_biometrico;

```

```

//Patrones de identificación unica de un individuo
private String[] vectores;
private String distorsion;

public Biometrico(String id_biometrico, String[] vectores, String distorsion){
 this.id_biometrico=id_biometrico;
 this.vectores=vectores;
 this. distorsion=distrosion;
}

public Biometrico(){
 vectores=new String[32];
}

////////////////////////////SETERS Y GETERS////////////////////////////
public String getId_biometrico() {
 return id_biometrico;
}
public void setId_biometrico(String id_biometrico) {
 this.id_biometrico = id_biometrico;
}

public String getDistorsion() {
 return distorsion;
}
public void setDistorsion(String distorsion) {
 this.distorsion = distorsion;
}

public String[] getVectores() {
 return vectores;
}

public void setVectores(String[] vectores) {
 this.vectores = vectores;
}
}

//ARCHIVO ManejadorOntologico.java
package com;

import BaseDatos.ConexionBD;
import com.hp.hpl.jena.ontology.Individual;
import com.hp.hpl.jena.ontology.OntClass;
import com.hp.hpl.jena.ontology.OntModel;
import com.hp.hpl.jena.rdf.model.Property;
import com.hp.hpl.jena.rdf.model.RDFNode;
import com.hp.hpl.jena.rdf.model.Statement;

```

```

public class ManejadorOntologico {
 private OntModel ontologia;

 private String
uri="http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/Autentificacion
Usuarios.owl#";

 public ManejadorOntologico(){
 ConexionBD
 ConexionBD("sistemaautentificacion","root","password","MySQL","autentificacion");
 //Se carda la ontologia guarda en la base de datos
 c=new
c.cargarModeloBD();
 ontologia=c.getOntologia();

 }

 //Método para ingresar un nuevo usuario al sistema.
 //Recibe el Usuario a ser creado y el Biometrico a el asociado
 //En este caso solo recibe biometricos de tipo BiometricoVoz
 public void crearUsuario(Usuario u){
 //Se agrega un nuevo individuo de la clase usuario y se le asocia su biometrico
 OntClass usuario=ontologia.getOntClass(uri+"Usuario");

 if(usuario==null)
 System.out.println("No se escuentra al usuario");

 Individual i=usuario.createIndividual(uri+Integer.toString(u.getId_usuario()));
 i.addProperty(ontologia.getProperty(uri+"id_usuario"),
 Integer.toString(u.getId_usuario()));
 i.addProperty(ontologia.getProperty(uri+"nombre"), u.getNombre());
 i.addProperty(ontologia.getProperty(uri+"appat"), u.getAppat());
 i.addProperty(ontologia.getProperty(uri+"apmat"), u.getApmat());
 i.addProperty(ontologia.getProperty(uri+"edad"), Integer.toString(u.getEdad()));
 i.addProperty(ontologia.getProperty(uri+"sexo"), u.getSexo());
 i.addProperty(ontologia.getProperty(uri+"domicilio"), u.getDomicilio());
 i.addProperty(ontologia.getProperty(uri+"folio"), Integer.toString(u.getFolio()));
 i.addProperty(ontologia.getProperty(uri+"anio_registro"),
 Integer.toString(u.getAnio_registro()));
 i.addProperty(ontologia.getProperty(uri+"clave"), u.getClave());
 i.addProperty(ontologia.getProperty(uri+"estado"),
 Integer.toString(u.getEstado()));
 i.addProperty(ontologia.getProperty(uri+"distrito"),
 Integer.toString(u.getDistrito()));
 i.addProperty(ontologia.getProperty(uri+"municipio"),
 Integer.toString(u.getMunicipio()));
 i.addProperty(ontologia.getProperty(uri+"localidad"),
 Integer.toString(u.getLocalidad()));
 i.addProperty(ontologia.getProperty(uri+"seccion"),
 Integer.toString(u.getSeccion()));

 }
}

```

```

 i.addProperty(ontologia.getProperty(uri+"fase"), Integer.toString(u.getFase()));

 ontologia.commit();

 ontologia.write(System.out);
 }

public void crearBiometrico(Biometrico bio, int id_usuario){
 OntClass biometrico=ontologia.getOntClass(uri+"Biometrico");
 Individual i2=biometrico.createIndividual(uri+bio.getId_biometrico());
 //Estableciendo propiedad de tipo objeto
 Property pName = ontologia.getProperty(uri+Integer.toString(id_usuario));
 RDFNode rdfName = (RDFNode)pName.as(RDFNode.class);
 i2.setPropertyValue(ontologia.getProperty(uri+"usuario_biometrico"), rdfName);

 String[] vectores=bio.getVectores();
 for(int i=0;i<32;i++){
 i2.addProperty(ontologia.getProperty(uri+"v"+(i+1)), vectores[i]);
 }

 i2.addProperty(ontologia.getProperty(uri+"distorsion"), bio.getDistorsion());

 ontologia.commit();
}

//Metodo para obtener el determinado Biometrico de Voz de un usuario a autenticar
//El usuario introduce su clave de usuario y a la vez su voz para ser autenticado
public Biometrico obtenerBiometrico(int id_usuario){
 Biometrico b=null;

 Individual ind=ontologia.getIndividual(uri+"voz"+id_usuario);
 if(ind!=null){
 b=new Biometrico();

 Statement st=ind.getProperty(ontologia.getProperty(uri+"distorsion"));
 b.setDistorsion(st.getString());

 String[] vectores=b.getVectores();

 for(int i=0;i<32;i++){
 st=ind.getProperty(ontologia.getProperty(uri+"v"+(i+1)));
 vectores[i]=st.getString();
 }
 }
 else
 System.out.println("No se encuentra al Biometrico");
 return b;
}

//Metodo para obtener el determinado Biometrico de Voz de un usuario a autenticar

```

```

//El usuario introduce su clave de usuario y a la vez su voz para ser autenticado
public Usuario obtenerUsuario(int id_usuario){
 Usuario u=null;

 Individual ind=ontologia.getIndividual(uri+id_usuario);
 if(ind!=null){
 u=new Usuario();
 u.setId_usuario(id_usuario);

 Statement st=ind.getProperty(ontologia.getProperty(uri+"nombre"));
 u.setNombre(st.getString());
 st=ind.getProperty(ontologia.getProperty(uri+"appat"));
 u.setAppat(st.getString());
 st=ind.getProperty(ontologia.getProperty(uri+"apmat"));
 u.setApmat(st.getString());
 st=ind.getProperty(ontologia.getProperty(uri+"edad"));
 u.setEdad(st.getInt());
 st=ind.getProperty(ontologia.getProperty(uri+"sexo"));
 u.setSexo(st.getString());
 st=ind.getProperty(ontologia.getProperty(uri+"domicilio"));
 u.setDomicilio(st.getString());
 st=ind.getProperty(ontologia.getProperty(uri+"folio"));
 u.setFolio(st.getInt());
 st=ind.getProperty(ontologia.getProperty(uri+"anio_registro"));
 u.setAnio_registro(st.getInt());
 st=ind.getProperty(ontologia.getProperty(uri+"clave"));
 u.setClave(st.getString());
 st=ind.getProperty(ontologia.getProperty(uri+"estado"));
 u.setEstado(st.getInt());
 st=ind.getProperty(ontologia.getProperty(uri+"distrito"));
 u.setDistrito(st.getInt());
 st=ind.getProperty(ontologia.getProperty(uri+"municipio"));
 u.setMunicipio(st.getInt());
 st=ind.getProperty(ontologia.getProperty(uri+"localidad"));
 u.setLocalidad(st.getInt());
 st=ind.getProperty(ontologia.getProperty(uri+"seccion"));
 u.setSeccion(st.getInt());
 st=ind.getProperty(ontologia.getProperty(uri+"fase"));
 u.setFase(st.getInt());
 }
 else
 System.out.println("No se encuentra al Usuario");

 return u;
}

public boolean subirFaseUsuario(int id_usuario){
 boolean r=false;

 Individual ind=ontologia.getIndividual(uri+id_usuario);

```

```

 if(ind!=null){
 Statement st=ind.getProperty(ontologia.getProperty(uri+"fase"));
 int fase=st.getInt();
 fase++;
 st.changeObject(Integer.toString(fase));

 ontologia.commit();
 }
 else
 System.out.println("No se escuentra al Usuario");

 return r;
 }
}

```

#### //EJEMPLO DE FUNCIONAMIENTO

```

public static void main(String args[]){
 ManejadorOntologico mo=new ManejadorOntologico();
 Usuario u=new Usuario(23,"Luz","Maria","Guzman","Miranda",
 "conocido","MyClave","M",21,15,2006,15,0,105,1,5082);
 String[] vectores={"29.309645127935593 20.853206816045073
7.439592302933658 8.626174945957045 5.8264665061824905 4.970954531827567
2.883281984547241 4.463699763821816 3.7512530592123743 1.593661727360119
1.2748517607809204 2.273457605865112 2.7528918416898573 2.5981475236459803
0.8991471010919279 2.1686326557161757 2.4997374527032266
1.5096932832156469 2.1080346561354464 1.6089075970032658 ",
"153.0970722413339 53.299642266164796 -5.43499949975232
10.988555185556402 -3.053125343159971 -7.20449784009769 -8.385029982482271
6.915129487803637 1.7489637230169208 4.579881462658105 -7.918779025313321 -
3.7573371100431445 5.291754202054211 -7.298965574026199 -2.743132461317874
3.84004959502017 -3.6862725874006816 3.1802405187333727 -1.0476006837825482
1.8363405933089834 ",
"153.08523689516437 55.67602788574365 -11.411403771622037 -
0.2528772291264173 2.613842170587476 -5.505946704831578 0.3013610572465678
5.8412397254434305 1.8447251377645113 6.867575411098317 -
11.270274692468451 -6.416018651825038 1.884685150972557 -0.5033944078220214
-4.0282990189901104 4.866728472288473 -4.238484177994996 2.7877280000372497
-3.5240495025764624 3.625762658912505 ",
"31.930121642198547 18.627594696202305 6.100047702514185
8.2296321545652 3.600419570642974 6.9528735971617 4.1839028664387685
8.161635375679614 5.045429783649264 -1.3316528505946503 2.046753013867167
3.5756200043910114 0.881715708994262 7.248495586085997 8.010840561551655
1.8759263675343796 -0.7269162360036654 0.7919250122591209
2.3625599224795546 -1.3484797303523497 ",
"145.26111119361343 66.1786937996308 -9.254872932391757 -
9.75614010860409 -8.459379413850403 7.016411766823382 11.262687480161615
0.9200071042362908 -6.428156270792918 -1.2592283122850236 -
3.4984969765762286 -0.03264163838558939 -8.426260176549945
5.797855339043728 -4.569246439215381 3.686142813393059 -2.074686757160156
0.692630125831195 -1.2933037986059865 2.7048300958412463 ",

```

"116.03058566733502 54.438137228605065 6.593787796967056  
 16.70261291942734 4.499866085859069 -11.700080336003408 -11.501089799237768  
 6.691820914672738 -2.0793580545624977 3.2387013124183657 -3.798185409512389  
 0.6615677153831124 4.80964160921256 -5.1427238178059715 -4.815359486971398  
 3.0676912592862724 -3.4485261653322588 0.6370400675304831 -  
 0.8684698449184456 3.4554974593984498 ",  
 "128.09509179184357 45.175826723947836 9.931069500929137  
 24.438908232677516 -0.2323261968861659 -14.289765702101537 -  
 10.558713558670478 11.903077108771663 -11.909196380597468  
 3.337090310280496 -2.0180425944601916 2.58938736247483 1.8263040159667205 -  
 4.6255482629392715 -1.560221926104685 2.5832580997246852 -5.299442321058284  
 0.6897509575611327 1.167563493636166 1.7955460033483817 ",  
 "171.98198551289653 47.91219546057398 -9.958771260708719 -  
 1.5587111342357984 2.149484302515132 -3.290822258067306 0.2822355225347644  
 4.425831817857385 1.8909863426131572 1.269988444151802 -12.524373329474804  
 -2.2529912633024822 1.338107000611093 -0.15005788763299727 -  
 3.5704187558808997 4.760957733892374 -6.104204071178344 4.550153989816995 -  
 2.5537832393439297 2.9821586493403296 ",  
 "147.86339934966546 53.142222006616095 2.221465879402391 -  
 2.190894563979847 5.719829275413959 -10.215014033596667 -5.8112055741826145  
 6.015625603074229 4.205765639891506 6.190866207440246 -10.158333938473437 -  
 2.0759291563137072 3.3883939652436315 -3.0533425045415923 -  
 5.658051137371179 2.575608801997452 -3.261984301257801 4.344164603121242 -  
 4.6835751144638325 4.155985231221437 ",  
 "123.6040615677931 51.64940298134205 -1.8321457399555205  
 19.211007879122935 9.427915048546843 -4.786955223066995 -3.429028172057018  
 -7.329536526726953 -12.989216150356803 12.77743003498289 -2.480930805400981  
 -7.675723182910746 -4.737107591734526 -2.675703721180154 -0.994946259270614  
 0.41483416133110274 -1.7390747411888527 5.091714844986559  
 2.5790536769586705 5.154175440433743 "  
 ,"97.80630888122127 55.02974270533498 6.993575996173714  
 21.42446711576703 8.971444947631468 -4.46662389007255 -6.2135401101400465 -  
 6.117088142135938 -9.797258051582007 11.496072071669014 -  
 0.13206554604816909 -9.28757578690858 -0.047412239726315364 -  
 1.9033908800022217 -1.3424062426368253 -1.4601810993102315 -  
 3.335650900772071 1.2147894974821343 3.842143973858674 4.3161659896219415 "  
 ,"128.7498176543145 66.6316615380771 -4.924176897910289  
 2.1135451316336105 -7.287160316599654 -1.3606250941854614 7.98408402750738  
 3.929327244676176 -7.303594141549625 1.3456797219754533 -4.230452433351189  
 -4.3367787770860176 -3.125831968907667 1.1464276099503274 -  
 0.6807178293555608 0.5663609064565331 -1.7443763255872737  
 0.6296100783780904 0.9873942632225557 2.3083083024364055 "  
 ,"109.64141904376321 57.86290386793947 -1.047715387007667  
 16.757073621856495 7.18599842073585 0.9562295330420567 8.158123346695668 -  
 3.8047805605501557 -14.366439131095827 1.1010263416844364 -  
 4.493689064064913 -6.430138606727934 -1.7182544788658143 -  
 1.8062839475486294 -1.1376197013982199 0.7832719004685916 -  
 0.7252962338306812 3.840148035949369 2.651638349393 3.765781735685071 "  
 ,"143.65083277739438 65.12833392059969 -17.22007204711332 -  
 0.7589625525132135 -0.06676486089134098 5.688198348502134

|                        |                      |                                         |
|------------------------|----------------------|-----------------------------------------|
| 2.4468199061967386     | -0.152622856627266   | -2.0436601909705145                     |
| 2.7407201697348165     | -6.270842220665674   | -5.121471460231607 -                    |
| 1.6408513372641964     | 2.6404403428779784   | -0.6234700053372728                     |
| 1.1437657437884454     | -5.951324505986552   | 3.9032934477391623 -                    |
| 1.1856147546558682     | 0.5734122428179989 " |                                         |
|                        | ,"131.82637578126136 | 52.497361646586896 -7.022864921382732   |
| 9.123435215862676      | 6.618993444124529    | 3.6223904119747323 -4.728877853111221   |
| 3.249734091651353      | -10.644708479486521  | 4.2863598338385 -3.4994725615282647 -   |
| 9.591188814235888      | -0.2019885265405855  | -0.915079616959089 -                    |
| 1.8944508374355293     | 0.7329269686929931   | -1.4658748752284882                     |
| 4.3223401993441115     | 1.4980274902092297   | 2.849211402976773 "                     |
|                        | ,"134.62781232817022 | 54.65480980642233 -5.179931014922036    |
| 8.847301533063835      | 1.2254613757301474   | -2.1727535671739417 -8.37877657477601   |
| 10.40396314273674      | 0.24434419991674994  | 2.0881831135974434 -                    |
| 6.296905420308544      | -5.084841658719173   | 4.961237844848607 -4.119888739440161 -  |
| 2.795355261278882      | 1.5772110237202355   | -4.604046962199448 3.1043474658476 -    |
| 2.1436362972466725     | 3.7053701955540688 " |                                         |
|                        | ,"145.8757823200031  | 43.53714764881357 2.5330946490945645    |
| 18.968333934011007     | -4.73344332339073    | -13.405381633016376 -                   |
| 4.203316299889659      | 14.38155763727023    | -6.547289866536024 3.008375796724283 -  |
| 3.4390072688614497     | 1.3976796534342655   | 0.11323215342149784 -                   |
| 8.003917833872572      | 0.18659629042863218  | 1.7856921003232749 -                    |
| 6.019323649824112      | 3.638989315259665    | 0.2732792894819824 0.8301746436670099 " |
|                        | ,"143.6765178786212  | 40.785245852796244 10.852939761402142   |
| 20.010851136147277     | -1.083495032113542   | -21.641844768037313 -                   |
| 2.2471542786269025     | 8.531852570890036    | -8.044606709013934 3.9096964110785453   |
| -0.5990095192035887    | 3.4174603480685084   | -0.6250414708588587 -                   |
| 4.553715574493878      | 0.05279127792809803  | -0.7528403170451471 -                   |
| 7.669576718438795      | 4.042771264641365    | -0.20455781315543714                    |
| 3.2172901807087197 "   |                      |                                         |
|                        | ,"110.71044006297302 | 47.67211886707726 -6.98072491377561     |
| 1.9931613158268344     | 6.882000595441721    | 9.605698495278755 -1.7396094218694638   |
| 5.233802172185492      | 1.7724186765857999   | -1.4763928062176541 -9.895052120926099  |
| -1.2835990732782527    | 5.392031892438884    | 0.9154509433235899 -5.513821987928976   |
| 2.6069353513745246     | -7.371576004877489   | 2.376314784015783 -1.0478518489849316   |
| -0.19396912847589154 " |                      |                                         |
|                        | ,"157.67186014493382 | 51.746332118048926 -15.62938263782285   |
| 12.396182292337443     | -7.195359332453962   | 1.2338094821361199 -8.463246659258044   |
| 15.628773065624218     | -3.0614411107584742  | -1.7396095905796591 -                   |
| 9.130979100496852      | -3.8368617922498016  | 5.097280333726751 -6.966616503898366    |
| -0.9345781473066296    | 3.9530368392081217   | -3.984200636647495                      |
| 2.3142170949392407     | 0.790608047397161    | 1.672234998927037 "                     |
|                        | ,"145.66377293254794 | 57.52566035433974 -14.625349789882605   |
| 2.501961106720204      | 0.6284076331165587   | 5.654922981918822 -6.686976210631233    |
| 5.6604211290289115     | -1.3655543799224001  | 1.987655073193367 -6.787312333547056    |
| -10.036260680096456    | 4.108557582695016    | 0.2802860439104478                      |
| 0.14845039148008918    | -0.8898301208044987  | -4.874853657912645                      |
| 5.535914980299729      | -1.127401749505967   | 2.0083194447497794 "                    |
|                        | ,"161.54202732078974 | 58.333132240110835 -19.15543263501713 - |
| 0.11540447317959189    | 6.347417688392891    | 0.42008348279319846 -                   |

|                      |                      |                       |                     |
|----------------------|----------------------|-----------------------|---------------------|
| 6.7294795692470935   | 6.268641846803561    | 0.5024189417025126    | -                   |
| 0.6349634141250361   | -10.689839575250717  | -1.3935861562985956   |                     |
| 1.6467506922040818   | 2.5229325533902767   | -4.609691981406109    |                     |
| 1.2509431326207279   | -4.358204816432983   | 3.613644989537665     | -2.416397999501659  |
| 3.3161721029364983 " |                      |                       |                     |
|                      | ,"157.25299396502243 | 55.80235034114398     | -18.276487122240546 |
| 0.7933022488815842   | 2.855394700584684    | 4.880321499345186     | -1.6049532712518673 |
| 2.151993920167126    | -0.21669035955812432 | 3.9638259127650954    | -12.4763914941308   |
| -4.421527779499845   | 2.8713161804938854   | 0.738803655787669     | -2.2886123860555885 |
| 2.529935156068923    | -6.932258059005787   | 5.362902734215157     | -0.8259519967707907 |
| 1.0544176077955698 " |                      |                       |                     |
|                      | ,"160.91236371304717 | 58.731698818195525    | -14.308960409536574 |
| 5.937393514161034    | -0.2992366908551465  | 6.793181325051729     | 4.7285625649112655  |
| -4.4617402137309154  | 0.4049427632171884   | 4.012528527242284     | -7.995060599777553  |
| -3.3737257470438657  | -6.933798341286634   | 7.2196480280179465    | -                   |
| 3.091346067208366    | 4.444498759595152    | -5.85446983496051     | 1.9217618016749607  |
| 0.755016531038439    | 1.1563921840302585 " |                       |                     |
|                      | ,"122.75791372211759 | 51.91770453234117     | -1.5436839993055436 |
| 19.1396560783135     | 7.9248195452155485   | 3.13639715169493      | 8.308379291289176   |
| 4.582190683641571    | -14.660381534612066  | 1.1555830438584689    | -6.233421899210443  |
| -5.642195968557775   | -2.973360114675407   | -0.44241999450270475  | -                   |
| 1.3302049960921436   | 0.5647539693741896   | -0.28613749183293724  |                     |
| 3.94981077245727     | 3.741825883877649    | 3.2381007046169343 "  |                     |
|                      | ,"118.66037224909532 | 56.2972207946712      | 0.19148631438078992 |
| 16.05573914712273    | 0.48462185709134353  | -1.4116786842027222   |                     |
| 7.833503474129437    | -2.221942325248196   | -10.489340246700843   | 4.747258465885284   |
| -6.378489172185376   | -6.670969870697533   | -4.005130337898005    | -2.739119565888522  |
| 0.3571947284437507   | 1.7784316860649936   | -1.4132204041909695   |                     |
| 3.3131856348854734   | 3.1249241830543113   | 2.7411990032758466 "  |                     |
|                      | ,"137.1244983369803  | 46.818312652344815    | 5.431801452162468   |
| 7.785600874493904    | 7.42122604159088     | -12.872133876488668   | -6.105049597869046  |
| 10.43082203954732    | -0.18912319630258162 | 1.6617366847643251    | -                   |
| 7.284806841379799    | 3.021874683953873    | -0.11430012701185321  | -4.583937717673477  |
| -4.234961496774325   | 4.2393138223574764   | -4.626886366853456    | 1.6095535458202919  |
| -2.35476189943911    | 4.5107968594160255 " |                       |                     |
|                      | ,"153.85395505718074 | 38.91840838511652     | 4.343178999150674   |
| 26.525309295201932   | -13.462979391891286  | -11.239739207407071   | -                   |
| 9.924657651521029    | 13.964347673191016   | -14.894656301308478   | 7.24136558421403    |
| 1.7320623429053779   | 2.2158416422991807   | 0.36724606736702475   | -                   |
| 7.025388646028003    | 0.15745490931930775  | -0.82899462642313     | -5.859230284584491  |
| 4.648854817807337    | 4.136521598132906    | 0.16439899837612573 " |                     |
|                      | ,"157.39594756512088 | 40.53912684645709     | 8.213146346135549   |
| 25.83946177978556    | -7.039533844011276   | -20.252314673505627   | -2.157475577112162  |
| 7.1483333043319846   | -13.131383121745415  | 6.353614667411858     | -                   |
| 1.2737333438723568   | 1.908288823261016    | -1.5094524636078852   | -3.709474948210969  |
| -0.30315945668671385 | -0.5482810461684906  | -5.469792046790762    |                     |
| 6.674758190202651    | -0.30074749663570477 | 1.3386262411916774 "  |                     |
|                      | ,"162.92588018143118 | 50.8679932677549      | -16.158298588572425 |
| 1.769453031115156    | 6.8151244302600755   | 2.7936345773649434    | -8.28156950241693   |
| 9.439547481667427    | -4.145090362217803   | -1.2398548982652904   | -8.328486252850341  |

```

-6.5587058731836185 6.06404325554533 -0.13099898078361835 -
3.198847842813902 2.0147646016637037 -4.471575257045969 5.107968011353633 -
1.942859496734455 0.701331848054573 "
,"172.34665664306834 40.51494114762657 -0.39781244171418556
19.928548025519643 -12.185135976449889 -16.03909849006889 -
2.989942399776141 11.468204956073668 -8.89059463062078 4.97992587261151 -
1.483572343498488 2.037234380498791 -0.905917927271182 -6.682331606867763
-1.6013718656056646 0.7251894380057665 -7.305587422050829 7.145943050097015
1.8864659412746185 1.8726144356086163 "
,"161.1182883932686 48.25759305281039 -8.596403247858678
20.92559454573236 -11.68006243264535 -14.070986881282161 -6.115896785224966
15.943910075216536 -2.8780758956880814 2.894030775962172 -
3.5856171123578418 0.5965713610237491 0.10161046012577828 -
8.83130570312617 0.48198798829307793 2.0411303926319104 -
6.378862495074261 2.639819285051993 1.2135341141390532 3.3609246788957705
"};

```

```

Biometrico b=new Biometrico("voz"+u.getId_usuario(),vectores,"distorsion");
System.out.println("Biometrico encontrado!!!!"+b.getVectores()[0]);

//Se crea un usuario en la base de datos
mo.crearUsuario(u);

//Se crea le biometrico
mo.crearBiometrico(b, u.getId_usuario());

//Se incrementa la fase del usuario
mo.subirFaseUsuario(u.getId_usuario());

//Se recupera el usurio
Usuario u2=mo.obtenerUsuario(23);
if(u2!=null)
 System.out.println("Usuario encontrado!!!!. FASE: "+u2.getFase());

//Se recupera el biometrico
Biometrico b2=mo.obtenerBiometrico(u.getId_usuario());
if(b2!=null)
 System.out.println("Biometrico encontrado!!!!");
}
}
}
```

# CAPÍTULO 5. PRUEBAS, RESULTADOS, CONCLUSIONES Y TRABAJO A FUTURO

---

## 5.1 CAPA DE USUARIO

---

Se realizaron pruebas registrando a diversos usuarios, y posteriormente se intentó autenticarlos con el fin conocer el rango de falsos positivos en la implementación.

## 5.2 CAPA DE NAVEGACIÓN

---

Al hacer pruebas el navegador presentó diversas fallas, por ejemplo en ocasiones al abrir la página no da las indicaciones con voz y es necesario reiniciarlo para que la aplicación de ejecute correctamente.

Otro problema que presentó en la grabación de audio, es que se debe ser muy cuidadoso al momento de grabar la voz y verificar que el micrófono este en buenas condiciones ya que de lo contrario se escucha interferencia o la voz no es grabada correctamente.

Se realizaron pruebas satisfactorias en las interfaces de usuario siguientes:

- **Modulo de captura de voz**

La capa de usuario cuenta con una pantalla de login donde el usuario se registra, ya sea en modo de alta o como un usuario activo que desea accesar a un servicio.

En esta interfaz la página web da las instrucciones por medio de voz para pedir al usuario ingresar su contraseña y que pueda accesar a los servicios.

El usuario debe presionar el botón de voz mientras dice su contraseña y soltarlo en cuanto termine, después deberá presionar el botón de login.

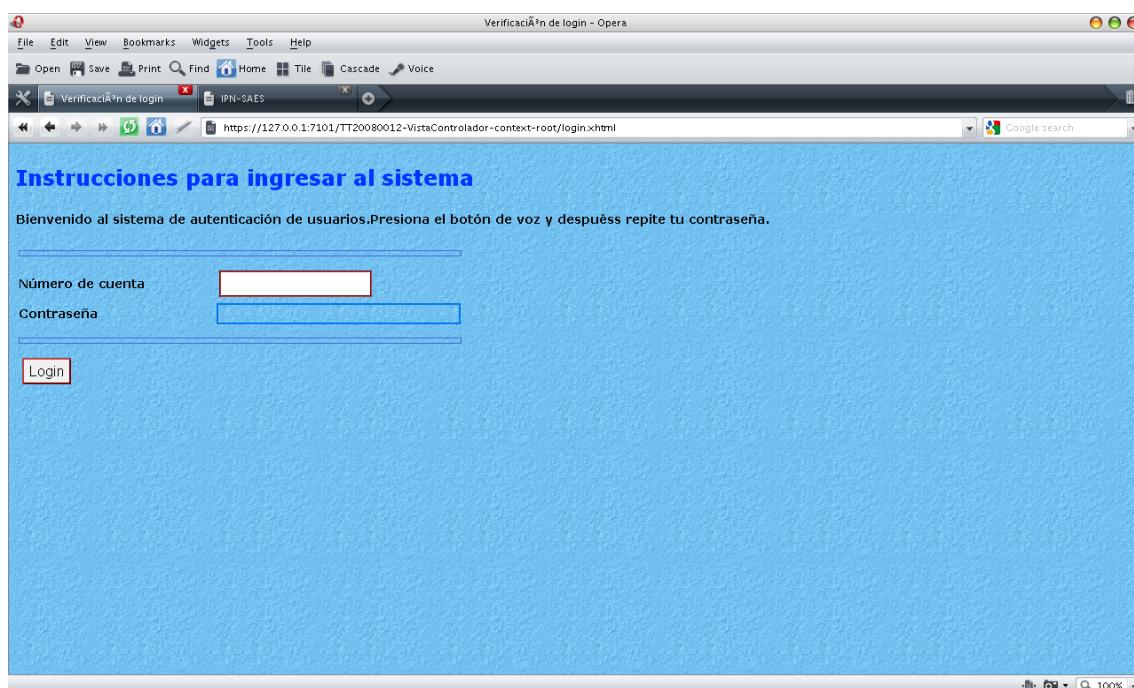


Figura 26. Login de usuarios

- **Modulo de registro de Usuarios**

**Registro de Usuarios**

**Nombre:**   
**Edad:**   
**Sexo:**   
**Domicilio:**   
**Folio:**   
**Anio de Registro:**   
**Clave de Elector:**   
**Estado:**   
**Distrito:**   
**Municipio:**   
**Localidad:**   
**Sección:**   
**Apellido Paterno:**   
**Apellido Materno:**   
**Perfil:**

**Crear Usuario**

Figura 27. Registro de usuarios

- **Modulo de Modificación de Usuarios**

| Clave de Elector | Nombre | Apellido Paterno | Apellido Materno | Modificar                 |
|------------------|--------|------------------|------------------|---------------------------|
| 1232             | Joel   | Villanueva       | Chavez           | <a href="#">Modificar</a> |

**Detalles del usuario seleccionado**

**Nombre:**   
**Edad:**   
**Sexo:**   
**Domicilio:**   
**Folio:**   
**Anio de Registro:**   
**Clave de Elector:**   
**Estado:**   
**Distrito:**   
**Municipio:**   
**Localidad:**   
**Sección:**   
**Apellido Paterno:**   
**Apellido Materno:**   
**Perfil:**

**Guardar Cambios**

Figura 28. Modificación de usuarios

- **Modulo de eliminación**

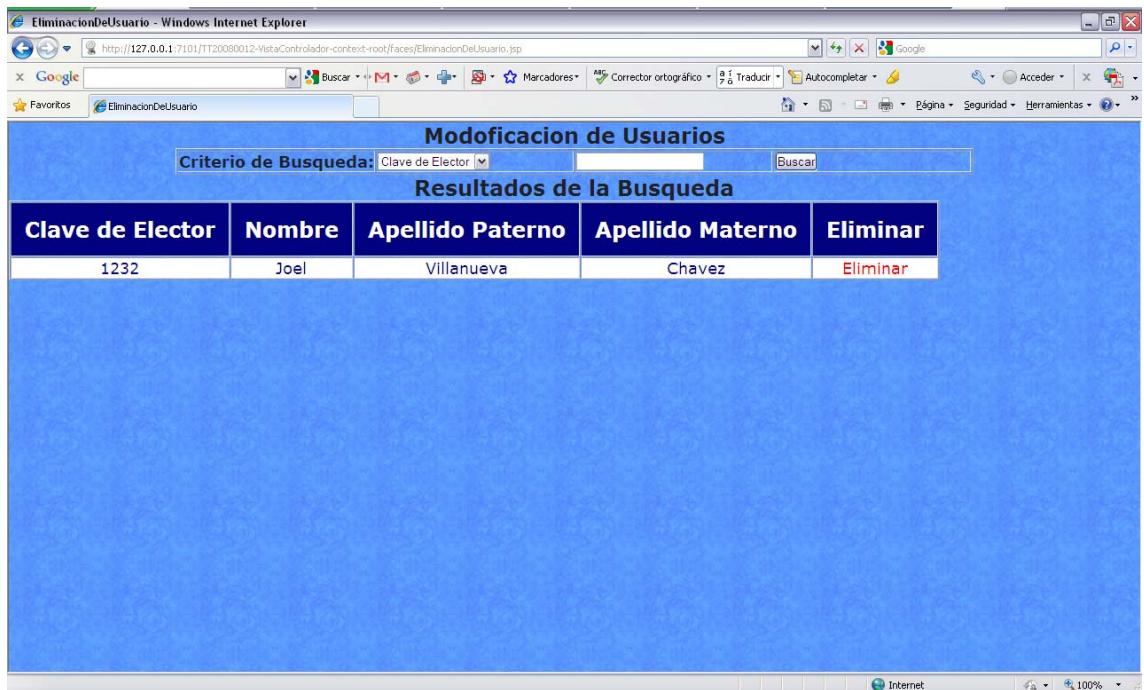


Figura 29. Eliminar usuario

- **Modulo de asignación de servicios a usuario:**



Figura 30. Asignación de usuarios

- **Modulo de consultas de usuarios:**

The screenshot shows a web application window titled "Modificacion de Usuarios". At the top, there is a search bar labeled "Criterio de Busqueda: Clave de Elector" with a dropdown menu and a "Buscar" button. Below the search bar, the title "Resultados de la Busqueda" is displayed. A table with the following columns is shown:

| Clave de Elector | Nombre | Apellido Paterno | Apellido Materno | Eliminar                 |
|------------------|--------|------------------|------------------|--------------------------|
| 1232             | Joel   | Villanueva       | Chavez           | <a href="#">Eliminar</a> |

Figura 31. Consulta de usuarios.

- **Modulo de registro de Servicios:**

The screenshot shows a web application window titled "Registro de Servicios". The form has two input fields: "Nombre:" and "Servicio (Archivo:)". Below the "Servicio (Archivo:)" field is a "Examinar..." button. At the bottom right of the form is a "Guardar" button.

Figura 32. Registro de servicios

- **Modulo de modificación de servicios.**

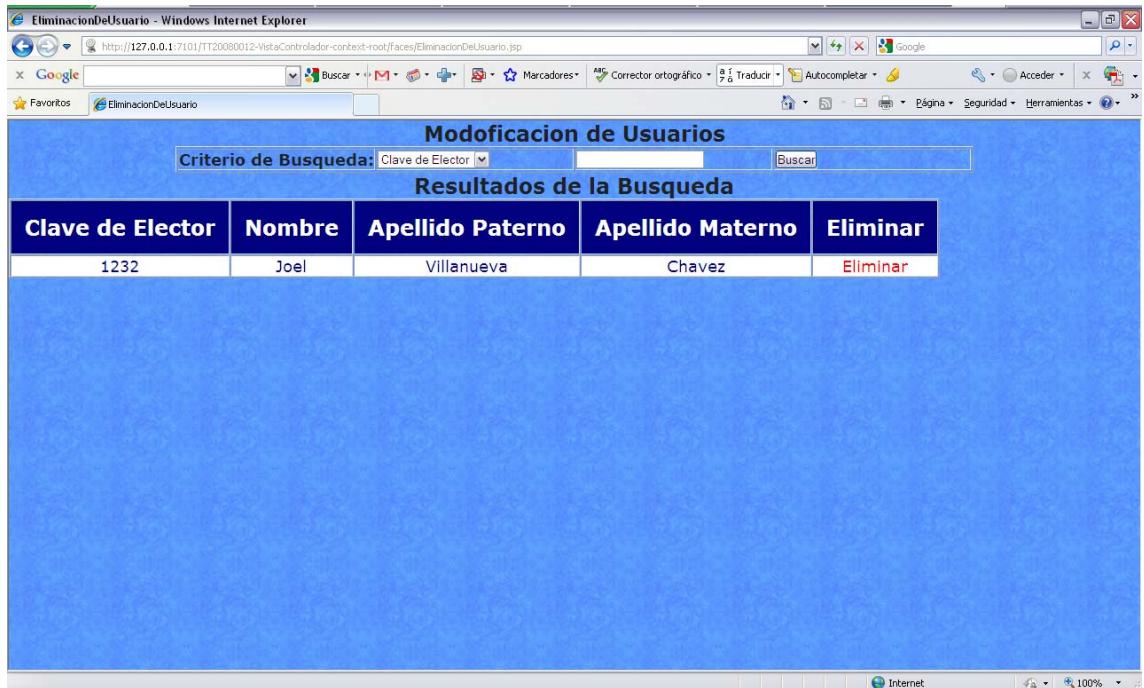


Figura 33. Modificar servicios

- **Modulo de Eliminación de Servicios**

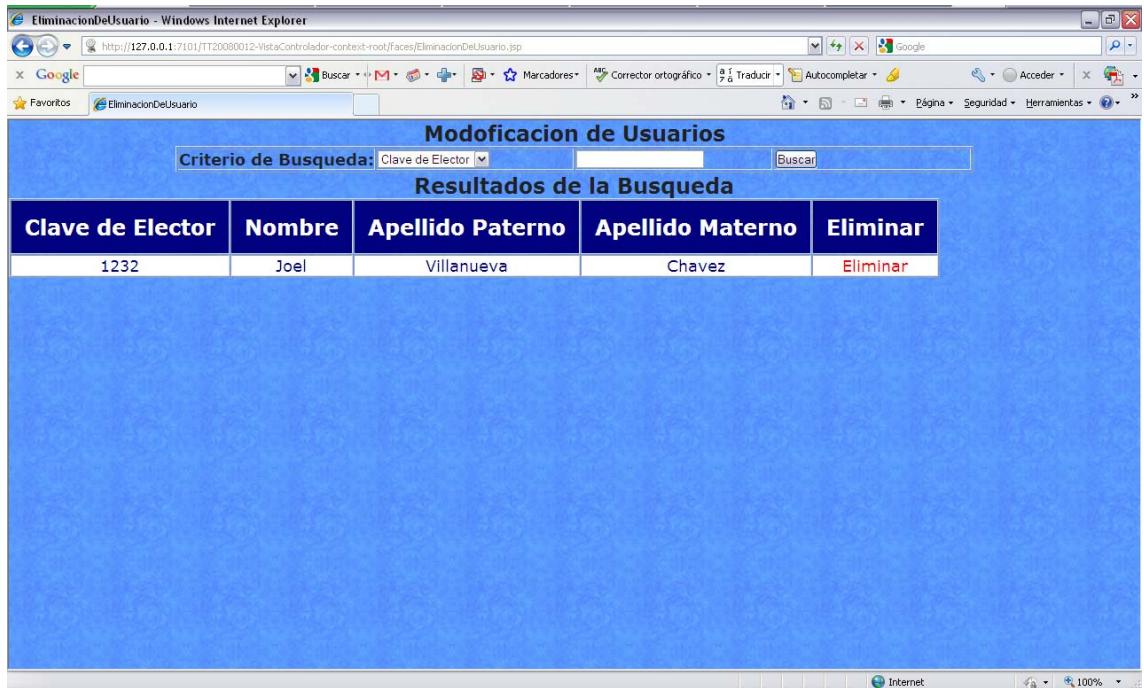


Figura 34. Eliminar servicios

## CAPA ZONA DE INTERNET

Mediante la configuración del servidor se obtuvo una página HTTPS para garantizar el envío de datos por un canal cifrado.

Comprobamos que el servidor acepta páginas HTTPS.

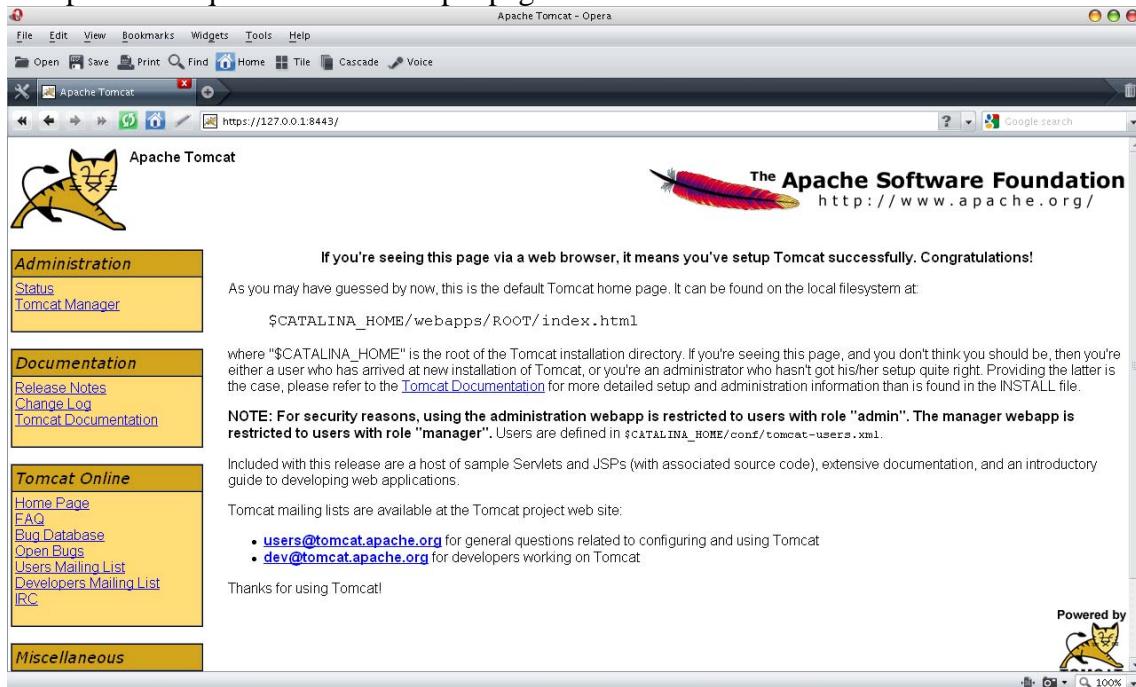
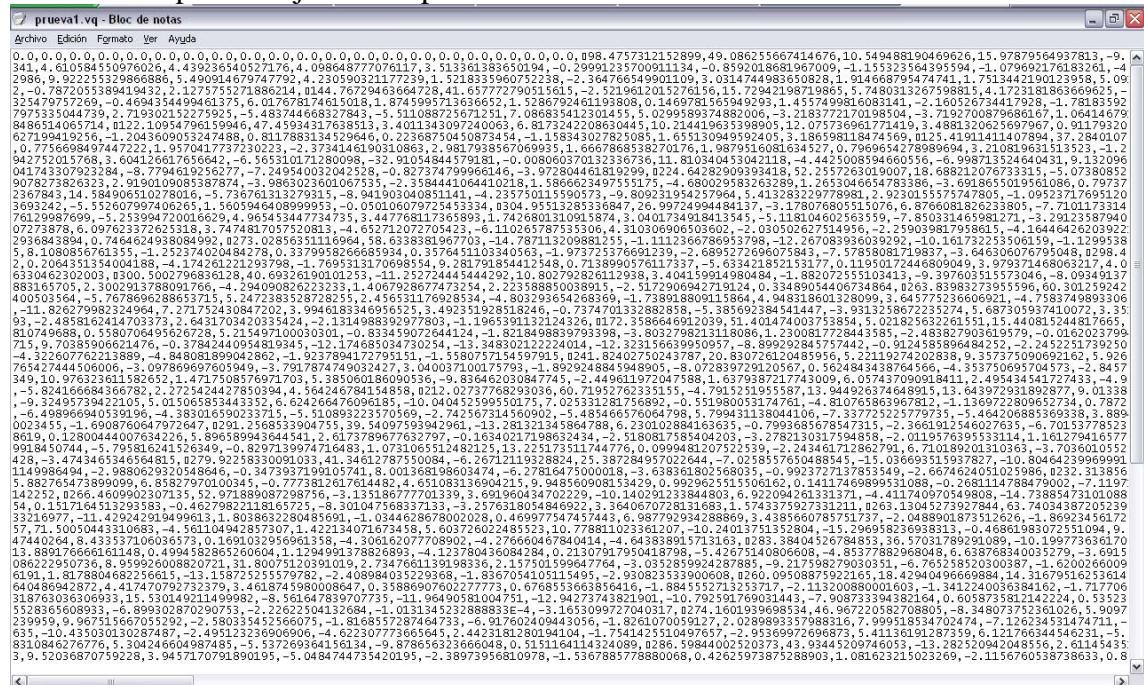


Figura 35. Funcionamiento del servidor Apache Tomcat en HTTPS

Cuando se envíe la voz esta viajará por un canal cifrado.

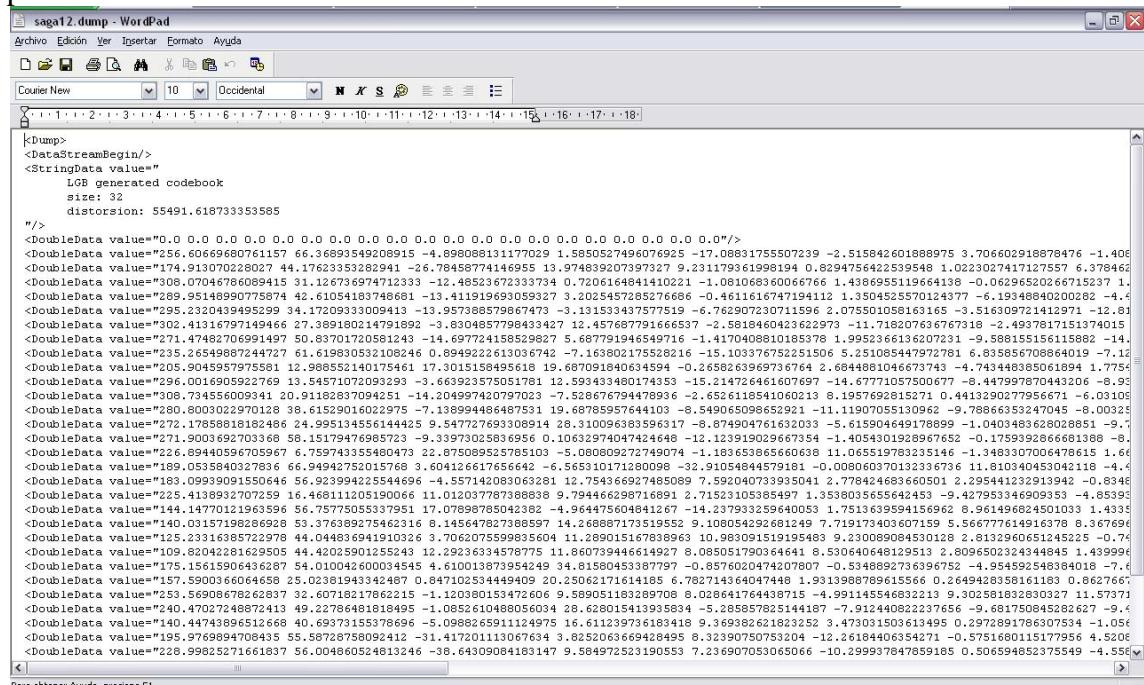
## 5.4 CAPA DE INTEGRACIÓN

Se realizaron pruebas para ver si se generaban los vectores acústicos, y después de generados se guardaban en archivos de texto plano: a continuación un archivo generado con éxito después de ejecutar el proceso de la extracción de los MFCC:



*Figura 36. Generación de archivo de vectores de voz*

Posteriormente se probó que se generara correctamente el libro de códigos en un archivo de salida después de que la prueba fue satisfactoria se obtuvo un archivo parecido a este:



*Figura 37. Generación del códec*

## 5.5 CAPA ONTOLÓGICA

La creación de la Ontología de usuario necesaria se hizo con la interfaz de Protégé. En primer lugar se selecciona crear new OWL ontology:

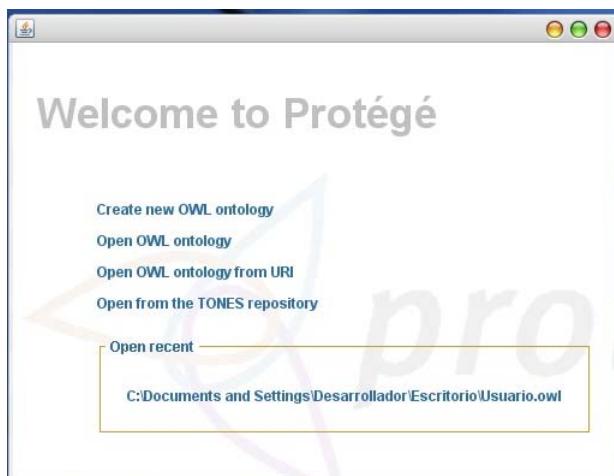


Figura 38. Ventana inicial Protégé

Se generó el esquema ontológico el cual sirvió para la creación de la identidad virtual de los usuarios.

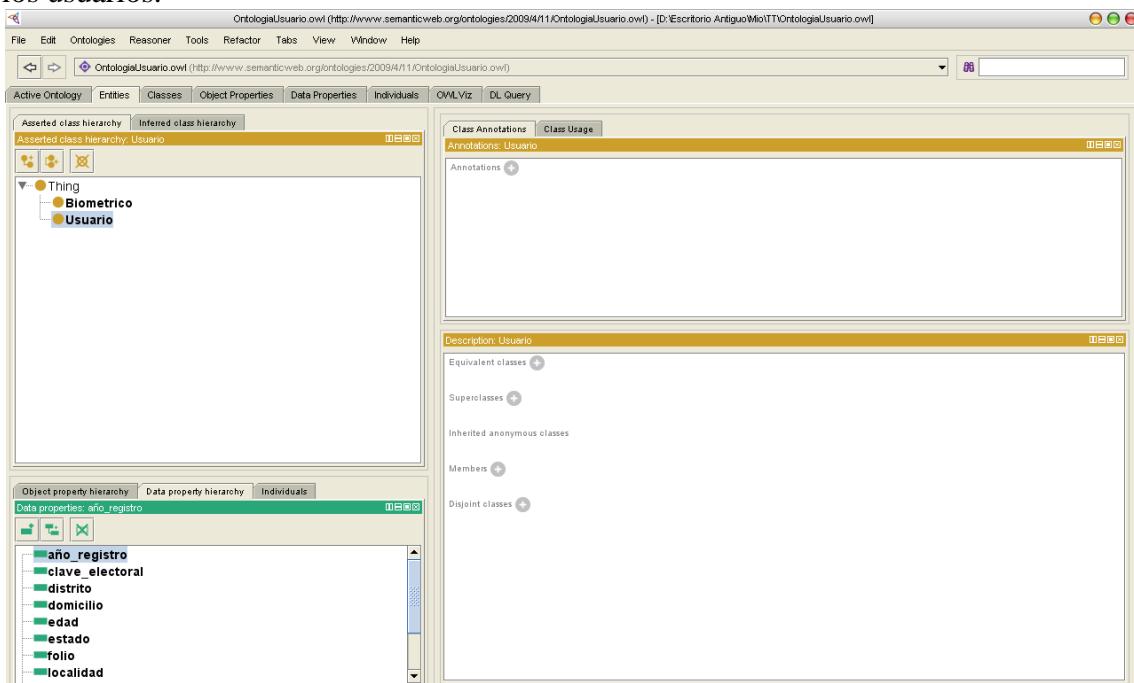


Figura 39. Generación de esquema ontológico con Protégé

El archivo .OWL que se genera con el proceso anterior es el siguiente:

```
//ARCHIVO AutentificacionUsuario.owl formato RDF/XML
<?xml version="1.0"?>
```

```
<!DOCTYPE rdf:RDF [
```

```

<!ENTITY owl "http://www.w3.org/2002/07/owl#" >
<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
<!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
<!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
<!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
<!ENTITY AutentificacionUsuarios
"http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsu
arios.owl#" >
]>

```

```

<rdf:RDF
xmlns="http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/Autentificac
ionUsuarios.owl#"

xml:base="http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/Autentifi
cationUsuarios.owl"

xmlns:AutentificacionUsuarios="http://www.arquitectura_autentificacion.com/ontologi
es/2009/4/25/AutentificacionUsuarios.owl#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
<owl:Ontology rdf:about="" />

```

```

<!--
///////////////
// Object Properties
//
/////////////
-->

```

```

<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#usuario_biometrico -->

<owl:ObjectProperty rdf:about="#usuario_biometrico">
 <rdf:type rdf:resource="&owl;FunctionalProperty"/>
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="#Usuario"/>
</owl:ObjectProperty>

```

```
<!--
///////////
// Data properties
//
/////////
-->
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#anio_registro -->
```

```
<owl:DatatypeProperty rdf:about="#anio_registro">
 <rdfls:domain rdf:resource="#Usuario"/>
 <rdfls:range rdf:resource="&xsd:int"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#apmat -->
```

```
<owl:DatatypeProperty rdf:about="#apmat">
 <rdfls:domain rdf:resource="#Usuario"/>
 <rdfls:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#appat -->
```

```
<owl:DatatypeProperty rdf:about="#appat">
 <rdfls:domain rdf:resource="#Usuario"/>
 <rdfls:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#clave -->
```

```
<owl:DatatypeProperty rdf:about="#clave">
 <rdfs:domain rdf:resource="#Usuario"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#distorsion -->
```

```
<owl:DatatypeProperty rdf:about="#distorsion">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#distrito -->
```

```
<owl:DatatypeProperty rdf:about="#distrito">
 <rdfs:domain rdf:resource="#Usuario"/>
 <rdfs:range rdf:resource="&xsd:int"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#domicilio -->
```

```
<owl:DatatypeProperty rdf:about="#domicilio">
 <rdfs:domain rdf:resource="#Usuario"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#edad -->
```

```
<owl:DatatypeProperty rdf:about="#edad">
 <rdfs:domain rdf:resource="#Usuario"/>
 <rdfs:range rdf:resource="&xsd:int"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#estado -->
```

```
<owl:DatatypeProperty rdf:about="#estado">
 <rdfs:domain rdf:resource="#Usuario"/>
 <rdfs:range rdf:resource="&xsd:int"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#fase -->
```

```
<owl:DatatypeProperty rdf:about="#fase">
 <rdfs:domain rdf:resource="#Usuario"/>
 <rdfs:range rdf:resource="&xsd:int"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#folio -->
```

```
<owl:DatatypeProperty rdf:about="#folio">
 <rdfs:domain rdf:resource="#Usuario"/>
 <rdfs:range rdf:resource="&xsd:int"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#id_usuario -->
```

```
<owl:DatatypeProperty rdf:about="#id_usuario">
 <rdfs:domain rdf:resource="#Usuario"/>
 <rdfs:range rdf:resource="&xsd:ID"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#localidad -->
```

```
<owl:DatatypeProperty rdf:about="#localidad">
```

```
<rdfs:domain rdf:resource="#Usuario"/>
<rdfs:range rdf:resource="&xsd:int"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#municipio -->
```

```
<owl:DatatypeProperty rdf:about="#municipio">
 <rdfs:domain rdf:resource="#Usuario"/>
 <rdfs:range rdf:resource="&xsd:int"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#nombre -->
```

```
<owl:DatatypeProperty rdf:about="#nombre">
 <rdfs:domain rdf:resource="#Usuario"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#seccion -->
```

```
<owl:DatatypeProperty rdf:about="#seccion">
 <rdfs:domain rdf:resource="#Usuario"/>
 <rdfs:range rdf:resource="&xsd:int"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#sexo -->
```

```
<owl:DatatypeProperty rdf:about="#sexo">
 <rdfs:domain rdf:resource="#Usuario"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v1 -->
```

```
<owl:DatatypeProperty rdf:about="#v1">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v10 -->
```

```
<owl:DatatypeProperty rdf:about="#v10">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v11 -->
```

```
<owl:DatatypeProperty rdf:about="#v11">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v12 -->
```

```
<owl:DatatypeProperty rdf:about="#v12">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v13 -->
```

```
<owl:DatatypeProperty rdf:about="#v13">
 <rdfs:domain rdf:resource="#Biometrico"/>
```

```
<rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v14 -->
```

```
<owl:DatatypeProperty rdf:about="#v14">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v15 -->
```

```
<owl:DatatypeProperty rdf:about="#v15">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v16 -->
```

```
<owl:DatatypeProperty rdf:about="#v16">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v17 -->
```

```
<owl:DatatypeProperty rdf:about="#v17">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v18 -->
```

```
<owl:DatatypeProperty rdf:about="#v18">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v19 -->
```

```
<owl:DatatypeProperty rdf:about="#v19">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v2 -->
```

```
<owl:DatatypeProperty rdf:about="#v2">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v20 -->
```

```
<owl:DatatypeProperty rdf:about="#v20">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v21 -->
```

```
<owl:DatatypeProperty rdf:about="#v21">
 <rdfs:domain rdf:resource="#Biometrico"/>
```

```
<rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v22 -->
```

```
<owl:DatatypeProperty rdf:about="#v22">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v23 -->
```

```
<owl:DatatypeProperty rdf:about="#v23">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v24 -->
```

```
<owl:DatatypeProperty rdf:about="#v24">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v25 -->
```

```
<owl:DatatypeProperty rdf:about="#v25">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v26 -->
```

```
<owl:DatatypeProperty rdf:about="#v26">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v27 -->
```

```
<owl:DatatypeProperty rdf:about="#v27">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v28 -->
```

```
<owl:DatatypeProperty rdf:about="#v28">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v29 -->
```

```
<owl:DatatypeProperty rdf:about="#v29">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v3 -->
```

```
<owl:DatatypeProperty rdf:about="#v3">
 <rdfs:domain rdf:resource="#Biometrico"/>
```

```
<rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v30 -->
```

```
<owl:DatatypeProperty rdf:about="#v30">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v31 -->
```

```
<owl:DatatypeProperty rdf:about="#v31">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v32 -->
```

```
<owl:DatatypeProperty rdf:about="#v32">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v4 -->
```

```
<owl:DatatypeProperty rdf:about="#v4">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v5 -->
```

```
<owl:DatatypeProperty rdf:about="#v5">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v6 -->
```

```
<owl:DatatypeProperty rdf:about="#v6">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v7 -->
```

```
<owl:DatatypeProperty rdf:about="#v7">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v8 -->
```

```
<owl:DatatypeProperty rdf:about="#v8">
 <rdfs:domain rdf:resource="#Biometrico"/>
 <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsua
rios.owl#v9 -->
```

```
<owl:DatatypeProperty rdf:about="#v9">
 <rdfs:domain rdf:resource="#Biometrico"/>
```

```
<rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
```

```
<!--
///////////
//
// Classes
//
///////////
-->
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#Biometrico -->
```

```
<owl:Class rdf:about="#Biometrico">
 <rdfs:subClassOf rdf:resource="&owl;Thing"/>
</owl:Class>
```

```
<!--
http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#Usuario -->
```

```
<owl:Class rdf:about="#Usuario">
 <rdfs:subClassOf rdf:resource="&owl;Thing"/>
</owl:Class>
```

```
<!-- http://www.w3.org/2002/07/owl#Thing -->
```

```
<owl:Class rdf:about="&owl;Thing"/>
</rdf:RDF>
```

```
<!-- Generated by the OWL API (version 2.2.1.1090) http://owlapi.sourceforge.net -->
```

```
//ARCHIVO AutentificacionUsuario.owl format OWL/XML
<?xml version="1.0"?>
```

```

<!DOCTYPE owl2xml:Ontology [
 <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
 <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
 <!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
 <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
 <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
 <!ENTITY
 "http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#" >
]>

<owl2xml:Ontology
 xmlns="http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#"
 xml:base="http://www.w3.org/2006/12/owl2-xml#"

 xmlns:AutentificacionUsuarios="http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#"

 xmlns="http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"

 owl2xml:URI="http://www.arquitectura_autentificacion.com/ontologies/2009/4/25/AutentificacionUsuarios.owl">
 <owl2xml:SubClassOf>
 <owl2xml:Class owl2xml:URI="&Biometrico"/>
 <owl2xml:Class owl2xml:URI="&owl;Thing"/>
 </owl2xml:SubClassOf>
 <owl2xml:SubClassOf>
 <owl2xml:Class owl2xml:URI="&Usuario"/>
 <owl2xml:Class owl2xml:URI="&owl;Thing"/>
 </owl2xml:SubClassOf>
 <owl2xml:FunctionalObjectProperty>
 <owl2xml:ObjectProperty owl2xml:URI="&usuario_biometrico"/>
 </owl2xml:FunctionalObjectProperty>
 <owl2xml:ObjectPropertyDomain>
 <owl2xml:ObjectProperty owl2xml:URI="&usuario_biometrico"/>
 <owl2xml:Class owl2xml:URI="&Biometrico"/>
 </owl2xml:ObjectPropertyDomain>
 <owl2xml:ObjectPropertyRange>
 <owl2xml:ObjectProperty owl2xml:URI="&usuario_biometrico"/>
 <owl2xml:Class owl2xml:URI="&Usuario"/>
 </owl2xml:ObjectPropertyRange>
 <owl2xml:DataPropertyDomain>

```

```

<owl2xml:DataProperty owl2xml:URI="&;anio_registro"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;anio_registro"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:int"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;apmat"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;apmat"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;appat"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;appat"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;clave"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;clave"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;distorsion"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;distorsion"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;distrito"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;distrito"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:int"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;domicilio"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>

```

```

</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;domicilio"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;edad"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;edad"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:int"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;estado"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;estado"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:int"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;fase"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;fase"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:int"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;folio"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;folio"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:int"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;id_usuario"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;id_usuario"/>
 <owl2xml:Datatype owl2xml:URI="&xsd;ID"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;localidad"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>

```

```

<owl2xml:DataProperty owl2xml:URI="&;localidad"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:int"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;municipio"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;municipio"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:int"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;nombre"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;nombre"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;seccion"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;seccion"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:int"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;sexo"/>
 <owl2xml:Class owl2xml:URI="&;Usuario"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;sexo"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;v1"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;v1"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;v10"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;v10"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>

```



```

<owl2xml:DataProperty owl2xml:URI="&;v17"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;v17"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;v18"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;v18"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;v19"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;v19"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;v2"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;v2"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;v20"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;v20"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;v21"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;v21"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;v22"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>

```

```

</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&v22"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&v23"/>
 <owl2xml:Class owl2xml:URI="&Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&v23"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&v24"/>
 <owl2xml:Class owl2xml:URI="&Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&v24"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&v25"/>
 <owl2xml:Class owl2xml:URI="&Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&v25"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&v26"/>
 <owl2xml:Class owl2xml:URI="&Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&v26"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&v27"/>
 <owl2xml:Class owl2xml:URI="&Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&v27"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&v28"/>
 <owl2xml:Class owl2xml:URI="&Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>

```

```

<owl2xml:DataProperty owl2xml:URI="&;v28"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;v29"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;v29"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;v3"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;v3"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;v30"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;v30"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;v31"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;v31"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;v32"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;v32"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&;v4"/>
 <owl2xml:Class owl2xml:URI="&;Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&;v4"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>

```

```

</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&v5"/>
 <owl2xml:Class owl2xml:URI="&Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&v5"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&v6"/>
 <owl2xml:Class owl2xml:URI="&Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&v6"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&v7"/>
 <owl2xml:Class owl2xml:URI="&Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&v7"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&v8"/>
 <owl2xml:Class owl2xml:URI="&Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&v8"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
<owl2xml:DataPropertyDomain>
 <owl2xml:DataProperty owl2xml:URI="&v9"/>
 <owl2xml:Class owl2xml:URI="&Biometrico"/>
</owl2xml:DataPropertyDomain>
<owl2xml:DataPropertyRange>
 <owl2xml:DataProperty owl2xml:URI="&v9"/>
 <owl2xml:Datatype owl2xml:URI="&xsd:string"/>
</owl2xml:DataPropertyRange>
</owl2xml:Ontology>

```

<!-- Generated by the OWL API (version 2.2.1.1090) http://owlapi.sourceforge.net -->

## 5.6 CONCLUSIONES

---

La autenticación dentro de cualquier sistema de información es parte vital para las organizaciones que requieran dar a sus usuarios la seguridad de que su información será respetada. El presente trabajo ha dado pauta a sacar las siguientes conclusiones:

La autenticación dentro de cualquier sistema de información es parte vital para las organizaciones que requieran dar a sus usuarios la seguridad de que su información será respetada. El presente trabajo ha dado pauta a sacar las siguientes conclusiones:

- La autenticación de usuarios requiere un esquema bien definido para que esta sea efectiva, no basta tener datos biométricos de un usuario, hay que saber administrarlos.
- Es importante que un usuario sea capaz de accesar a su información o servicios determinados desde cualquier parte en la que se encuentre y es por ello necesario hacer uso de la tecnología móvil existente.
- La compartición de la información a través de Internet va evolucionando y por ello es necesario hacer uso de los nuevos esquemas para generar contenidos inteligentes como lo son las ontologías.
- El proceso de la extracción de características únicas de la voz es un proceso complejo pero si se sabe llevar de manera adecuada llega a ser uno de los medios más robustos para poder autenticar a las personas, pudiendo concluir que los MFCC son las características más fiables al momento de autenticar a alguien.
- El proceso de compresión de vectores y de cuantificación vectorial seguido en el presente proyecto LGB en nuestro caso es de los mejores métodos para cuantificación vectorial existentes aunque el poder de cómputo usado para el cálculo del código de libros en ocasiones es elevado.
- El cálculo de los rangos y umbrales en un proceso como el que se llevó a cabo suele ser un factor difícil de calcular y en algunos casos la incorrecta elección del mismo puede elevar de manera significativa el número de falsos positivos o negativos del sistema, por lo que es un punto importante a considerar.
- El envío de información debe ser enviado por un canal cifrado que permita garantizar que la información llegue de manera segura a su destino.
- El navegador permite dar soporte a aplicaciones web Multimodal, soportando XHTML+Voice, lo que da como resultado una aplicación más completa con interacción visual y vocal.

## 5.7 TRABAJO A FUTURO

---

Debido a la extensibilidad de la arquitectura para poder trabajar con cualquier biométrico sea voz, huella dactilar, retina de los ojos, etc., resultara fácil implementar la arquitectura en cualquier sistema de autenticación deseado.

Con los avances tecnológicos se podrá mejorar la eficiencia de cada una de las capas, VoiceXML 3.0 tiene como objetivo a futuro identificar las características biométricas de voz sin necesidad de algoritmos complejos.

- En la capa de internet se podrá utilizar cualquier algoritmo de cifrado que permita mejorar la seguridad en el envío de datos.
- La arquitectura presentada podrá ser empleada en entidades bancarias, sistemas seguros de información, etc. a fin de proteger la información de los usuarios.
- El presente esquema ontológico es susceptible de ser aumentado con el fin de abarcar mas biométricos y elementos portátiles.
- Se puede implementar la arquitectura modificando cualquiera de sus elementos, como lo son otros navegadores, otros protocolos de seguridad y nuevas versiones de VoiceXML.

## GLOSARIO

---

**Biometría:** es el estudio de métodos automáticos para el reconocimiento único de humanos basados en uno o más rasgos conductuales o físicos intrínsecos. El término se deriva de las palabras griegas "bios" de vida y "metron" de medida.

**Biometría informática:** es la aplicación de técnicas matemáticas y estadísticas sobre los rasgos físicos o de conducta de un individuo, para “verificar” identidades o para “identificar” individuos.

**Autenticar (Autenticar):** es el acto de establecimiento o confirmación de algo (o alguien) como auténtico, es decir que reclama hecho por, o sobre la cosa son verdadero. La autenticación de un objeto puede significar (pensar) la confirmación de su procedencia, mientras que la autenticación de una persona a menudo consiste en verificar su identidad. La autenticación depende de uno o varios factores de autenticación.

**Ontología:** El término ontología en informática hace referencia a la formulación de un exhaustivo y riguroso esquema conceptual dentro de un dominio dado, con la finalidad de facilitar la comunicación y la compartición de la información entre diferentes sistemas. Aunque toma su nombre por analogía, ésta es la diferencia con el significado filosófico de la palabra ontología.

**Seguridad:** Podemos entender como seguridad un estado de cualquier sistema (informático o no) que nos indica que ese sistema está libre de peligro, daño o riesgo. Se entiende como peligro o daño todo aquello que pueda afectar su funcionamiento directo o los resultados que se obtienen del mismo. Para la mayoría de los expertos el concepto de seguridad en la informática es utópico porque no existe un sistema 100% seguro. Para que un sistema se pueda definir como seguro debe tener estas cuatro características:

- Integridad: La información sólo puede ser modificada por quien está autorizado.
- Confidencialidad: La información sólo debe ser elegible para los autorizados.
- Disponibilidad: Debe estar disponible cuando se necesita.
- Irrefutabilidad: (No-Rechazo o No Repudio) Que no se pueda negar la autoría.

## REFERENCIAS

---

- [1] The Semantic Web, Tim Berners-Lee, James Hendler and Ora Lassila, Mayo de 2001, disponible en, <http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21>
- [2] ¿Qué es la Web Semántica?, inkel, Septiembre de 2005, disponible en, <http://f14Web.com.ar/inkel/que-es-la-Web-semantica>
- [3] La Web Semántica Claudio Gutiérrez, Octubre de 2001, disponible en <http://www.dcc.uchile.cl/~cgutierrez/Websemantica/Websemantica.pdf>
- [4] The Semantic Web: An Introduction, Sean B. Palmer, Septiembre de 2001, disponible en <http://infomesh.net/2001/swintro/>
- [5] W3C Semantic Web Activity, Marja-Riitta Koivunen and Eric Millar, Noviembre de 2001, disponible en <http://www.w3.org/2001/12/semWeb-fin/w3csw>
- [6] Biblioteca Digital y Web Semántica, Carolina García Cataño y David Arroyo Menéndez, visitada en Octubre de 2005, disponible en, <http://www.sindominio.net/biblioWeb/telematica/bibdigWebsem.html>
- [7] SWRL: A Semantic Web Rule Language Combining OWL and RuleML, dalm.org, Noviembre de 2003, disponible en, <http://www.daml.org/2003/11/swrl/>
- [8] Jess the Rule Engine for the JavaTM Platform, jessrules, Noviembre de 2005, disponible en, <http://www.jessrules.com/jess/index.shtml>
- [9] The project HOPS: Enabling an Intelligent Natural LanguageBased Hub for the Deployment of Advanced SemanticallyEnriched Multi-channel Mass-scale Online Public Services, Marta Gatius, Meritxell González Centro de investigación TALP Departamento de Lenguajes y sistemas Informáticos Universitat Politècnica de Catalunya disponible en <http://www.lsi.upc.es/~gatius/proyectorhops.pdf>.
- [10] Enabling and Intelligent Natural Language Based Hub for the Deployment of Advanced Semantically Enriched Multi-channel Mass-scale Online Public Services, Oscar Corcho, School of Computer Science University of Manchester disponible en <http://www.bcn.es/hops/>
- [11] New technology gives Web a voice, Wylie Wong, Julio de 2001, disponible en, <http://news.com.com/2100-1033-270242.html>
- [12] VoiceXML and the Voice-Driven Internet, David Houlding, Abril de 2001, disponible en, <http://www.ddj.com/documents/s=868/ddj0104g/>
- [13] What is BeVocal Café?, visitado en Octubre de 2005, disponible en, <http://cafe.bevocal.com/>
- [14] Loquendo C@fé, visitado en octubre de 2005, disponible en, <http://www.loquendo.com/es/index.htm>
- [15] The Semantic Browser: Improving the User Experience, Mark Birbeck, x-port.net, and Steven Pemberton, disponible en <http://www.w3.org/2005/Talks/05-steven-www2005/>
- [16] How to Make a Semantic Web Browser, Dennis Quan, David R. Karger, disponible en <http://www.www2004.org/proceedings/docs/1p255.pdf>.
- [17] Piggy Bank, What is this?, visitado en septiembre de 2005, disponible en, <http://simile.mit.edu/piggy-bank/>
- [18] Magpie – towards a semantic Web browser; Martin Dzbor, John Domingue, and Enrico Motta, disponible en <http://kmi.open.ac.uk/people/domingue/papers/magpie-iswc-03.pdf>. jornadas de trabajo internacionales sobre tecnologías de voz, lenguaje natural y Web semántica, visitado en octubre de 2005, disponible en, [http://www.isoco.com/noticias/05\\_01\\_27.html](http://www.isoco.com/noticias/05_01_27.html)

- [19] Diseño de pruebas para la evaluación de habla sintetizada en español y su aplicación a un sistema de conversión de texto a habla, Lourdes Aguilar, Josep M. Fernández, Juan M. Garrido y Joaquim Llisterri, Agosto de 2003, disponible en, [http://liceu.uab.es/~joaquim/publicaciones/cordoba\\_94.html](http://liceu.uab.es/~joaquim/publicaciones/cordoba_94.html)
- [20] Desarrollo de aplicaciones vocales en VoiceXML, Luis Ángel García Muñoz, Luis Rodero Merino, visitado en Septiembre de 2005, disponible en, <http://verbo.dcs.fi.uva.es/~cesargf/Personal/prototipo/proyVXMLlagm.pdf>  
<http://www.tid.es/presencia/publicaciones/comsid/esp/articulos/vol23/habla/habla.html>
- [21] Acerca la Web Semántica a Europa, REYNOLDS, Dave, octubre de 2005, disponible en, [http://icadc.cordis.lu/fep-cgi/srchidadb?CALLER=OFFR\\_O\\_SCIE\\_ES&ACTION=D&RCN=2212&DOC=6&CAT=OFFR&QUERY=1](http://icadc.cordis.lu/fep-cgi/srchidadb?CALLER=OFFR_O_SCIE_ES&ACTION=D&RCN=2212&DOC=6&CAT=OFFR&QUERY=1)
- [22] Autenticación de usuarios, Nikos Drakos, CBLU, University of Leeds, disponible en, <http://www.ibiblio.org/pub/Linux/docs/LuCaS/Manuales-LuCAS/doc-unixsec/unixsec-html/node110.html>
- [23] Andrew Tanenbaum, *Operating Systems: Design and Implementation*, Prentice Hall, 1991
- [24] Eric Guerrino, Mike Kahn, and Ellen Kapito, User authentication and encryption overview, 1997.
- [25] Gruber T., “Toward Principles for the Design of Ontologies Used for Knowledge Sharing” Technical Report KSL-93-04, Knowledge Systems Laboratory, Stanford University, CA, 1993.
- [26] Studer S, Benjamins R., and Fensel D., “Knowledge Engineering: Principles and Methods”, Data and Knowledge Engineering, 25, 161-197, 1998.