Instituto Tecnológico de Orizaba
División de Estudios de Posgrado e Investigación
Maestría en Sistemas Computacionales

Desarrollo de una aplicación para realizar el registro de notas médicas en el expediente clínico electrónico de un sanatorio mediante el reconocimiento de voz.

TESIS

PRESENTA:

ISC Mario Ezequiel Villalobos Gallegos

PARA OBTENER EL GRADO DE MAESTRO EN:

SISTEMAS COMPUTACIONALES

DIRECTOR DE TESIS:

MCE Beatriz Alejandra Olivares Zepahua

Orizaba, Veracruz, México Mayo de 2016

Autorización de impresión

Revisión de trabajo escrito

Agradecimientos

A Dios por brindarme la oportunidad de vivir en estos tiempos donde el acceso a la educación y a la información es más fácil.

De aquí en adelante, el nombre de las personas a las que les dedico este trabajo van en orden alfabético, pues para mi todos ellos son igual de importantes.

Araceli y Ezequiel, mis padres por todo el apoyo que me han dado a lo largo de mi vida.

Monserrat, mi esposa por llenarme el corazón de amor y el estómago de deliciosas comidas todos los días, además de tener la paciencia cuando me hablaba y no le hacía caso pues me encontraba escribiendo este documento o codificando la aplicación.

Romina, mi hija por ser el motor de mi vida y el principal motivo para seguir superándome continuamente tanto de forma académica como personal.

A todas las demás personas que conscientemente o no han colaborado con un granito de arena para la culminación de este trabajo.

De igual forma agradezco al CONACYT por la beca otorgada para la realización de la maestría, así como al Tecnológico Nacional de México por las facilidades otorgadas para el desarrollo de la misma.

MUCHAS GRACIAS.

Contenido

Lista de figuras.

Lista de tablas.

Índice de listados.

Resumen.

Se sabe que no es del agrado del personal del sector salud la actualización del expediente clínico y como una de las actividades de esta labor se encuentra la redacción de notas médicas, pues consideran que pierden tiempo que podrían usar atendiendo a más pacientes. Sin embargo, la nota médica es una de las características esenciales que debe estar incluida en el expediente de cada paciente. Además, está comprobado que el uso del reconocimiento de voz disminuye el tiempo de entrega de reportes médicos.

En este proyecto se desarrolló una aplicación que realiza el registro de notas médicas en un expediente clínico electrónico, el cual se encuentra implementado dentro de un sanatorio, haciendo uso del reconocimiento de voz y el vocabulario específico de una especialidad médica, todo esto apoyándose en el manejo de vocabularios controlados y ontologías.

Para incluir el uso de reconocimiento de voz en la aplicación se hizo uso de la biblioteca *CMU-Sphinx*, la cual permite configurarse según las necesidades de la solución. Se anexaron más de 500 términos del área médica en el diccionario de conceptos permitidos, además, el uso de gramáticas dio como resultado un 88% de precisión para un usuario con el cual el software fue entrenado, mientras que, para otro usuario, con el cual el sistema no se entrenó, se alcanzó hasta un 78.4% de efectividad.

Por otro lado, se utilizó una ontología para asegurar la interoperabilidad con diversos gestores de bases de datos, para este fin se hace uso de la biblioteca *Jena.* Se realizaron pruebas de conexión con *PostgresQL* y *MySQL*, teniendo resultados satisfactorios para ambos casos. La codificación de acceso a base de datos en combinación con la ontología, permite añadir más gestores incluyendo solo la biblioteca al proyecto e indicando la información necesaria para la conexión en la ontología.

Abstract.

Is well known that the update of the health record is not liked by the personal working on that sector, and as one of the activities in the update of health records, we can find the writing of medical notes which is considered as a time consuming task and this time can be used in the attention of another patient. Nevertheless, the medical note is a fundamental characteristic which has to be included in every patient record. Besides, has been proved that the use of voice recognition decreases the delivery time of medical reports.

In this project, an application was developed that can register medical notes in an electronic health record which is implemented in a hospital, the application uses voice recognition and the specific vocabulary from a medical field, all of this thanks to the management of controlled vocabularies and ontologies.

To make possible adding voice recognition to the desktop app, we use the CMU-Sphinx library which can be configured according to the solution's needs. We added more than 500 terms from the medical sector in the allowed dictionary of concepts. Moreover, the use of grammars resulted in an 88% of precision with an user in which the software has been trained, on the other hand, for an user in which the system has not been trained, we reach an 78.4% of accuracy.

Furthermore, an ontology was used to ensure interoperability with several database managers, for this the Jena library was used. We performed connection tests with PostgresQL and MySQL, having satisfactory results in both cases. The access code to the database in combination with the ontology has made possible add more managers, only by adding his library to the project and pointing out the necessary information for the connection on the ontology.

Introducción.

El uso de aplicaciones que mediante el reconocimiento de voz permitan la introducción de texto a documentos electrónicos es una alternativa innovadora que se ha puesto en práctica en los últimos años para agilizar el trabajo en distintas áreas de la ciencia, sobre todo en la medicina, en donde ha tenido mayor auge es en las áreas de odontología y radiología pues es necesario el uso de ambas manos por parte de los doctores para realizar sus actividades correctamente mientras documentan sus descubrimientos en una computadora.

El reconocimiento de voz tiene especial importancia ya que en algunos casos contar con una mayor rapidez en la captura de notas médicas da como resultado una mejor atención y tratamiento de los pacientes en situaciones donde la pronta obtención de información es crucial para unos buenos cuidados por parte del personal en las instituciones médicas. Por otro lado, las ontologías son otra de las tecnologías en auge y ampliamente usadas para asegurar la interoperabilidad entre sistemas de expediente clínico electrónico.

Es por lo anterior que este trabajo de tesis presenta una propuesta para solucionar la problemática que aqueja a un sanatorio de la región de Orizaba, en donde el uso de notas médicas no se cumple a cabalidad debido a la falta de tiempo y exceso de trabajo, aunque dichos documentos son una pieza fundamental en los expedientes de los pacientes según lo indica la normatividad vigente en materia de salud y apoyándose en el uso una ontología para el control del vocabulario desarrollado.

El objetivo fue implementar el uso de una aplicación de escritorio que mediante el reconocimiento de voz permita registrar notas médicas en el expediente clínico electrónico con el que cuenta actualmente la institución. Para ello se procedió a analizar las normas oficiales vigentes en México con respecto al expediente clínico y a los sistemas de información de registro electrónico de salud, además de buscar ontologías en el área de pediatría.

Se analizaron las opciones de herramientas gratuitas para desarrollar la solución propuesta y una vez que se encontró la combinación de tecnologías adecuadas para tal fin se procedió a desarrollar el vocabulario de la especialidad médica y el modelo del lenguaje respectivo para realizar el reconocimiento de voz. Además, se realizó la arquitectura de la aplicación tomando en cuenta el uso de ontologías.

Por último, se desarrolló la aplicación y se realizaron pruebas de reconocimiento de voz con los médicos de la institución obteniendo resultados satisfactorios.

La estructura de la tesis es la siguiente:

En el capítulo 1 se presentan los términos y sus definiciones que son necesarios para comprender la información contenida en los capítulos siguientes, así como el planteamiento de la problemática, el objetivo general y los objetivos específicos necesarios para cumplirlo, además de la justificación correspondiente.

En el capítulo 2 se encuentra el estado de la práctica, la cual incluye una serie de trabajos relacionados en el tema del reconocimiento de voz como apoyo a las labores diarias dentro del campo de la medicina, así como el uso de ontologías para la comunicación entre aplicaciones de gestión de la salud. También se hace un análisis comparativo de los trabajos presentados y se muestra la propuesta de solución para la problemática.

En el capítulo 3 se detalla el proceso que se siguió para la construcción de la aplicación; en primer lugar, indicando la lista del producto y lista de pendientes, que son artefactos usados en la metodología *Scrum.* Se continúa con la arquitectura de la aplicación, la explicación de la manera de entrenar el modelo acústico para el sistema de RV y pruebas de reconocimiento tanto en línea de comando como en vivo. Seguido de la codificación del módulo de RV, la conceptualización y la construcción de la ontología que permite la interoperabilidad entre diversos gestores de bases de datos, así como la codificación del módulo para tener acceso a ella. Por último, se indica el módulo de acceso a bases de datos.

En el capítulo 4 se muestran los resultados obtenidos con la implementación de la aplicación, seguido de los resultados con pruebas controladas, además se incluye el caso de estudio aplicando el reconocimiento de voz con pediatras del sanatorio.

En el capítulo 5 se detallan las conclusiones de este proyecto y las recomendaciones para trabajos a futuro.

Capítulo 1 Antecedentes

En este capítulo, se dan a conocer los conceptos que se utilizan a lo largo de la tesis, se presenta el problema a resolver, los objetivos del proyecto, así como la justificación del mismo.

1.1 Marco teórico

En esta sección se definen los conceptos más importantes para la comprensión de este trabajo.

1.1.1 Ontología

En el contexto de las Ciencias de la Computación e Información, una ontología define un conjunto de primitivas de representación con las cuales es posible modelar un dominio del conocimiento. Las primitivas de representación son típicamente clases (o conjuntos), atributos (o propiedades), y relaciones (o las relaciones entre los miembros de la clase). Las definiciones de las primitivas de representación incluyen información acerca de su significado y las restricciones en su aplicación lógicamente consistentes [1].

Una ontología es el resultado de una formulación exhaustiva y rigurosa de la conceptualización de un dominio. Esta conceptualización a menudo se dice que es parcial, ya que es ilusorio creer que es posible captar la complejidad de la extensión de un dominio en este tipo de formalismos. Es necesario tener en cuenta también que el grado de formalización de una ontología varía de acuerdo al uso previsto para ella [2].

1.1.2 Expediente Clínico Electrónico.

Un expediente clínico electrónico (ECE) es una versión digital de los registros en papel de un paciente. Los ECE se actualizan en tiempo real y los registros están centrados en el paciente, lo que hace que la información se encuentre disponible al instante y de forma segura a los usuarios autorizados. Mientras un ECE contiene las historias clínicas y de tratamiento de los pacientes, un sistema de ECE está construido para ir más allá de los datos clínicos estándar recolectados en el consultorio y cuenta con la posibilidad de incluir una visión más amplia de la atención del paciente. Un ECE tiene las siguientes características:

- Contiene la historia médica de un paciente, diagnósticos, medicamentos, planes de tratamiento, fechas de vacunación, alergias, imágenes de radiología y laboratorio y resultados de pruebas.
- Permite el acceso a herramientas basadas en evidencia que los médicos utilizan para tomar decisiones sobre el cuidado de un paciente.
- Automatizan y agilizan el flujo de trabajo de los médicos.

Una de las características clave de un ECE es que la información de salud se crea y se gestiona por médicos autorizados a través de un formato digital capaz de compartirse con otros practicantes a través de más de una organización de atención médica. El ECE se construye para compartir información con otros proveedores y organizaciones de atención de la salud - tales como laboratorios, especialistas, centros de imágenes médicas, farmacias, servicios de emergencia - por lo que contiene información de todos los médicos involucrados en el cuidado del paciente [3].

1.1.3 Normas Oficiales Mexicanas (NOM)

Son regulaciones técnicas de carácter obligatorio. Regulan los productos, procesos o servicios, cuando éstos tengan la probabilidad de constituir un riesgo para las personas, animales y vegetales, así como el medio ambiente en general, entre otros

[4]. Para el caso de este proyecto aplican la NOM-004-SSA3-2012 correspondiente al expediente clínico y que cuenta con la información relativa a notas médicas, y la NOM-024-SSA3-2012 que se refiere a los Sistemas de Información de registro electrónico para la salud, la cual contiene los lineamientos para el intercambio de información entre dichos sistemas.

1.1.4 Notas médicas

Documento en donde se expresa el estado de salud del paciente además de sus datos de identificación personal como son: nombre completo, edad, sexo y, en su caso, número de cama o expediente, entre otra información que se considere pertinente según sea el caso, como por ejemplo para notas de progreso en psicología o notas de trabajo social. Se utilizan para el registro e incorporación de información en el expediente clínico durante acciones de diagnóstico, tratamiento y rehabilitación.

Además, contienen la fecha, hora y nombre completo de quien la elabora, así como la firma autógrafa, electrónica o digital. Deben expresarse en lenguaje técnico-médico, sin abreviaturas, con letra legible, sin enmendaduras ni tachaduras y conservarse en buen estado [5].

1.1.5 Reconocimiento de voz

El reconocimiento de voz (RV) utiliza probabilidades matemáticas de cuándo y con qué frecuencia aparecen las palabras en un contexto particular. El modelo acústico capta las propiedades acústicas del habla y ofrece las probabilidades de la señal acústica observada dada una secuencia de palabras hipotética. El modelo de lenguaje capta las propiedades lingüísticas del idioma y proporciona una probabilidad *a priori* de la secuencia de palabras, por lo general basado en conceptos estadísticos.

Para descomponer los sonidos en lenguaje escrito, el motor de voz toma la señal digitalizada desde el micrófono y la convierte de una señal basada en tiempo a un conjunto de frecuencias. A partir de estas frecuencias, la posición de los formantes del tracto vocal se extrae y representa como un conjunto de números. Estas cifras se comparan con una tabla de posiciones de formantes conocidos para fonemas escritos. La tabla de formantes se desarrolla previamente utilizando datos capturados de muchos cientos de muestras de hablantes nativos de una lengua específica y los resultados promedio de dichas muestras. Cuando se encuentra una coincidencia, el fonema correspondiente se trasmite a la siguiente etapa del proceso de reconocimiento, el análisis de la frase. En esta etapa, el motor de RV analiza las palabras reconocidas y estadísticamente las compara con otras palabras en el modelo de lenguaje utilizando un árbol de probabilidad. Se registra el número de veces que una palabra aparece en combinación con otros términos. El análisis calcula la probabilidad de que una palabra siga después de otra, o de que aparezca al principio o al final de una frase [6].

1.1.6 METHONTOLOGY

Este marco de referencia permite la construcción de ontologías a nivel de conocimiento e incluye la identificación del proceso de desarrollo de ontologías [7], el cual cuenta con un ciclo de vida que se basa en la construcción de prototipos evolutivos, además de técnicas particulares para llevar a cabo cada actividad. Las actividades están divididas en tres grupos y se realizan en diversos momentos durante el ciclo de vida de una ontología, éstas son:

1. Actividades de administración de proyecto.
2. Actividades orientadas al desarrollo.

La segunda parte está dividida en tres grupos de actividades:

a. Conceptualización. El primero de estos tres grupos describe 11 actividades de apoyo para la construcción de una ontología, las cuales hacen esta tarea más clara y fácil.
   i. Construcción del glosario de términos.
   ii. Construcción de la taxonomía de conceptos.
   iii. Construcción del diagrama de relaciones binarias.
   iv. Construcción del diccionario de conceptos.
   v. Descripción en detalle de las relaciones binarias.
   vi. Descripción de los atributos de instancia.
   vii. Descripción de los atributos de clase.
   viii. Descripción de las constantes.
   ix. Descripción de axiomas formales.
   x. Descripción de reglas.
   xi. Descripción de instancias.
b. Formalización.
c. Implementación.

3. Actividades de soporte.

1.1.7 Vocabulario controlado.

Es un conjunto de términos que proveen referencia común para sistemas de información vinculados [8]. También se define como una lista de palabras y frases para su uso en la recuperación de información, mostrando los términos que están autorizados para su uso y son válidos [9].

1.1.8 Reconocimiento independiente del hablante.

Al contrario de los sistemas dependientes del hablante, no requieren de un entrenamiento por parte del usuario, lo que los hace más atractivos para su uso. Algunos sistemas de reconocimiento de este tipo realizan una forma de adaptación a

la voz del usuario para mejorar su rendimiento. Se utiliza en situaciones donde el entrenamiento es difícil o imposible de realizar [10] [11].

1.1.9 Software médico de dictado.

Aplicación en la cual los profesionales médicos completan complejos informes sin pulsar ninguna tecla, simplemente usando la voz, mejorando su eficiencia en la redacción de informes. El facultativo tiene la capacidad de examinar, mientras dicta, otros informes, imágenes, entre otros elementos, sin necesidad de concentrarse en la computadora donde está redactando el informe, mejorando la productividad del médico y de sus herramientas de trabajo [12].

1.1.10 Ontología médica.

Se centra principalmente en la representación y la (re) organización de la terminología médica. Los médicos desarrollaron sus propios idiomas y léxicos especializados para ayudarles a almacenar y comunicar el conocimiento médico general y la información relacionada con el paciente de manera eficiente. Tales terminologías, optimizadas para el procesamiento humano, se caracterizan por una cantidad significativa de conocimiento implícito. Los sistemas de información médica, por otro lado, tienen que ser capaces de comunicarse entre sí conceptos médicos complejos y detallados (posiblemente expresado en diferentes idiomas) sin ambigüedades [13].

1.2 Planteamiento del problema.

Para los médicos, la actualización del expediente clínico, manual o electrónico, es una de las tareas menos agradables de su profesión, lo que aunado al corto tiempo disponible para tales tareas origina problemas en el manejo de los expedientes; esta situación se presenta tanto en el ámbito público como en el privado, en países

desarrollados y en países emergentes [14]. En varios ambientes se optó por esquemas de dictado donde el médico graba o dicta las notas y una persona (o software) trascribe la información para agregarla al expediente clínico.

Entre las herramientas actuales para RV y actualización de expedientes electrónicos, se encuentran casos como el de *Dragon Medical* de *Nuance*, que, una vez entrenado, permite dictar las notas o hallazgos médicos y actualizar con ello un sistema de expediente electrónico a un costo aproximado de $1,599 dólares [15] [16]. Existen otras soluciones semejantes como *CLIN1 Transcription*, *Digital Transcription Service* y *Emdat Clinical Documentation* entre otras [17], sin embargo todas adolecen de los mismos problemas para efectos de México: costo alto, reconocimiento de palabras en inglés únicamente (aunque existe la versión de *Dragon Medical* en español, está obtiene su mayor potencial de reconocimiento con el uso del micrófono *PowerMic* con un costo adicional de $424 dólares) y actualización de sistemas de expediente clínico electrónico muy específico.

Por lo anterior se propone desarrollar una aplicación que, mediante RV en español, sin entrenamiento, actualice un sistema de expediente clínico electrónico de acuerdo a la normativa vigente en México para una especialidad médica específica, auxiliándose en esta parte de ontologías existentes para el manejo de vocabularios y registros médicos [13] [18] [19] [20] [21].

1.3 Objetivo general y objetivos específicos.

En esta sección se presentan el objetivo general de la tesis, así como los objetivos específicos que permitirán llegar al objetivo general propuesto.

1.3.1 Objetivo general.

Desarrollar una aplicación que realice el registro de notas médicas en un expediente clínico electrónico, implementado dentro de un sanatorio, mediante el reconocimiento

de voz relacionándolo con un vocabulario específico apoyándose en vocabularios y ontologías.

1.3.2 Objetivos específicos.

- Identificar los elementos necesarios requeridos por la “Norma NOM-004-SSA3-2012, correspondiente al expediente clínico” y por la “Norma NOM-024-SSA3-2012, correspondiente a los Sistemas de información de registro electrónico para la salud” en sus apartados referentes a las Notas Médicas para identificar los requerimientos legales y de normatividad.
- Identificar una especialidad médica para capturar su vocabulario específico.
- Identificar ontologías existentes de la especialidad médica elegida, así como de expedientes clínicos e intercambio de información de salud para realizar las adecuaciones pertinentes de acuerdo a los requerimientos de las normas oficiales mexicanas.
- Desarrollar el vocabulario de la especialidad médica elegida a través de la reutilización de ontologías existentes y los términos recabados que no existen en ella para modelar el dominio de la especialidad.
- Analizar algoritmos para el reconocimiento de voz para determinar cuál de éstos tiene un mejor desempeño.
- Modelar la arquitectura para la aplicación considerando el uso de ontologías para la actualización del expediente clínico electrónico.
- Construir la aplicación que permita la actualización del expediente clínico electrónico mediante el reconocimiento de voz siguiendo la arquitectura definida.
- Realizar pruebas de la aplicación para verificar que reconoce correctamente la terminología médica y la refleja en el expediente clínico electrónico.

1.4 Justificación.

Se identificó que no es del agrado del personal del sector salud realizar la redacción de notas médicas, pues consideran que les consume un tiempo que sería mejor dedicar a la atención de más pacientes.

Sin embargo, la nota médica es una de las características que es indispensable incluir en el expediente clínico de todo paciente para llevar un seguimiento de los padecimientos por los que asiste a recibir atención en cualquier institución de salud.

Por otro lado, diversos estudios indican que el uso del RV para ingresar información en los registros de salud electrónicos permite disminuir el tiempo de entrega de reportes médicos, así como el número de éstos que se entregaron en un corto periodo de tiempo después de que se realizaron revisiones en los pacientes por parte del personal médico [22]. Esto permite una mejor atención en el caso de los padecimientos que requieren una respuesta rápida y en los que la demora en la entrega de resultados afecta la salud del paciente. También se facilita el ingreso de datos en actividades en las que el personal de salud debe tener ambas manos ocupadas como es el caso de los dentistas [23].

Para el caso de las ontologías, éstas se usan ampliamente en el área de medicina para el intercambio de información electrónica y la normatividad vigente marca que los estándares como el *eXtensible Markup Language* (XML) se usan para el intercambio de información. Cabe mencionar que el *Web Ontology Language* (OWL), lenguaje con el que se escriben las ontologías, es una extensión del estándar XML.

Es por todo lo anterior que se propone el desarrollo de una aplicación de RV que permita introducir datos en el registro de salud electrónico de un hospital mediante las notas médicas, haciendo uso de una ontología para el manejo del vocabulario médico y con el uso de dicha aplicación fomentar una mejor atención a los pacientes.

Capítulo 2 Estado de la práctica.

En este capítulo se da a conocer el estado de la práctica del proyecto de tesis, es decir algunos trabajos relacionados directa o indirectamente con el tema del proyecto de tesis.

2.1 Trabajos relacionados.

Se da inicio a esta sección con una serie de artículos en los que se desarrollaron aplicaciones que hacen uso del RV de manera exitosa como apoyo a las actividades médicas cotidianas.

Para fines prácticos se entenderán como sinónimos los términos “reconocimiento de voz”, “reconocimiento del habla”, “reconocimiento automático del habla” y se usará en acrónimo RV correspondiente al primer término para hacer referencia a los demás casos.

En [23] se presentó la evolución del desarrollo de los registros de salud electrónicos (RSE) en el centro *EuroMISE* ubicado en Praga, Republica Checa, el cual se nombró *MURDLite* y su componente gráfico para dentistas llamado *DentCross*. Los problemas con la inserción de datos en los RSE durante la revisión de pacientes con problemas dentales llevaron a una mayor investigación en el área del RV en la práctica médica. La investigación resultó en un software llamado *DentVoice*, el cual fue una aplicación exitosa del módulo RV y del componente *DentCross*. La unión del control por voz y la representación gráfica del arco dental hicieron más fácil, rápida y cómoda la realización de las actividades en las cuales las manos se encuentran ocupadas durante la praxis dental. Lo anterior dio como resultado una mejor calidad en los datos guardados en un formulario estructurado en el RSE dental y por lo tanto fue posible una mejor toma de decisiones y mejor uso de los sistemas de soporte de decisiones.

Los sistemas de RV son herramientas para crear información legible, exacta y comprensible en los registros médicos de los pacientes. En [24] se presentó un estudio como parte del proyecto para crear una base de datos de textos para un software de RV el cual se usó en el campo de la cirugía reconstructiva de manos. Se usaron 1863 descripciones de cirugías reconstructivas de manos en un periodo de 3 meses en el 15o hospital de sub-especialidades de Khordad, en Irán, para crear la base de datos. Se recolectaron alrededor de 108 voces de médicos después de realizar una cirugía. La base de datos contenía un vocabulario central y más de 1200 palabras para su uso en la unidad de cirugía de manos en hospitales y clínicas. La falta de grabación de datos computarizados es una de las razones de la falla e ilegibilidad de los registros médicos en los centros de salud. Para resolver estos problemas, se utilizó un software para la entrada de datos en el registro médico y se crearon registros electrónicos comprensibles. El software diseñado se probó en la sala de operaciones. La precisión obtenida demostró el potencial de la aplicabilidad práctica del software desarrollado.

El RV se ha convertido recientemente en un recurso suficientemente fiable como para permitir su utilización en el entorno médico. El estudio de [22] midió el efecto del RV para el proceso de dictado en radiología y se estimaron las diferencias en los tiempos de entrega de informe (TEI). Se siguió el flujo de trabajo de 14 radiólogos periódicamente durante 2 años en un hospital universitario. El tamaño de la muestra fue de más de 20,000 exámenes, y los radiólogos fueron los mismos durante todo el estudio. Un TEI se definió como el tiempo desde la formación de imágenes hasta el momento en que el informe estaba disponible para el médico. El RV cortó el TEI en un 81%. La proporción de los informes disponibles dentro de 1 hora escaló de 26% a 58%. El RV disminuye los tiempos de respuesta y por lo tanto permite acelerar todo el proceso de atención al paciente, facilitando la presentación de informes en línea. El RV se aceptó y adoptó fácilmente por los radiólogos. Los hallazgos fomentan la utilización del RV, lo que mejora la productividad y acelera el flujo de trabajo con excelente satisfacción del usuario final.

En el estudio de [25] se evaluó la aplicación del RV para documentar los encuentros de consulta externa en el sistema de RSE en un hospital militar y sus 12 clínicas periféricas. Setenta y cinco médicos se ofrecieron para utilizar el RV, y 64 de ellos (85%) respondieron a un cuestionario en línea posterior a la aplicación para identificar variables relacionadas con la continuidad o discontinuidad en el uso del sistema de RV. Las variables investigadas fueron: características de los usuarios, la experiencia de entrenamiento, logística y utilidad del RV. Cuarenta y cuatro encuestados (69%) continuaron usando el RV y en general consideraron que el software era preciso, más rápido que escribir, mejoró la calidad de las notas y permitieron cerrar el encuentro con el paciente el mismo día. La tasa de abandono del 31% se relacionó con la ubicación en una clínica periférica y la percepción de la insuficiencia de la formación, la disminución de la productividad debido a las inexactitudes del RV, y ninguna mejora en la calidad de las notas.

Los recientes avances en la tecnología de RV y la amplia disponibilidad de computadoras baratas significan que la transcripción en tiempo real en el aula se está haciendo una opción factible para su uso práctico. En [26] se informa sobre la exactitud y legibilidad lograda utilizando un sistema de transcripción de voz de bajo costo que ayuda a los estudiantes sordos y con problemas de audición con la toma de notas en clase. El diseño se presenta para un sistema que intenta equilibrar los problemas de disponibilidad y costo con la facilidad de uso y precisión para proporcionar una alternativa viable y totalmente automática para un firmante o tomador de notas humano. Si bien este estudio se centra en medidas empíricas de opciones de diseño y precisión, también se presentan resultados anecdóticos, con especial atención en la compensación práctica entre el costo, la exactitud y la legibilidad de un sistema de transcripción en tiempo real.

Los recientes avances en hardware y tecnología de software han mejorado los sistemas de RV utilizados para la presentación de informes de radiología. En [27], se diseñó un sistema de dictado de RV continuo en idioma mandarín para los informes de radiología en el Sistema de Información de Radiología (SIR) basado en el motor

de reconocimiento *CMU Sphinx*. Se comparó el sistema con el *IBM ViaVoice pro* v9.1. Los resultados del experimento muestran que ambos sistemas logran una tasa de precisión alta cuando se utiliza el reconocimiento para la generación de informes de tomografía computarizada de Cerebro-Cráneo en el entorno de laboratorio. Además, el sistema se desempeña un poco mejor que *ViaVoice* cuando los datos de entrenamiento y pruebas son ambos seleccionados al azar del mismo corpus textual. Se llegó a la conclusión de que es técnicamente factible emplear el RV para la generación de informes en el SIR. Se llegó a la conclusión de que la tecnología de RV ha madurado hasta el punto en que es ahora viable como medio para proporcionar una entrega más oportuna y de mejor calidad de los servicios médicos a los pacientes, y esta tecnología se aplicará ampliamente en la industria de la salud en un futuro próximo.

A continuación, se mencionan algunos de los trabajos referentes al uso de ontologías para asegurar la interoperabilidad entre sistemas de Registro de Salud Electrónicos (RSE), lo que sustenta el uso de una ontología para el desarrollo del presente proyecto.

Los Registros Personales de Salud permiten a los pacientes mantener su propia información de salud y se perciben como una herramienta importante para el autocontrol del paciente. Sin embargo, la adopción de estos sistemas se ha visto obstaculizada por la gran carga que cae sobre los pacientes para registrar y organizar la información que se transferirá desde otros sistemas clínicos. La opción preferida de la transferencia de información de otros sistemas se ve limitada por la falta de interoperabilidad semántica y sintáctica entre sistemas personales y Registros de Salud Electrónicos. En [19] se describió un modelo de información, aún en desarrollo, que utiliza una ontología para garantizar la integridad semántica entre conceptos registrados por ambos tipos de sistemas de registro y estándares HL7 (*Health Level* 7) para mantener la estructura y la función equivalente. El modelo de información actúa como una capa intermedia entre los sistemas de registro y por lo

tanto no está ligada a ninguna aplicación personal o Registros de Salud Electrónicos específicos.

El uso de RSE ha traído múltiples beneficios para el sector de la salud. Sin embargo, estas ventajas serían mayores si se consiguiese una interoperabilidad sin fallas de los RSE entre Sistemas de Información Sanitaria heterogéneos. En [28] se presentó una propuesta que va un paso más allá y aborda el problema de la interoperabilidad desde una perspectiva impulsada por una ontología formal. Por lo tanto, dicha propuesta permite a un sistema interpretar sobre la marcha los datos clínicos enviados por otro sistema incluso cuando éstos utilizan diferentes representaciones. Se mostraron tres componentes: 1) una ontología que proporciona una representación canónica de declaraciones de RSE, más precisamente de observaciones médicas, que más adelante se especializan por las instituciones de salud de acuerdo a sus modelos propios; 2) un módulo traductor que facilita la definición del bajo nivel de la ontología proveniente de las estructuras de almacenamiento de datos del RSE en particular siguiendo un enfoque semi-automático; 3) un módulo de mapeo que ayuda en la tarea de definir los vínculos entre los términos de alto y bajo nivel de la ontología. Como consecuencia se obtiene un mapeo declarativo especificado en OWL2 (Lenguaje OWL versión 2) lo cual permite una amplia gama de escenarios de asignación al alcance de los administradores de los sistemas de salud.

Los esfuerzos globales de estandarización de los RSE condujeron a la necesidad de un modelo que permita a los practicantes médicos interactuar con los nuevos sistemas de información médica estandarizados y enfocarse en los conceptos y procesos médicos más que en cómo están representados los datos. Se han desarrollado arquetipos que sirven como modelos para representar dichos conceptos o procesos, lo que permite la captura de la información relevante de manera transparente para el usuario. Sin embargo, es necesario asegurar que se captura con precisión toda la información relevante. Lo que conlleva a la inserción de una estructura semántica en cada uno de los arquetipos. En [29] se propuso el desarrollo

de una sub-ontología para cada arquetipo con lo que se representa el contenido semántico de cada uno de ellos. La sub-ontología fue extraída de manera semi-automática desde una ontología de salud estandarizada, como es el caso particular de SNOMED-CT.

La entrega de información correcta al paciente indicado en los puntos de cuidado es esencial para un proceso de cuidados de la salud bien integrado. Sin embargo, la alta sensibilidad del dominio clínico y las vastas diferencias entre los datos y sistemas de salud plantea un gran desafío de interoperabilidad para las soluciones que no emplean fuertes principios semánticos como núcleo en su proceso. En [30] se presentó una aproximación basada en ontologías de dominio combinadas con modelos extensibles, llevados a cabo mediante una amplia terminología de dominio y servicios de conocimiento como centro del proceso para permitir la gobernanza de datos autónoma y la interoperabilidad semántica. El documento aborda las necesidades resultantes y propone una solución que describe los resultados del prototipo utilizado para el desarrollo de este enfoque.

2.2 Análisis comparativo.

A continuación, se muestra el análisis comparativo de los trabajos presentados en la sección anterior.

Tabla 1. - Cuadro comparativo de trabajos relacionados con la implementación del RV.

Título del trabajo	Objetivo	Ámbito	Tecnologías utilizadas	Tipo de aplicación	Estatus
Voice-controlled data entry in dental electronic health record. [23]	Solucionar los problemas con la inserción de datos en los registros de salud electrónicos durante la revisión de pacientes con problemas dentales.	Medicina Odontología	Sistema de RSE *MURDLite*, *Microsoft SQL, Microsoft .NET Framework, Microsoft Visual Studio .NET 2003*	Escritorio	Terminado
The Design of a Reconstructive Hand Surgery Text Database based on a Speech Recognition System in Iran. [24]	Desarrollar un software para minimizar los fallos en la entrada de datos y aumentar la legibilidad de los registros electrónicos de los pacientes.	Medicina. Cirugía de manos.	Reproductor de MP3 *Zen Stonplus*, Micrófono ANDREA, *Microsoft .NET framework*	Escritorio	Terminado
Improvement of report workflow and productivity using speech recognition – A follow-up study. [22]	Mostrar los resultados de la incorporación de un sistema de RV en el área de radiología de un hospital universitario durante dos años.	Medicina Radiología.	*Philips SpeechMagic, Philips Speech Recognition Systems GmbH*	No se específica	Terminado
Lessons Learned from Implementation of Voice Recognition for Documentation in the Military Electronic Health Record System. [25]	Evaluar la aplicación del RV para documentar los encuentros de consulta externa en el sistema de RSE en un hospital militar y sus 12 clínicas periféricas.	Medicina	*Dragon NaturallySpeaking Medical 9*	Escritorio	Terminado
The Application of Speech Recognition in Radiology Information System. [27]	Comparar la tasa de precisión entre un sistema de RV basado en *CMU Sphinx* y el *IBM ViaVoice pro v9.1* usando el idioma mandarín.	Medicina Radiología.	*CMU Sphinx, IBM ViaVoice pro v9.1*, Micrófono *Sennheiser* PC30	Escritorio	Terminado

Tabla 1 - Cuadro comparativo de trabajos relacionados con la implementación del RV (continuación).

Título del trabajo	Objetivo	Ámbito	Tecnologías utilizadas	Tipo de aplicación	Estatus
Achieving acceptable accuracy in a low-cost, assistive note-taking, speech transcription system. [26]	Implementar un sistema de transcripción de voz de bajo costo que ayuda a los estudiantes sordos y con problemas de audición con la toma de notas en clase.	Educación	*Villanova University Speech Transcription System (VUST), Dictionary Building Sosftware (DiBS), Microsoft Speech Recognition Engine (MSRE), Java*, Micrófono inalámbrico *Nady Systems UHF-3*	Escritorio	Terminado

Tabla 2. Cuadro comparativo de trabajos relacionados con el uso de ontologías para asegurar la interoperabilidad entre sistemas.

Título del trabajo	Objetivo	Ámbito	Tecnologías utilizadas	Tipo de aplicación	Estatus
An Ontology-Driven Information Model for Interoperability of Personal and Electronic Health Records. [19]	Proponer el uso de una ontología para asegurar la interoperabilidad entre registros personales de salud y registros de salud electrónicos.	Medicina	PHP, *Protégé*, OWL	Aplicación Web	En proceso
Semantic Interoperability of Clinical Data. [28]	Presentar una propuesta para resolver el problema de la interoperabilidad entre sistemas clínicos mediante el uso de ontologías y razonadores.	Medicina	OWL	No se específica	Terminado
Archetype sub-ontology: Improving constraint-based clinical knowledge model in electronic health records. [29]	Demostrar que con el uso de una sub-ontología para cada uno de los arquetipos utilizados se mejora la comunicación entre registros de salud electrónicos.	Medicina	*Ocean Archetype Editor*	No se específica	En proceso
Service and Model-Driven Dynamic Integration of Health Data. [30]	Presentar una propuesta basada en ontologías de dominio para permitir la interoperabilidad semántica entre diversos sistemas de salud.	Medicina	Estándar CCR, XML	Escritorio / Web	En proceso

Como se aprecia en los cuadros comparativos, en la mayoría de los casos la aplicación de RV para el dictado se realiza dentro del ámbito de la medicina. Además, se ha elegido en todos los estudios el desarrollo de aplicaciones de escritorio para la implementación de la solución. En al menos la mitad de los casos presentados se emplea el modelo oculto de Markov (*Hidden Markov Model* o HMM) el cual es un modelo estadístico para la representación del lenguaje en el RV. En cuanto al software utilizado para el desarrollo y el hardware elegido para la implementación, se demuestra que existe una amplia gama de posibilidades para utilizar, el punto central en la elección de las herramientas usadas en este proyecto se basó en el costo de las mismas, su facilidad de implementación y, en el caso del receptor de audio, su efectividad en la captura de voz sin sonidos ambientales, pues estos últimos causan deterioro en la captura de información acústica.

Para el caso de las ontologías se encontró que se usaron ampliamente en el campo de la medicina, en situaciones donde se resuelve el problema de interoperabilidad entre los diversos tipos de sistemas para gestión de expedientes clínicos electrónicos.

2.3 Propuesta de solución.

Aquí se presenta la alternativa de solución planteada para abordar la problemática mencionada en la sección 1.2 así como las razones que justifican la elección hecha.

Se plantea como solución el lenguaje de programación *Java*, el ambiente de desarrollo *Eclipse*, la biblioteca de funciones *Jena* para la comunicación con la ontología, la metodología de desarrollo *Scrum*, la biblioteca de RV *CMU Sphinx* y como herramienta para desarrollo de ontologías *Protégé* [31]. El esquema de arquitectura propuesta se muestra en la figura 1.

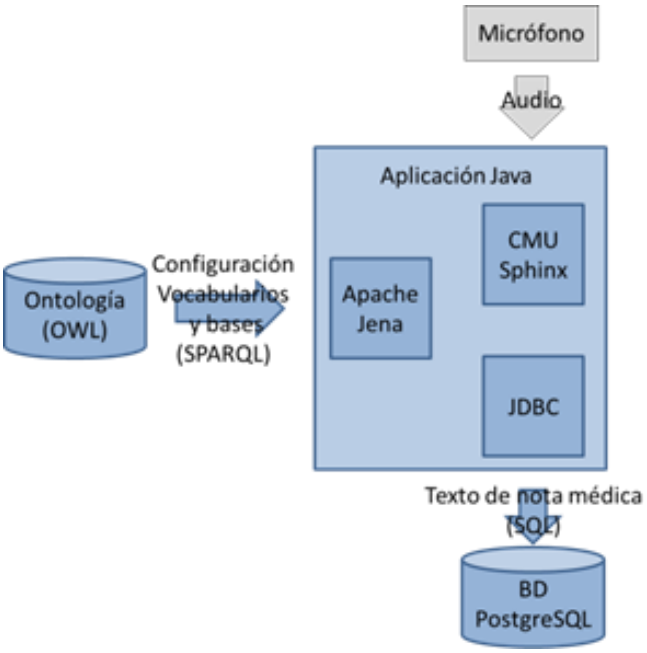

Fig. 1 Esquema de Arquitectura de la aplicación.

Justificación de la solución propuesta.

Se eligió *Java* como lenguaje para desarrollo de la aplicación de RV debido a que su curva de aprendizaje es mucho menor en comparación con otros lenguajes de programación, pues se ha trabajado con él desde el inicio de la maestría. Además, se cuenta con acceso a bibliotecas de funciones para realizar las actividades de RV y acceso a ontologías.

Se optó por *Eclipse* debido a que utiliza menos recursos del sistema en comparación con otras herramientas que existen en el mercado y como es el caso de *Java* también cuenta con una menor curva de aprendizaje, lo que facilita la rápida implementación de la solución.

Se adoptó la biblioteca *Jena* a causa de que cuenta con una documentación mucho más extensa en comparación con otras bibliotecas para el mismo fin.

Se eligió la metodología basada en los principios de *Scrum* ya que cuenta con una lista de características a cumplir en cada iteración, la cual se verifica al término de

ésta. Se tiene más tiempo para realizar las actividades de la iteración en contraste con otras metodologías.

CMU Sphinx, propiedad de la Universidad de *Carnegie Mellon* es la elección en el caso del reconocimiento de voz ya que cuenta con el mayor soporte en comparación con otras bibliotecas. Además de que no requiere conexión a Internet para su funcionamiento.

Por último, se optó por el uso de *Protégé* debido a que se encuentra en constante actualización por su propietario, la Universidad de Stanford, además de contar con amplio soporte por parte de la comunidad de desarrolladores.

Capítulo 3. Aplicación de la metodología.

El presente capítulo describe el desarrollo de la aplicación para el registro de notas médicas mediante el reconocimiento de voz en el expediente clínico electrónico de un sanatorio. Basado en los principios de *Scrum* [32] se realizó una lista del producto, así como tres listas de pendientes que representan las iteraciones que se llevaron a cabo para realizar la aplicación.

3.1. Lista del producto.

A continuación, en la tabla 3, se muestra la lista del producto (*Product Backlog*) que describe cada uno de las características y funcionalidades que contiene el producto al final del proceso de desarrollo. La estimación corresponde al tiempo estimado que se planteó para el desarrollo del elemento y su valor incluye una de tres posibles opciones: necesario, deseable y opcional.

Tabla 3. Lista del producto.

Núm.	Descripción	Estimación	Valor
1	Reconocimiento de voz independiente del hablante.	13 semanas	Necesario
2	Acceso a ontología.	2 semanas	Necesario
3	Acceso a base de datos.	1 semanas	Necesario
4	Interfaz gráfica amigable.	4 semanas	Deseable

3.2. Lista de pendientes por iteración (*Sprint*).

En esta sección se muestran las listas de pendientes (*Sprint Backlog*) de cada *Sprint*, las cuales definen las tareas a realizar en cada una de las iteraciones.

El *Sprint* 1 se refiere a la creación del módulo para el reconocimiento de voz independiente del hablante como se muestra en la tabla 4.

Tabla 4. Lista de pendientes del Sprint 1.

Núm.	Descripción	Estimación
	Reconocimiento de voz independiente del hablante.	**Total 11 semanas**
1	Captura de audios de diversas personas conteniendo terminología del área médica.	2 semanas
2	Selección de audio limpio.	4 semanas
3	Realizar transcripción fonética del audio seleccionado.	2 semana
4	Anexar palabras al vocabulario	1 semana
5	Realizar entrenamiento y pruebas de reconocimiento.	1 semana
6	Codificar el módulo de reconocimiento de voz.	1 semana

En el *Sprint* 2 se implementa el acceso a la ontología y a base de datos como se describe en la tabla 5.

Tabla 5. Lista de pendientes del Sprint 2.

Núm.	Descripción	Estimación
	Acceso a ontología	**Total 2 semanas**
1	Conceptualización y construcción de la ontología.	1 semana
2	Desarrollo del módulo de acceso a la ontología.	1 semana
	Acceso a base de datos	**Total 1 semana**
1	Desarrollo del módulo de acceso a base de datos.	1 semana

El tercer *Sprint* se dedica a la construcción de la interfaz gráfica de usuario, esto se muestra con detalle en la tabla 6.

Tabla 6. Lista de pendientes del Sprint 3.

Núm.	Descripción	Estimación
	Interfaz gráfica amigable	**Total 4 semanas**
1	Diseño de la interfaz gráfica.	1 semana
2	Codificación de la interfaz.	1 semana
3	Prueba y retroalimentación por parte del usuario final.	1 semana
4	Mantenimiento de la interfaz tomando en cuenta retroalimentación.	1 semana

3.3. Arquitectura de la aplicación.

Tomando como base el esquema de la arquitectura presentada en la sección 2.3 de este trabajo, se llevó a cabo el refinamiento de la misma, se explican a continuación cada uno de los módulos que la componen, al final de esta sección se muestra la arquitectura completa.

Módulo de reconocimiento de voz.

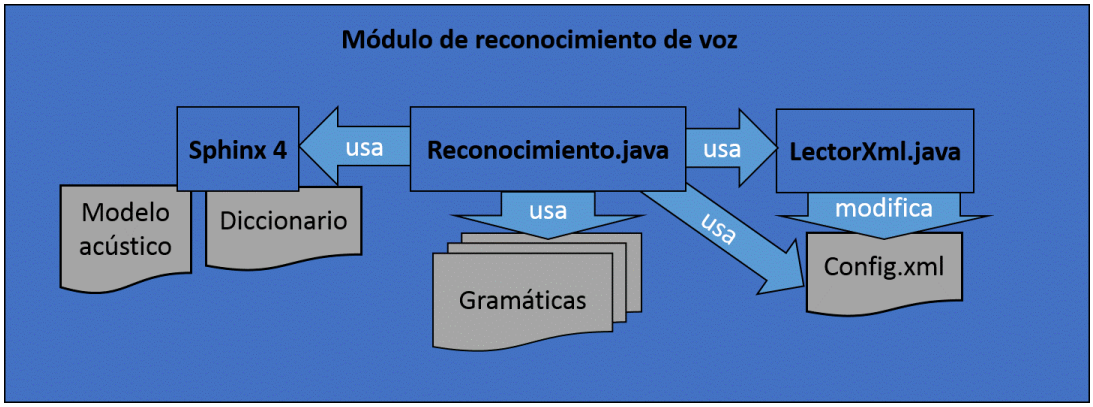

Fig. 2 Módulo de reconocimiento de voz.

En la figura 2 se muestran los elementos que componen el módulo de reconocimiento de voz, la clase en *Java* se llama Reconocimiento y hace uso de la biblioteca *Sphinx* versión 4, la cual a su vez incluye el **modelo acústico** cuya función es contener toda la información estadística del entrenamiento que se hizo con el reconocedor de voz, proceso que se detalla en la sección 3.4. Por otra parte, el **diccionario** contiene una lista de todas las palabras que reconoce el sistema, así como la transcripción fonética de cada una de ellas, en la figura 3 se muestra una porción del archivo de diccionario incluido en la aplicación.

```
sonrIIsa s o n r i_7 s a
sonreIIr s o n r e i_7 r(
sonriEEnte s o n r i e_7 n t e
sonrojAAron s o n r o x a_7 r( o n
sontecomAApan s o n t e k o m a_7 p a n
```

Fig. 3 Porción de palabras en el diccionario.

La clase Reconocimiento también hace uso de un archivo de configuración en formato XML en donde se encuentra toda la información referente a donde se localizan los archivos necesarios para realizar la tarea de reconocimiento de voz, en el listado 1 se muestra una parte del archivo de configuración en donde se señala la ubicación del archivo que contiene el modelo acústico mediante la propiedad *location*.

Listado 1 Segmento del archivo de configuración.

```
1  <componente name="trigramModel"
2   type="edu.cmu.sphinx.linguist.language.ngram.large.LargeTrigramModel"
3   <property name="unigramWeight" value="0.7" />
4   <property name="maxDepth" value="3" />
5   <property name="logMath" value="logMath" />
6   <property name="dictionary" value="dictionary" />
7   <property name="location" value="/prueba1/etc/TESIS.ug.lm.DMP" />
8  </component>
```

Por último, es necesario contar con un archivo de **gramática** en donde se específica la estructura de los enunciados que espera recibir el reconocedor de voz. Dicha gramática se encuentra en el formato *Java Speech Grammar Format* (JSGF) y se muestra un ejemplo de la misma en la figura 4.

```
#JSGF V1.0;

/**
 * JSGF Gramática para la descripción inicial de un paciente
 * Admite enunciados como:
 * El/La paciente de 1-99 años de edad
 */

grammar descripcion;

public <desc> = [<arti>]paciEEnte<anios>;

<arti> = EEl | lAA;
<anios> = dEE (<numero> | <diezytantos> | <veintes> | <decenas>[II<numero>])
<numero> = UUn | UUno | dOOs | trEEs | cuAAtro | cIInco | sEEis | siEEte | O
<diezytantos> = diEEz | OOnce | dOOce | trEEce | catOOrce | quIInce | diecis
dieciOOcho | diecinuEEve;
<veintes> = vEEinte | veintiUUno | veintidOOs | veintitrEEs | veinticuAAtro
veintisiEEte | veintiOOcho | veintinuEEve;
<decenas> = trEEinta | cuarEEnta | cincuEEnta | sesEEnta | setEEnta | ochEEn
```

Fig. 4 Ejemplo de una gramática en formato JSGF.

Módulo de acceso a la ontología.

La ontología es un archivo en lenguaje OWL que permite obtener información acerca de la configuración para conexión a la base de datos en la que se registran las notas médicas. Se presenta en la figura 5 la arquitectura de este módulo.

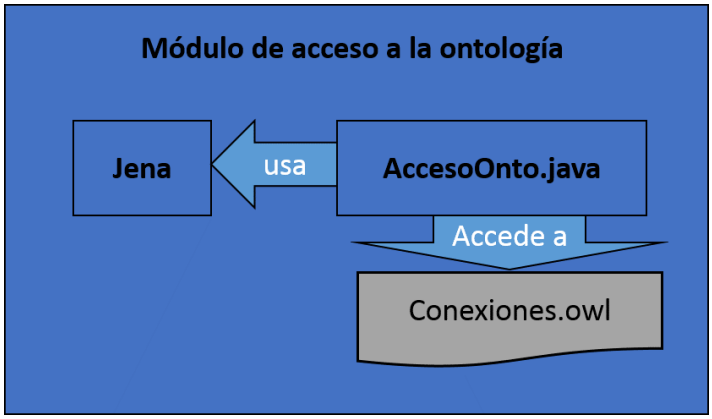

Fig. 5 Módulo de acceso a la ontología.

La clase AccesoOnto.java hace uso de la biblioteca *Jena* para el acceso a la ontología, para obtener la información se realizan consultas en el lenguaje *SPARQL* [33]. En el listado 2 se muestra el ejemplo de una consulta en este lenguaje. Actualmente se trabaja con el acceso a dos bases de datos, la primera en *PostgresQL* y la segunda en *MySQL*, pero se dejaron los fundamentos en la ontología para que a futuro sea posible agregar información de conexión a nuevos gestores de bases de datos.

Listado 2 Ejemplo de consulta con SPARQL.

```
1  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
2  PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
3  PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
4  PREFIX owl: <http://www.w3.org/2002/07/owl#>
5  PREFIX ito: <http://www.ito-depi.edu.mx/tesis/conexiones#>
6
7  SELECT ?z ?a ?b WHERE {
8  ?< ito:nombreEsp “pediatría” ^^xsd:string.
9  ?x ito:usaConexion ?y. ?y ito:baseDeDatos ?z.
10 ?y ito:nombreUsuario ?a. ?y ito:contraseña ?b };
```

Módulo de acceso a la base de datos.

Este módulo permite conectarse a la base de datos con la información obtenida de la ontología en el módulo descrito anteriormente, hace uso de la biblioteca *Postgresql JDBC* debido a que la base de datos del hospital se implementó en esta plataforma. Además, es posible comunicarse con bases de datos *MySQL*. La figura 6 muestra la arquitectura de este módulo.

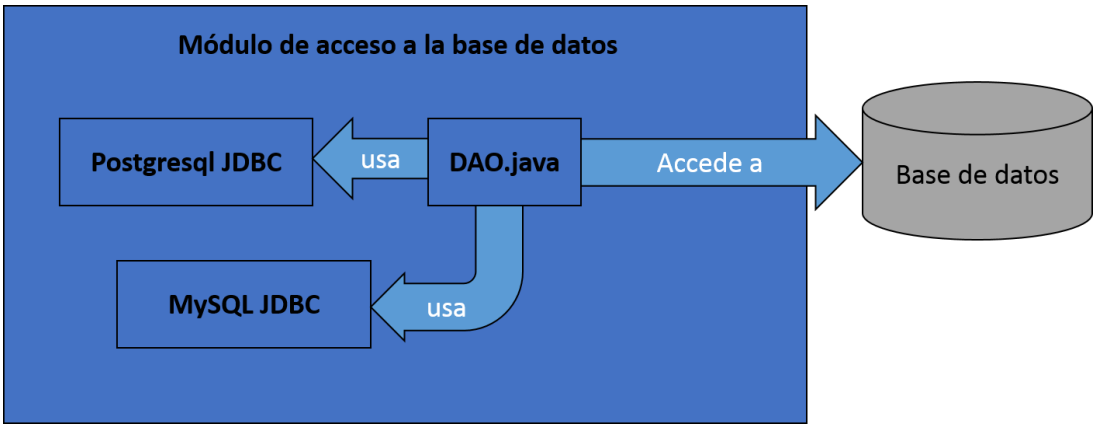

Fig. 6 Módulo de acceso a la base de datos.

Módulo de interfaz gráfica.

Como este módulo solo cuenta con una clase que hace uso de la biblioteca estándar de *Java, javax.swing*, se optó por mostrar su diseño en la figura 7, en lugar de un diagrama con los elementos que lo componen.

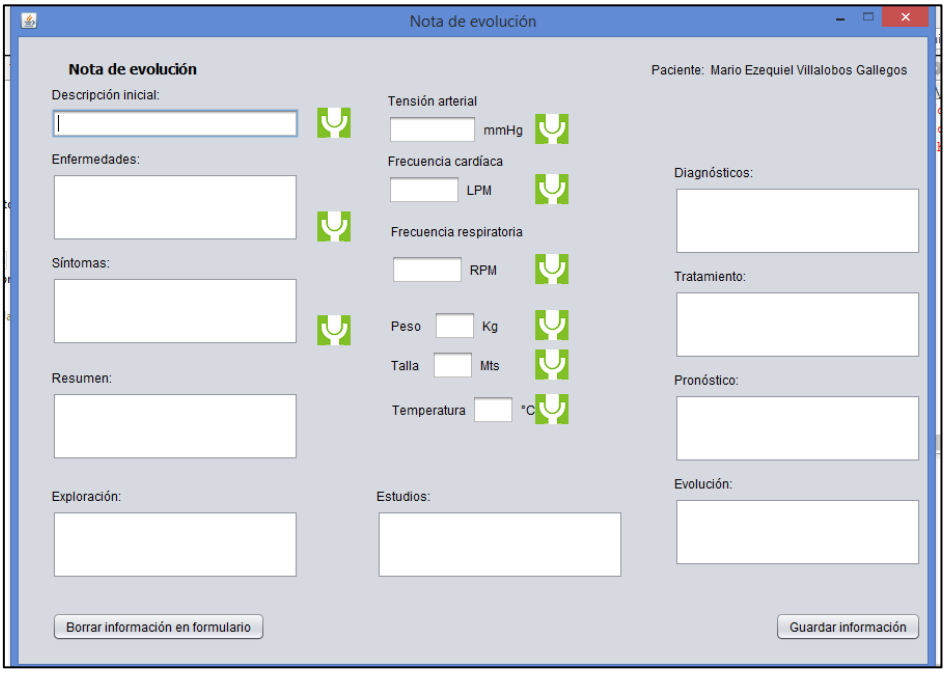

Fig. 7 Interfaz gráfica para el registro de nota médica.

A continuación, se presenta en forma general la arquitectura de la aplicación después del refinamiento.

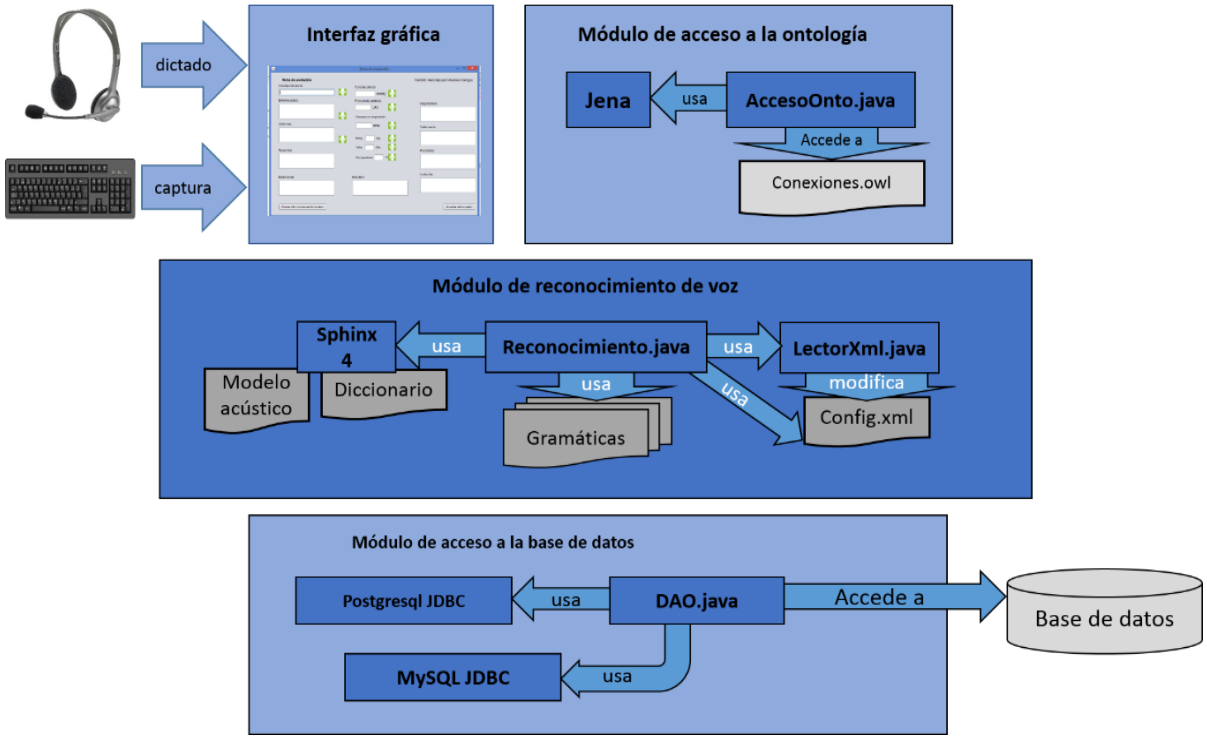

Fig. 8 Arquitectura de la aplicación.

En la sección siguiente se detalla el procedimiento para cumplir con los pendientes del primer Sprint como se indica en la tabla 4.

3.4. Entrenamiento del sistema *Sphinx*.

Para realizar el entrenamiento del sistema *Sphinx* y obtener el modelo acústico se siguió el tutorial que se encuentra en [34] el cual está pensado para ambientes *UNIX*. Por lo anterior fue necesario instalar la versión 12 de *Linux Ubuntu* para seguir los pasos del tutorial. Cabe mencionar que ya existe una versión más nueva del mencionado sistema operativo, pero ocurrieron problemas al momento de ejecutar los *scripts* incluidos en la biblioteca de *Sphinx*. También se probó el proceso en otro

sistema basado en *UNIX*, *Mac OSX* versión 10.9 presentando de igual manera problemas para seguir el tutorial.

Nótese que el tutorial mencionado es para la versión 3 de *Sphinx* el cual se creó para el lenguaje C, pero una vez que se realizó el entrenamiento es posible usar los mismos archivos resultantes para la versión 4 que está hecha para el lenguaje *Java* y que es la versión que se usa en este proyecto.

La parte medular del tutorial y que conlleva mayor carga de trabajo, es la selección de archivos de audio para realizar el entrenamiento, los cuales deben estar libres de ruido. De manera inicial se tomaron como base las 17 horas de audio que contiene el proyecto de código abierto CIEMPIESS [35], sin embargo, fue necesario incluir audios con los términos del área de pediatría para asegurar que se encuentran los fonemas en el modelo acústico con el que trabaja el sistema de reconocimiento.

A continuación, se describe en detalle el procedimiento que se realizó para hacer la selección de audios y la creación de los archivos necesarios para realizar el entrenamiento.

Paso 1: Grabación de los archivos de audio.

Para este caso se tuvo el apoyo de un grupo de alumnos de servicio social, quiene hicieron lectura de un Manual de Pediatría [36] y grabaron sus voces en el software de edición de audio *Audacity* disponible en [37], para después exportar los archivos en el formato *WAV* de *Microsoft* ya que es el formato que admite *Sphinx* para realizar el entrenamiento.

Para tal fin fue necesario configurar las preferencias de grabación en *Audacity*, de tal manera que la frecuencia de muestreo estuviera en 16,000 Hz (Hertz), el formato de muestra como 16-bit y fijar los canales de grabación a uno (1-Mono).

Es muy importante no rebasar los límites mostrados en la figura 9 para evitar que se generen ruidos.

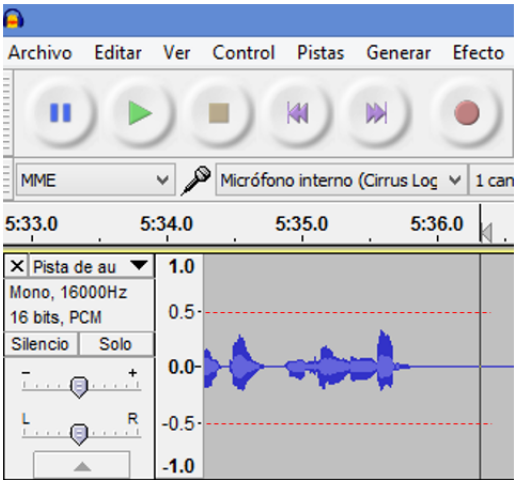

Fig. 9. Niveles de grabación máximos recomendados.

Paso 2: Selección de porciones de audio.

Una vez que se contó con todos los archivos de audio proporcionados por los alumnos de servicio social, se pasó a elegir las porciones que no contenían ruidos ambientales (por ejemplo, los sonidos emitidos por la respiración de la persona que realiza la grabación, el paso de automóviles, movimiento del micrófono, uso del teclado de la computadora, entre otros). De igual manera se eliminaron las secciones de audio donde hay una descompensación del sonido, es decir, momentos en los que el hablante disminuye o aumenta demasiado el volumen de su voz a tal grado que ésta se hace incomprensible. En la figura 10 se muestra la interfaz de *Audacity*, en donde se realizó la edición de los archivos de audio.

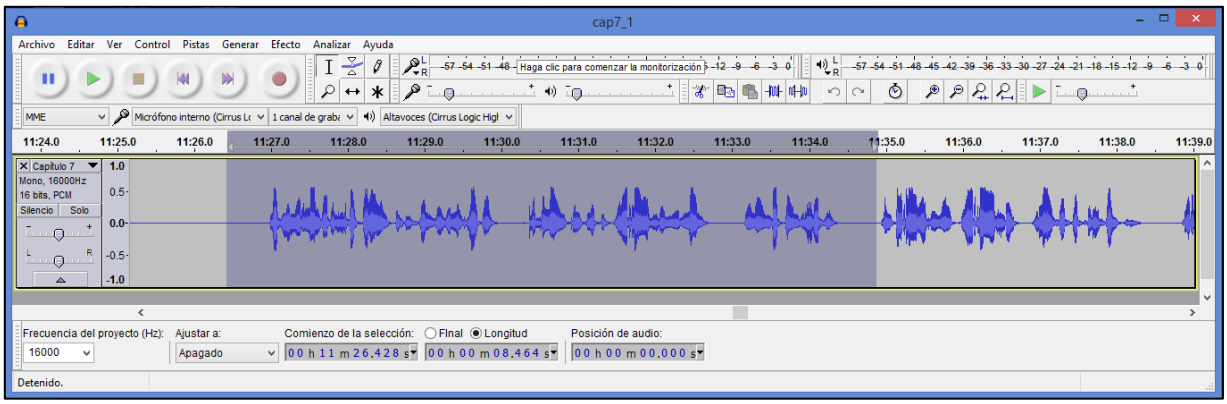

Fig. 10 Interfaz de Audacity.

Una vez que se eliminó el audio con ruido, el tiempo de sonido restante se segmentó en porciones de entre 5 a 10 segundos para ser exportadas en el formato *WAV PCM* de 16 *bit* de *Microsoft.*

Paso 3: Realizar transcripción fonética.

La transcripción fonética se define como la representación escrita de los sonidos que pronuncia una persona [38], entonces es necesario representar en un archivo de texto la transcripción de cada uno de los archivos de audio que se obtuvieron en el paso anterior, esto para que *Sphinx* identifique qué es lo que contienen los archivos de audio.

Se inició realizando un proceso de pre-transcripción usando el alfabeto fonético Mexbet [39]. En dicho proceso se indica la vocal tónica de cada palabra, para este fin el proyecto CIEMPIESS cuenta con una función codificada en *Python* que realiza la tarea por palabra, por lo que fue necesario crear una rutina que permitiera agilizar el proceso recorriendo el archivo de texto con los enunciados a convertir y crea un archivo con la misma información, pero indicando la vocal tónica de cada palabra.

Se muestra a continuación en el listado 3 la rutina mencionada, la cual se codificó en el entorno de desarrollo *LiClipse* [40].

Listado 3. Rutina para recorrer el archivo de texto.

```
1  from vocal_tonica import vocal_tonica
2  f = open('entrada.txt','r')
3  r = open('resultado.txt','w')
4  i = 73
5  for linea in f:
6     palabras = linea.split(' ')
7     linea_nueva ="<s><sil>"
8     for pal in palabras:
9        nueva_pal = vocal_tonica(pal)
10       linea_nueva = linea_nueva + nueva_pal + " "
11    linea_nueva = linea_nueva + "<sil></s>" + "(mario00" +
      str(i) +") \n"
12    i = i + 1
13    r.write(linea_nueva)
14 print( "terminado" ) #Terminó el proceso
```

En la línea 4 se indica el número consecutivo para indicar el nombre del archivo de audio correspondiente y que se usa en la línea 11 para facilitar el proceso de identificación.

Después se procedió a sustituir en el archivo resultante las letras *ñ* para evitar problemas de codificación y las letras *x* que en el español de México tienen 4 sonidos distintos según la palabra en donde se encuentren siguiendo los lineamientos indicados en [35].

Los elementos que componen cada una de las líneas en el archivo resultante se ejemplifican a continuación.

<s> <sil> aspEcto A princIpios bAsicos dE lA electrocardiografIa pUnto Uno colocaciOn dE derivaciOnes pUnto dOs <sil> </s> (mario0028)

Dónde: <s></s> Representa el inicio y fin del enunciado (*sentence* en inglés).

<sil> Indica que hay silencio en esa sección del archivo de audio.

(mario028) Indica el nombre del archivo de donde se está obteniendo la transcripción.

Nótese que en cada una de las palabras se representa la vocal tónica con letra mayúscula. También se observa que no es necesario que los enunciados tengan coherencia, como en el ejemplo “aspecto a principios básicos de la electrocardiografía punto uno colocación de derivaciones punto dos”, sin embargo, cumple con la característica de ser un archivo de audio de entre 5 a 10 segundos de duración.

Al finalizar este paso se obtuvieron todos los enunciados en formato de pre-transcripción de cada uno de los segmentos de audio seleccionados para el entrenamiento. En el paso 5 se describe como se termina el proceso de la transcripción fonética.

Paso 4: Especificar la ubicación de los archivos de audio.

Es necesario indicar a la herramienta *Sphinx* la ubicación y el nombre de los archivos de audio con los que realiza el entrenamiento del modelo acústico y, en su caso, también de los archivos para realizar pruebas en línea de comando.

Paso 5: Anexar palabras al vocabulario.

Para que el sistema reconociera las palabras incluidas en el archivo de pre-transcripción fue necesario incluirlas en el diccionario fonético o vocabulario. Dicho vocabulario es una lista de palabras que entiende el sistema de reconocimiento, así como la transcripción fonética de cada una de ellas. En la figura 11 se muestra un segmento del diccionario fonético indicando la palabra seguida de la transcripción fonética correspondiente.

```
$antocromIIa s a n t o k r( o m i_7 a
fiEEbre f i e_7 b r( e
fibrOOsis f i b r( o_7 s i s
miocardIItis m i o k a r( d i_7 t i s
mitocondriAAl m i t o k o n d r( i a_7 l
```

Fig. 11 Ejemplos de transcripción fonética dentro del diccionario.

A continuación, se explica a detalle la transcripción de la palabra xantocromía (coloración amarilla de una parte del cuerpo), en donde la "x" se reemplaza con $ (signo de moneda) pues es el símbolo para el sonido /s/ que toma la letra "x" en este caso; además puesto que el sistema *Sphinx* no identifica mayúsculas de minúsculas [41] es necesario señalar la vocal tónica duplicando la vocal correspondiente, para este ejemplo el acento lo tiene la letra "i" quedando como "II" (doble i, en mayúsculas). Del lado derecho de la palabra escrita en su formato de pre-transcripción se define la transcripción fonética de la misma, en donde se observa que el signo $ pasa a convertirse en "s", la letra "c" pasa a convertirse en "k", la letra "r" pasa a convertirse en "r(" y por último la vocal tónica "i" pasa a indicarse como "i_7". Lo anterior tomando en consideración las reglas del alfabeto Mexbet.

Paso 6. Conversión de formatos de audio.

Para que los formatos de los archivos de entrenamiento coincidan con los del corpus del CIEMPIESS, fue necesario cambiarlos usando *SoX* [42], una aplicación en línea de comando para este fin. A continuación, se muestra la rutina para realizar la conversión de todos los archivos contenidos en una carpeta.

```
for i in ./*.wav; do sox $i $.sph; done
```

Esta rutina se ejecutó en el ambiente *Ubuntu Linux* ya que ocurrieron problemas para ejecutar la herramienta *SoX* dentro del ambiente *Windows* 8.

Paso 7. Entrenamiento.

Una vez realizados todos los pasos anteriores se procedió al entrenamiento, el cual consistió en convertir los archivos de audio en vectores de características, pues la biblioteca de RV no trabaja directamente con las señales acústicas, lo anterior lo realiza el sistema *Sphinx* [34], para con esto obtener el modelo acústico. La rutina para crear los vectores de características se muestra a continuación.

```
perl scripts_pl/make_feats.pl -ctl etc/TESIS_train.fileids
```

Donde, *make_feats.pl* es un script incluido en *Sphinx* y TESIS_train.fileids es el archivo que contiene todas las rutas de los archivos que servirán en el proceso de entrenamiento. Los vectores de características creados constan de 13 dimensiones y se conocen como MFCC (por sus siglas en inglés, *Mel-frequency cepstral coefficients*) ya que está comprobado que es el algoritmo que tiene mejor desempeño para representación de señales acústicas en sistemas basados en modelos de *Markov*, tal como *Sphinx*. Cuando se terminó el proceso de crear los vectores se ejecutó el entrenamiento mediante la sentencia.

```
perl scripts_pl/RunAll.pl
```

Una vez que se presentó un mensaje como el de la figura 12 significó que el modelo acústico estaba entrenado.

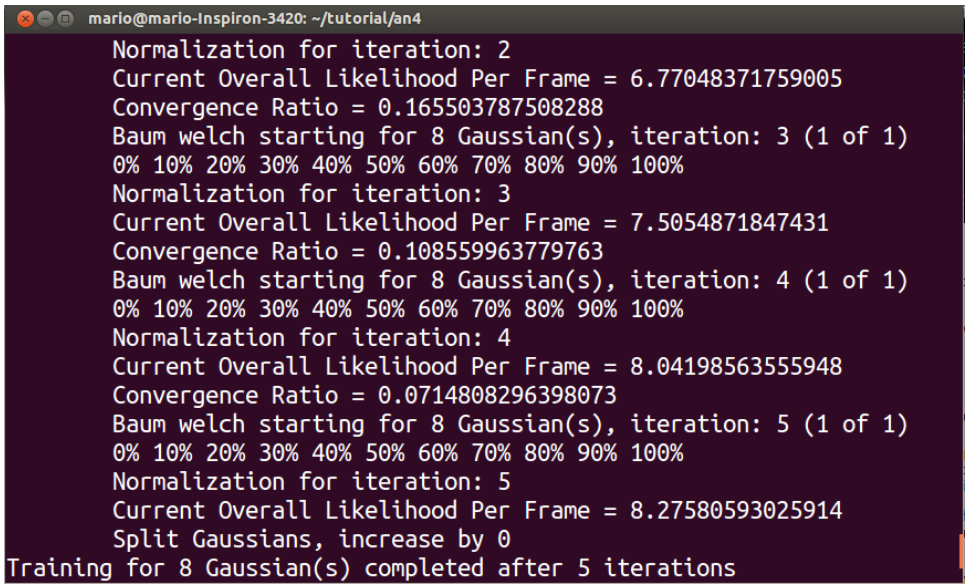

Fig. 12 Resultado del entrenamiento.

Paso 7.1. Prueba en línea de comando (opcional).

Es posible realizar una ronda de prueba en línea de comando para obtener de forma rápida una tasa de error por palabra (*WER*, por sus siglas en inglés), generalmente es mayor en comparación con la decodificación en vivo haciendo uso de una gramática. En la figura 13 se muestran los resultados para una ronda de prueba donde la tasa de error por palabra es de 16.9%.

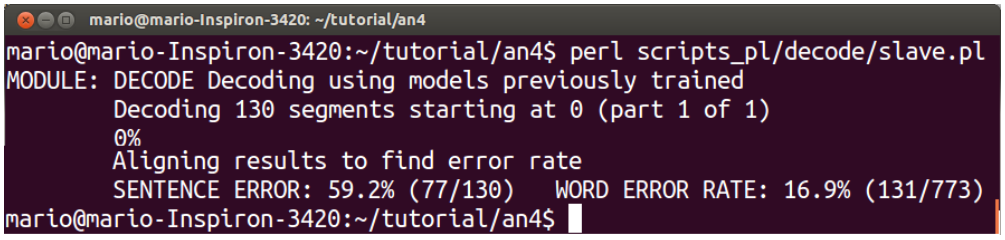

Fig. 13 Resultado de prueba en línea de comando.

Una vez terminado el entrenamiento del modelo acústico es posible empaquetar los archivos en un *JAR* para incluirlos en la aplicación mediante el comando:

```
jar -cf nuevo_nombre.jar nombre_de_la_carpeta
```

La estructura de los archivos en el paquete es:

Archivo *JAR*

- Carpeta etc
    - Archivo de fonemas.
    - Diccionario fonético.
    - Modelo acústico.
- Carpeta *model_parameters*
    - Archivos de definición de parámetros creados por *Sphinx*.

En el caso de la prueba en línea de comando el sistema calcula el WER, pero es posible realizar el cálculo manualmente cuando se realiza RV en vivo aplicando la fórmula:

$$WER = (I + D + S) / N$$

Donde:

- I es el número de palabras insertadas, es decir palabras que se encuentran de más en el texto resultante, por ejemplo, si se dictaron tres palabras y se reconocieron 4, el número I = 1.
- D es el número de palabras borradas, contrario a I, si se dictaron cinco palabras y se reconocieron 3, entonces D=2.
- S es el número de palabras sustituidas, es decir el número las palabras que el reconocedor escribe y que no fueron dictadas por el usuario.
- N se refiere al total de palabras dictadas.

El porcentaje de acierto está dado por:

$$\% \text{ de acierto} = (N - D - S) / N$$

Esta fórmula es similar a la del WER, pero no toma en cuenta las palabras insertadas.

El entrenamiento de un sistema independiente del usuario requiere contar con alrededor de 200 horas de audio, lo que resulta prohibitivo en el caso el proyecto de tesis, por lo que fue necesario buscar otros corpus orales para complementar las 17 horas iniciales. Aunque existen muchos corpus de este tipo, la gran mayoría tienen costo para su uso y donde el caso más económico ronda los $1,250 dólares [43]. La búsqueda de corpus gratuitos en español de México arrojó sólo un resultado: Corpus VoCMex [44]; este corpus contiene grabaciones realizadas desde un micrófono y llamadas por teléfono celular de 33 hablantes, 20 hombres y 13 mujeres del noroeste del país. Se creó en la Universidad Autónoma de Baja California y se encuentra disponible a través de una carpeta compartida de Google Drive y cuenta con archivos de etiquetado para una porción de los audios.

3.5 Decodificación en vivo.

Para realizar la decodificación o reconocimiento en vivo y aumentar el porcentaje de acierto, es necesario incluir una gramática, que es la manera en que el reconocedor espera se presentaran las palabras dictadas por el usuario. A continuación, se muestra una porción del archivo de gramática en la figura 14.

```
#JSGF V1.0;

/**
 * JSGF Grammar
 */

grammar ES-MX;

public <descripcion> = [<arti>]<paci><anios>;
public <problema> = padEEce <padecimiento>+;
public <subjetivo> = sEE refiEEre cOOn <sintomas>+;
//public <objetivo1> = sIIgnos vitAAles <signos>*;
public <objetivo2> = <signos>+;
```

Fig. 14. Segmento de archivo de gramática.

Como se observa, la gramática es similar a la notación de Backus-Naur, pero en este caso los enunciados válidos para ser reconocidos se definen como públicos y a continuación se específica su estructura. Los términos que no se especifican como públicos, p. ej. <padecimiento>, solo se usan si se encuentran dentro de otro término que sí lo sea. Los términos entre corchetes se refieren a que son opcionales, es decir el usuario puede o no pronunciarlos. Para el caso del signo más (+) se refiere a que el término aparece una o más veces en el enunciado, mientras que el asterisco (*) significa que la palabra se espera cero o más veces.

Se encontró que cuando se aumenta el número de enunciados públicos en la gramática, disminuye el porcentaje de acierto del reconocedor, por lo que fue necesario realizar un proceso en el que se cambia de gramática en cada una de las secciones que forman una nota médica. Esto para hacer uso de gramáticas que solo contienen un enunciado y no ver afectado el porcentaje de acierto de la aplicación de RV.

3.6. Codificación del módulo de reconocimiento de voz

Para la codificación del módulo de RV se usó el archivo *JAR* resultante que se describió en el apartado 3.4, además de las bibliotecas propias de *Sphinx*. En la figura 15 se muestran los archivos que contiene la biblioteca de RV para este proyecto, en donde, el archivo prueba1.jar es donde se encuentran contenidos los archivos resultantes del proceso de entrenamiento. Los demás archivos son necesarios para el correcto funcionamiento del reconocedor.

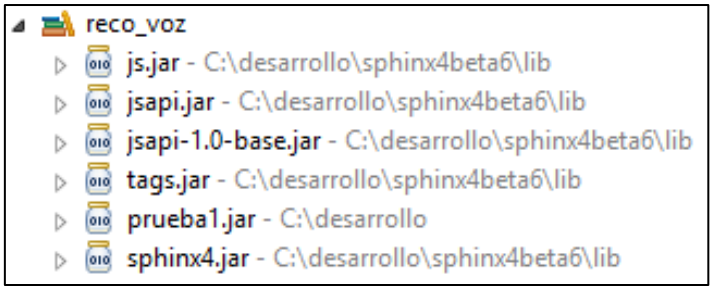

Fig. 15. Archivos JAR del módulo de RV.

Con lo anterior se procedió a la codificación del módulo. Al crearse una instancia de la clase Reconocimiento, se hace la búsqueda de los recursos necesarios para su funcionamiento, en este caso el micrófono y la asignación del archivo de configuración de la aplicación, en donde se indica dónde se encuentran los demás archivos necesarios para su funcionamiento. En el listado 4 se muestra el código de esta sección.

Listado 4. Código del constructor de la clase Reconocimiento.

```
1  public Reconocimiento(String nombreConfig){
2      this.nombreConfig = nombreConfig;
3      modificaArchivoConfiguracion(this.nombreConfig);
4      cm = new ConfigurationManager("ES-MX.config.xml");
5      recognizer = (Recognizer) cm.lookup("recognizer");
6      microphone = (Microphone) cm.lookup("microphone");
7  }
```

Cabe destacar la línea 3 en el listado 4, donde se modifica la información del archivo de configuración para indicar la gramática correspondiente. Para esto se codificó una clase llamada LectorXml.java, que utiliza la biblioteca *JDOM* para acceder al archivo XML desde *Java*. El código del método para cambiar el archivo de gramática se muestra en el listado 5, donde en la línea 11 se indica que se busca el atributo *grammarName* al cual se le asigna el valor ingresado en el atributo del método *modificaNombreGramatica*.

Listado 5. Código para cambiar archivo de gramática.

```
1  public void modificaNombreGramatica(String valor){
2   SAXBuilder builder = new SAXBuilder();
3   try{
4    Document document = (Document) builder.build("ES-MX.config.xml");
5    Element rootNode = document.getRootElement();
6    List<Element> lista = rootNode.getChildren();
7    for(Element ele : lista){
8     if(ele.getAttributeValue("name").equals("jsgfGrammar")){
9      List<Element> hijos = ele.getChildren();
10     for(Element h : hijos){
11      if(h.getAttributeValue("name").equals("grammarName")){
12       Attribute atri = new Attribute("value",valor);
13       h.setAttribute(atri);
14       XMLOutputter xmlOutput = new XMLOutputter();
15       xmlOutput.setFormat(Format.getPrettyFormat());
16       xmlOutput.output(document, new FileWriter("ES-MX.config.xml"));
17       }}}}}
18  catch(JDOMException j){ j.printStackTrace();  }
19  catch (IOException e) { e.printStackTrace();  }
20 }
```

Volviendo al código del módulo de RV, se encuentra el método reconocer, el cual realizar la decodificación en vivo y retorna una línea de texto. A continuación, en el listado 6 se muestra el código resultante.

Listado 6. Método reconocer del módulo de RV.

```
1  public String reconocer(){
2   String texto = "";
3   recognizer.allocate();
4   microphone.initialize();
5   if (!microphone.startRecording()){
6    System.out.println("No se puede iniciar el micrófono.");
7    recognizer.deallocate(); }
8   Result result = recognizer.recognize();
9   if (result != null){
10   texto = texto + "-" +
convertirAtextoNormal(result.getBestFinalResultNoFiller()); }
11  microphone.stopRecording();
12  recognizer.deallocate();
13  microphone.clear();
14  return texto;
15 }
```

En la línea 10 del listado 6 se encuentra el método *convertirATextoNormal*, el cual realiza el proceso de transformar el texto devuelto por el método *getBestFinalResultNoFiller.* Lo anterior es necesario debido a que el reconocedor realiza la conversión de voz a texto según el diccionario fonético descrito en la sección 3.4, por lo que se obtienen palabras en formato de pre-transcripción y es necesario convertir estas palabras en texto comprensible para el usuario final. En el listado 7 se muestra de manera parcial el código para la conversión a texto normal.

Listado 7. Código para conversión a texto normal.

```
1  private String convertirAtextoNormal(String txt){
2    String res = txt.replaceAll("aa", "a");
3    res = res.replaceAll("ee", "e");
4    res = res.replaceAll("ii", "i");
5    res = res.replaceAll("oo", "o");
6    res = res.replaceAll("uu", "u");
7    res = res.replaceAll("kkss", "x");
8    res = res.replaceAll("jj", "j");
9    res = res.replaceAll("nn", "ñ");
10   res = res.replaceAll("dieci", "1 ");
11   ...
12   res = res.replaceAll("noventai", "9 ");
13   res = textoAnumeros(res);
14   return res;
15 }
```

El método *convertirAtextoNormal* inicia reemplazando la vocal tónica, lo cual está indicado en el texto devuelto por el reconocedor duplicando la letra, como se indica de la línea 2 a la 6 del listado 7. Una vez hecho esto se procede a sustituir el texto “kkss” por la letra “x”, “jj” simplemente por “j” y la “nn” por la letra “ñ”. A continuación, se buscan las decenas en el texto se muestra en la línea 12 de código se está reemplazando el caso del “noventai” por el número “9”. Ya con esto, se termina convirtiendo las unidades restantes del 1 al 9, esto se realiza mediante el método *textoAnumeros* en la línea 13. En el listado 8 se muestra de manera parcial el código del método *textoAnumeros*.

Listado 8. Código para conversión de texto a número.

```
1  private String textoAnumeros(String txt){
2    String res = txt.replaceAll(" uno", "1");
3    res = res.replaceAll(" dos", "2");
4    ...
5    res = res.replaceAll(" nueve", "9");
6    res = res.replaceAll("diez", "10");
7    res = res.replaceAll("once", "11");
8    ...
9    res = res.replaceAll("quince", "15");
10   res = res.replaceAll("veinte", "20");
11   res = res.replaceAll("treinta", "30");
12   ...
13   res = res.replaceAll("noventa", "90");
14   return res;
15 }
```

El método *textoAnumeros* inicia convirtiendo el texto del uno al nueve a números como se indica de la línea 2 a la 5 del listado 8, después sigue convirtiendo los números del diez al quince y las decenas del veinte al noventa.

A partir de la sección 3.7, se describe la ejecución del segundo Sprint, el cual trata de la implementación del acceso a la ontología y a la base de datos, tal como se mencionó en la tabla 5.

3.7 Conceptualización de la ontología.

Para generar la ontología se realizó la especificación de la misma y se realizaron las 11 tareas que se llevan a cabo en el proceso de conceptualización de *METHONTOLOGY* [7].

Especificación. Se define la razón u objetivo por el cual la ontología se construye, los usos que tendrá y los usuarios finales a los que va dirigida.

El objetivo de esta ontología es permitir la interoperabilidad entre diversas bases de datos incluidas en los expedientes clínicos electrónicos. Va dirigida al personal técnico que actualiza la información de conexiones a bases de datos con la aplicación.

Además, permite obtener información de los tipos de notas médicas y de las gramáticas usadas por cada una de ellas.

Conceptualización. Aquí se tratan actividades relacionadas con el modelado conceptual del conocimiento, a estas actividades se les conoce como tareas.

Tarea 1: Glosario de términos.

En esta tarea se identifican todos los términos (conceptos, relaciones, instancias y atributos entre otros) importantes del dominio, se enlistan y se describen en lenguaje natural, lo cual se muestra en la tabla 7.

Tabla 7. Glosario de términos.

Nombre	Sinónimo	Acrónimo	Tipo	Descripción	Atributos
conexion	--	--	Concepto	Elemento que contiene los datos de conexión.	baseDeDatos nombreUsuario contraseña nombreTabla tablaComun
especialidad	--	--	Concepto	Especialidad médica.	nombreEspecialidad
nota	--	--	Concepto	Contiene los campos particulares de cada tipo de nota médica, así como las gramáticas usadas por la aplicación.	tipoNota usaGramatica campo
notaComp	--	--	Concepto	Contiene los campos comunes para todo tipo de nota médica.	tipoNota campo
Controlador	Driver	--	Concepto	Indica el paquete y la clase del controlador para un tipo de gestor de base de datos.	rutaClase
seDirigeA	--	--	Relación	Indica hacia la especialidad que se dirige una conexión.	--
usaConexion	--	--	Relación	Indica que conexión utiliza una especialidad en particular.	--
seConectaCon	--	--	Relación	Indica que una nota hace uso de un tipo de conexión.	--
usa	--	--	Relación	Indica que alguna conexión usa un tipo de controlador	--
otrosCampos	--	--	Relación	Indica que alguna nota cuenta con campos comunes.	--

Tarea 2: Taxonomía de conceptos.

En esta tarea se establece la jerarquía entre los conceptos del dominio, se basa en el glosario de términos. Se muestra que los conceptos son subclases de *Thing* (Cosa), el cual es el término principal predeterminado que maneja la aplicación *Protégé*. La taxonomía de conceptos resultante se muestra en la figura 16.

Fig. 16. Taxonomía de conceptos.

Tarea 3: Diagrama de relaciones binarias.

Aquí se construyen diagramas o mapas conceptuales para identificar relaciones entre conceptos (directas e inversas). Dicho diagrama representa la ontología y se muestra a continuación en la figura 17.

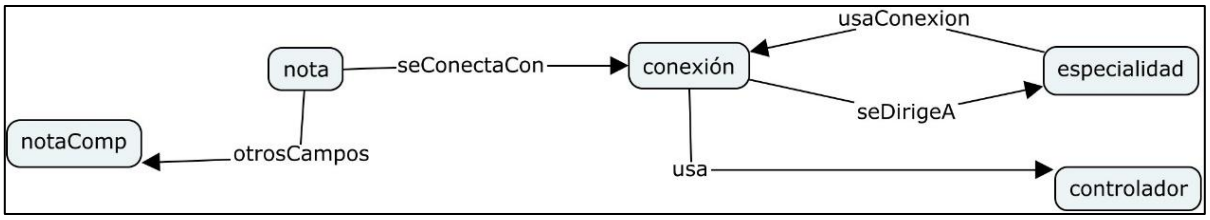

Fig. 17. Diagrama de relaciones binarias.

Tarea 4: Diccionario de conceptos.

Se identifican los conceptos más relevantes del dominio, así como las relaciones existentes entre estos. En la tabla 8 se muestran los resultados de esta tarea.

Tabla 8. Diccionario de conceptos.

Concepto	**Relaciones**
conexión	seDirigeA → especialidad usa → controlador
especialidad	usaConexion → conexión
nota	seConectaCon → conexión otrosCampos → notaComp

Tarea 5: Definición de relaciones binarias en detalle.

En esta tarea se identifican los detalles de las relaciones como dominio, rango y cardinalidad, el resultado es una lista de relaciones, concepto origen (dominio), concepto destino (rango) y cardinalidad (1:0 … 1, 1 … 1, 1:* … *). En la tabla 9 se muestran dichas definiciones.

Tabla 9. Relaciones binarias en detalle.

Relación	**Dominio**	**Rango**	**Cardinalidad**
usaConexion	especialidad	conexion	1:*
seDirigeA	conexion	especialidad	1:1
seConectaCon	nota	conexion	1:*
otrosCampos	nota	notaComp	*:1
usa	conexión	controlador	*:1

Tarea 6: Definición de atributos de instancia en detalle.

Se identifican los atributos que cualquier instancia de un concepto tiene, del mismo modo que existen atributos con alcance de instancia en la programación orientada a objetos (POO). Cada atributo cuenta con tipo de valor, rango de valores y

cardinalidad. El resultado es una lista de atributos, concepto al que pertenecen, tipo de valor, rango de valores y cardinalidad. A continuación, se muestra dicha lista en la tabla 10.

Tabla 10. Atributos de instancia en detalle.

Atributo	Concepto al que pertenecen	Tipo de valor	Rango de valores	Cardinalidad
nombreEspecialidad	especialidad	Texto	*	1
baseDeDatos	conexión	Texto	*	1
nombreUsuario	conexión	Texto	*	1
contrasena	conexión	Texto	*	1
nombreTabla	conexión	Texto	*	1
tablaComun	conexión	Texto	*	1
tipoNota	nota notaComp	Texto	*	1
usaGramatica	nota	Texto	*	1..*
campo	nota notaComp	Texto	*	1..*
rutaClase	controlador	Texto	*	1

Tarea 7: Definición de atributos de clase en detalle.

Se identifican los atributos a nivel de concepto o clase del mismo modo que existen atributos con alcance de clase en POO. Mientras que los atributos de instancia siempre existen, no ocurre lo mismo con los atributos de clase. En este caso no se identificaron atributos de clase.

Tarea 8: Definición de constantes en detalle.

Se identifican las características de las constantes del dominio, aunque no todos los dominios cuentan con constantes. En este caso no se identificaron constantes.

Tarea 9: Definición de axiomas formales.

Se identifican los axiomas (expresiones lógicas siempre verdaderas) que representan restricciones del dominio. En este caso se encontró que solo es necesario crear una instancia de la clase nota complementaria, pues esta contiene los campos comunes para todas las notas médicas.

Tarea 10: Definición de reglas.

Se identifican y describen las reglas del dominio. Las reglas permitirán identificar consecuencias o acciones en función de condiciones, dicho de otra forma, permiten inferir conocimiento. En este caso no se definieron reglas.

Tarea 11: Definición de instancias.

Por último, se describen algunos ejemplos de instancias del dominio (las que aparecieron en la tarea 1). Cada instancia cuenta con nombre, nombre del concepto al que pertenece y los valores de sus atributos de instancia, si se conocen. En la tabla 11 se definen algunos ejemplos.

Tabla 11. Definición de instancias.

Nombre	Concepto al que pertenece	Valores de sus atributos de instancia
radiologia	especialidad	nombreEspecialidad = “Radiología”
conexRadio	conexion	baseDeDatos = “jdbc:postgresql://localhost/radiologiaBD” nombreUsuario = “postgres” contraseña = “postgres1” nombreTabla=”Soap”
soap	nota	tipoNota = “soap” usaGramatica = “descripcion” campo=”idPaciente”
postgres	controlador	rutaClase = “org.postgresql.Driver”

3.8 Construcción de la ontología.

Como se mencionó en la sección 2.3, se usó *Protégé* para la construcción de la ontología una vez que se terminó la conceptualización de la misma. La interfaz amigable de la aplicación permite crear la ontología sin codificar una sola línea.

En la tarea 2 se definió una taxonomía de conceptos la cual se representa en la pestaña de clases de la aplicación, el resultado se muestra en la figura 18.

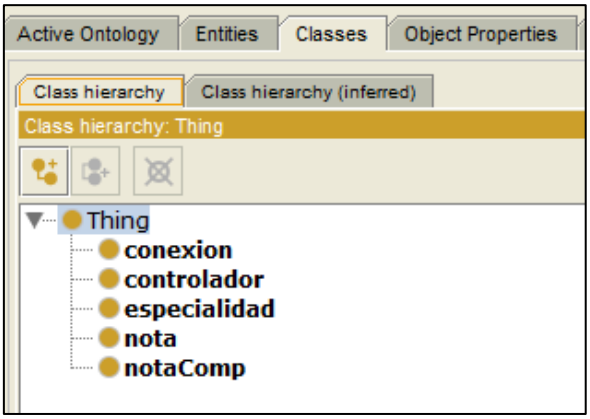

Fig. 18. Taxonomía de conceptos representada en Protégé.

Por otro lado, en la tarea 5 se definieron las relaciones que existen entre los conceptos. En la figura 19 se muestran estas relaciones que en *Protégé* se definen como propiedades de objeto.

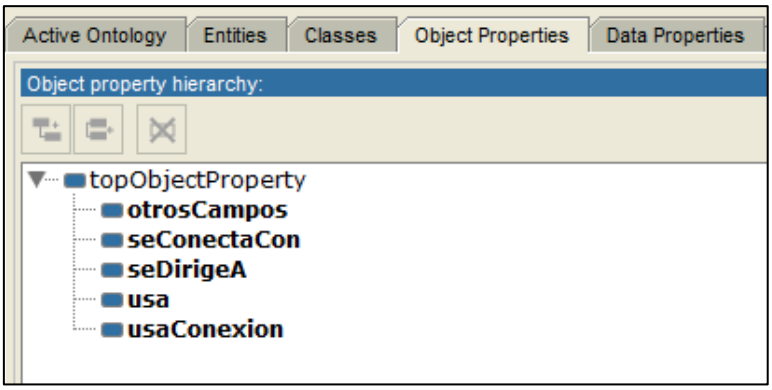

Fig. 19. Relaciones entre conceptos.

Además, se permite definir el dominio y rango de las relaciones binarias, tal como se describió en la tarea 5. En la figura 20 se muestra un ejemplo con la relación *seConectaCon*.

Fig. 20. Detalle de la relación binaria seConectaCon.

En la tarea 6 se definieron los atributos de instancia, los cuales quedan representados como se muestra en la figura 21. En este caso dentro de la pestaña propiedades de datos.

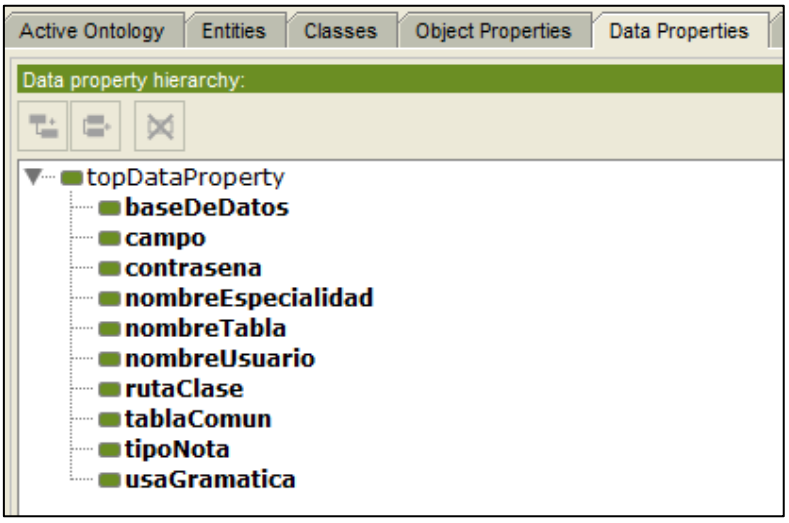

Fig. 21. Atributos de instancia.

También se permite detallar el concepto al que pertenece el atributo (dominio), así como el tipo de dato de este (rango), tal como se muestra en la figura 22 con el ejemplo del atributo *baseDeDatos*.

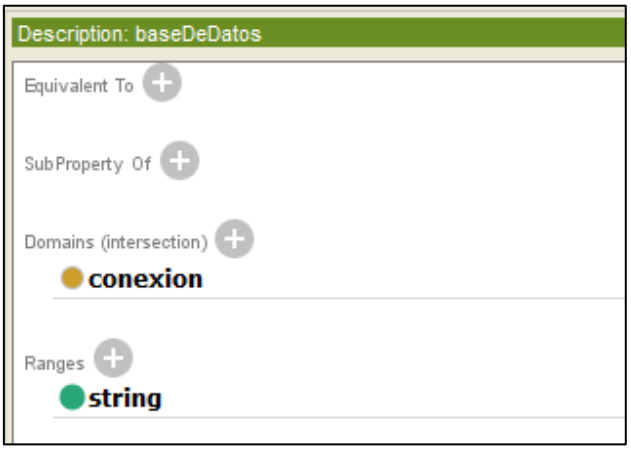

Fig. 22. Detalle del atributo baseDeDatos.

Una vez que se definieron conceptos, relaciones y atributos, se crearon instancias para formar la ontología. En la tabla 12 se detallan las instancias creadas y su uso.

Tabla 12. Instancias creadas en la ontología.

Nombre de la instancia	Concepto al que pertenece	Uso	Relaciones
camposComunes	notaComp	Permite indicar los campos que son comunes para todos los tipos de notas médicas.	ninguna
evolución	nota	Identifica el tipo de nota de evolución, los campos específicos de la misma, así como las gramáticas usadas en el RV.	otrosCampos → camposComunes seConectacon → conexPed
conexPed	conexión	Contiene la información para conexión a base de datos, tal como ubicación de la base, usuario, contraseña y nombre de las tablas donde se guardará la información.	usa → postgres
postgres	controlador	Indica el paquete y la clase (nombre calificado) del controlador para acceso a base de datos.	ninguna

3.9 Desarrollo del módulo de acceso a la ontología.

Como se mencionó en la sección 3.3, este módulo hace uso de la biblioteca *Jena* para acceder a la ontología creada en secciones anteriores. Además de la codificación necesaria en *Java*, también es necesario usar el lenguaje de consulta *SPARQL*. La estructura inicial de los tres métodos que contiene la clase *AccesoOnto.java* es similar, lo que cambia es la consulta realizada y el manejo de la información después de ejecutarse la consulta a la ontología. En el listado 9 se muestra la estructura en común que tienen los tres métodos para obtener información de la ontología.

Listado 9. Código para realizar una consulta en la ontología.

```
1  modOwl =
   ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM_MINI_RULE_INF);
2  modOwl.read("file:conexiones.owl");
3  sQuery = ""; //Asignación de la consulta según sea el caso
4  q = QueryFactory.create(sQuery);
5  qexec = QueryExecutionFactory.create(q, modOwl);
6  rs = qexec.execSelect();
```

En el listado 2 se mostraron los prefijos usados para todas las consultas en esta ontología. A continuación, se indica la estructura de las tres consultas usando estos mismos prefijos.

Consulta para obtener campos que la nota tiene en una tabla.

El método *obtenCampos* recibe una cadena de caracteres con el tipo de nota a buscar y devuelve un arreglo de cadenas con el nombre de todos los campos encontrados. En el listado 10 se muestra la consulta en lenguaje *SPARQL*, para obtener esta información.

Listado 10. Consulta para obtener los campos de una tabla.

```
1  SELECT ?x ?y WHERE {
2  ?x ito:tipoNota "Evolucion"^^xsd:string. ?x ito:campo ?y }
3  ORDER BY ?y
```

Consulta para obtener las gramáticas usadas por un tipo de nota.

El método *obtenGramaticas* recibe una cadena de caracteres con el tipo de nota a buscar y devuelve un arreglo de cadenas de caracteres con las gramáticas encontradas. En el listado 11 se muestra la consulta para obtener dicha información.

Listado 11. Consulta para obtener gramáticas usadas.

```
1  SELECT ?x ?y WHERE {
2  ?x ito:tipoNota "Evolucion"^^xsd:string. ?x ito:usaGramatica ?y }
3  ORDER BY ?y
```

Consulta para obtener información de conexión a una base de datos.

El método *obtenDatosConexion* recibe una cadena de caracteres con el tipo de nota a buscar y devuelve un arreglo de cadenas de caracteres con la información de conexión a base de datos. En el listado 12 se muestra esta consulta.

Listado 12. Consulta para obtener datos de conexión.

```
1  SELECT ?a ?b ?c ?d ?e ?f WHERE {
2  ?x ito:tipoNota "Evolucion"^^xsd:string.
3  ?x ito:seConectaCon ?y. ?y ito:usa ?z. ?z ito:rutaClase ?a.
4  ?y ito:nombreUsuario ?b. ?y ito:contrasena ?c.
5  ?y ito:baseDeDatos ?d. ?y ito:nombreTabla ?e.
6  ?y ito:tablaComun ?f}
```

3.10 Desarrollo del módulo de acceso a base de datos.

Para este módulo se reutilizó la clase denominada *DAO.java*, la cual fue proporcionada por la Maestra Beatriz Olivares Zepahua. A esta clase se le realizaron algunas modificaciones, las cuales se indican a continuación.

Inicialmente la clase contenía los campos para indicar:

- Nombre de usuario de tipo *String*.
- Contraseña de tipo *String*.
- URL de la base de datos de tipo *String*.
- Un objeto de conexión de tipo *java.sql.Connection*.

A esto cuatro campos se le agregaron dos:

- Tipo de base de datos, el cual es un campo de tipo *String* que contiene el nombre calificado de la clase que hace la función de controlador de base de datos.
- Nombre de la tabla en el que se guardará la información, de tipo *String*.

El constructor de la clase recibía tres argumentos, el URL, usuario y contraseña. El nuevo constructor de la clase recibe además el tipo de base de datos y el nombre de la tabla, en el listado 13 se muestra el código modificado.

Listado 13. Constructor de la clase DAO.java.

```
1  public DAO(String tipoBD, String sUsr, String sPwd, String sUrl,
   String tabla) throws Exception {
2    this.tipoBD = tipoBD;
3    this.sUsr = sUsr;
4    this.sPwd = sPwd;
5    this.sUrl = sUrl;
6    this.tabla = tabla;
7  }
```

Es importante destacar que la información con la que se alimenta el constructor de la clase al instanciarla proviene de la consulta hecha a la ontología y que fue descrita en la sección 3.9.

Por último, el método conectar se muestra en el listado 14. En la línea 4 se indica la carga del controlador para acceso a base de datos, como se observa, tomará el nombre calificado del gestor según sea el caso.

Listado 14. Método conectar.

```
1  public boolean conectar() throws Exception {
2   boolean bRet = false;
3   try {
4     Class.forName (this.tipoBD).newInstance();
5     oConexion = DriverManager.getConnection(sUrl, sUsr, sPwd);
6     bRet = true;
7   }catch(SQLException e) { throw e; }
8   return bRet;
9  }
```

Capítulo 4. Resultados.

En esta sección se indican los resultados obtenidos con la aplicación de reconocimiento de voz, en primera instancia los logros alcanzados en pruebas controladas, seguido de la aplicación con un caso de estudio.

Con base en el análisis de la NOM-004-SSA3-2012 del expediente clínico, se encontró que los diversos tipos de notas médicas cuentan con campos comunes tales como la talla, el peso, la edad del paciente, entre otros. Dicha información se reflejó en la ontología resultante.

Para el caso de la NOM-024-SSA3-2012 de los sistemas de información de registro electrónico para la salud, se encontró que se recomienda el uso del estándar HL7 y/o XML. El expediente clínico electrónico que se usó para realizar el caso de estudio cuenta con una base de datos de *PostgresQL*, por lo que se codificó el acceso a base de datos como se indica en el capítulo anterior. Además, la norma indica los datos mínimos para la identificación de una persona, y que se encuentran implementados en la base de datos del expediente clínico. Además, se cuenta también con los requerimientos mínimos de seguridad que marca la norma, los cuales son el uso de nombre de usuario y contraseña para tener acceso a la aplicación. Esta información se encuentra en la misma base de datos.

La especialidad médica usada para la captura de vocabulario fue pediatría, por lo que se buscaron ontologías de esta área, se encontró la ontología de términos pediátricos disponible en [45], sin embargo, estos términos se encuentran en idioma inglés, por lo que se optó por obtener los términos de [36]. En última instancia se incluyeron 546 términos de la especialidad médica, que en conjunto con los que ya se encontraban en el diccionario facilitado por [35] se cuenta con un total de 53,859 términos que es posible incluir en el reconocimiento de la aplicación siempre que se requieran.

Como en las notas médicas no todos los términos que se encuentran en el diccionario se usan, por ejemplo la palabra posgrado, y para no ver mermado el

porcentaje de acierto de reconocimiento, se optó por delimitar el número de términos que son reconocidos, en total la aplicación reconoce 386 síntomas y 67 enfermedades que un paciente es susceptible de presentar, además se cuenta con la funcionalidad de reconocer el dictado de la edad, frecuencia cardíaca, peso, talla, temperatura corporal y tensión arterial, los cuales son campos que tienen formato numérico.

En cuanto a la información usada para el entrenamiento del modelo acústico, no fue posible incluir el corpus VoCMex en el proceso pues no cumple con una de las características indicadas por *Sphinx*, la cual consta en incluir una porción de silencio de hasta 0.2 segundos al inicio y al final de cada archivo de audio. Por lo que se incluyó solo una hora de grabación que se realizó con lecturas del manual Harriet Lane de Pediatría, teniendo un total de 18 horas de audio para el entrenamiento del modelo acústico.

4.1. Resultados alcanzados en pruebas controladas.

Como primer punto a tratar, se encontró que el entrenamiento de un sistema de RV es una tarea muy laboriosa, la cual ocupó más tiempo de lo esperado. Inicialmente se planteó una duración de 8 semanas tan solo para la selección de audio y su respectiva transcripción fonética, pero en realidad este tiempo se duplicó.

Para calcular el porcentaje de acierto se tomó en cuenta la fórmula presentada en la sección 3.4. Se encontró que mientras se añadían enunciados a la gramática disminuía el porcentaje de acierto tal como se muestra en la tabla 13.

Tabla 13. Análisis de la gramática.

Núm. De enunciados	Porcentaje de acierto
1	88%
2	77%
3	59%
4	20%

Por lo anterior, fue necesario incluir varias gramáticas en la aplicación, cada una con solo un enunciado. La gramática cambia según sea la sección de la nota médica que se está dictando. Esto permite alcanzar hasta un 88% de acierto en RV.

Con respecto al uso de la ontología se tiene que, permite el uso de diversos gestores de bases de datos. Se realizaron pruebas con *PostgresQL* y *MySQL*, resultando en una comunicación satisfactoria entre la aplicación y ambos tipos de bases de datos. La estructura de la ontología permite agregar una nueva conexión y de ser necesario un nuevo gestor de base de datos. El uso del concepto *notaComp* permite indicar los campos que son comunes para todas las notas médicas, lo que da lugar a que cuando se agrega a la ontología un nuevo tipo de nota médica solo sea necesario indicar los campos específicos de la misma.

4.2. Caso de estudio: Prueba de reconocimiento de voz en vivo con personal médico.

Se realizó el caso de estudio con apoyo de un médico colaborador del Hospital y se alcanzó un porcentaje de acierto de 78.4%, proporción alta tomando en cuenta que no se tiene el software entrenado con la voz del doctor. Para asegurar un alto porcentaje de acierto se tienen dos opciones:

1) Entrenar el software con por los menos 20 minutos de audio del usuario que usará la aplicación.
2) Contar con un modelo acústico independiente del hablante, para esto es necesario entrenarlo con más de 50 personas y contar con al menos 200 horas de audio.

Algunos de los enunciados utilizados para realizar la prueba fueron los que se listan a continuación:

- Paciente de 50 años de edad (cambiando el número de edad en varias ocasiones).
- Hipertensión arterial, rinitis alérgica, sobrepeso, enfermedad de Kawasaki.

- Dermatitis atópica, diabetes tipo 2, síndrome de ansiedad
- Ojos infectados, tos, fiebre moderada, dolor abdominal leve.
- Fiebre alta, dolor constante en los dientes.
- Para el caso de los diversos campos numéricos se dejó a consideración del médico los números a dictar.

Se dictaron 102 palabras de las cuales, para el caso de los campos numéricos se obtuvo un porcentaje de acierto del 66.7%, mientras que para el caso de los campos de texto se alcanzó un 85.7%. El cálculo del porcentaje toma en cuenta que se dictaron 39 palabras para el caso de los campos numéricos y 63 términos para el caso de la descripción del paciente, sus padecimientos y síntomas asociados.

Además de la prueba de reconocimiento de voz se realizaron unas preguntas al doctor participante para tener registro de sus impresiones con el uso del software. Los resultados de estos cuestionamientos se muestran a continuación usando para las respuestas una escala de *Likert* con cinco posibles contestaciones.

1. **La manera de interactuar con el software es amigable.**

Totalmente de acuerdo	**De acuerdo**	Indiferente	En desacuerdo	Totalmente en desacuerdo

2. **La precisión del reconocimiento es aceptable para su labor cotidiana.**

Totalmente de acuerdo	De acuerdo	**Indiferente**	En desacuerdo	Totalmente en desacuerdo

3. **Recomendaría a mis colegas el uso de esta aplicación para agilizar su labor en la escritura de notas médicas.**

Totalmente de acuerdo	**De acuerdo**	Indiferente	En desacuerdo	Totalmente en desacuerdo

Capítulo 5. Conclusiones y recomendaciones.

En esta sección se indican las conclusiones obtenidas con el desarrollo de la tesis, así como las recomendaciones para trabajos futuros.

Conclusiones.

1. La creación de un modelo acústico robusto que permita un alto porcentaje de acierto en el RV, es una tarea muy amplia en cuanto a los tiempos que se necesitan pues la teoría indica que se debe tener una muestra de alrededor de 50 personas entre hombres y mujeres de diversas edades, además deben tenerse unas 200 horas de estas grabaciones dentro de varios ambientes tales como llamadas por celular, lecturas de textos, habla espontánea, entre otros.
2. La versión 3 de *Sphinx* que se usó para el entrenamiento del modelo acústico cuenta con la peculiaridad de funcionar sin errores únicamente en *Ubuntu Linux* versión 12, lo cual limita las opciones de desarrollo. Sin embargo, una vez obtenido el modelo entrenado, es posible portarlo a la versión 4 para *Java* sin ningún problema de compatibilidad.
3. El uso de la biblioteca *Sphinx* para *Java* permite portar la aplicación a diversos sistemas operativos, ya sea *Windows*, *Linux* o *Macintosh*.
4. Las restricciones de *Sphinx* en cuanto a las características que requieren cumplir los archivos de audio no permitieron usar el corpus VoCMex. Es necesario tener en cuenta dichas características antes de comenzar a crear el *corpus* que se usará para el entrenamiento.
5. Es posible aumentar el número de palabras que identifica la aplicación solo agregándolas al diccionario con su respectiva transcripción fonética, cabe mencionar que no es necesario que la palabra se encuentre en el *corpus* de entrenamiento, siempre y cuando se encuentren los fonemas que la conforman.

6. El uso de *METHONTOLOGY* para la conceptualización de la ontología permite hacer cambios y revisiones en ella antes de iniciar su creación en la herramienta de soporte. Esto permite ahorrar esfuerzo y tiempo, ya que si no se siguiera la metodología se corre el riesgo de eliminar el trabajo hecho en la herramienta para replantear la ontología y crear las instancias correspondientes.
7. El uso de una ontología permitió realizar la conexión a dos bases de datos distintas utilizando el mismo código para ambos gestores, esto fue posible tomando de la ontología la información como el nombre calificado de la clase perteneciente al gestor, así como los datos de acceso a la base de datos correspondiente, estos datos son guardados en campos de tipo cadena dentro de la clase de acceso a base de datos. Lo anterior permite agregar más gestores a la aplicación cuando sea necesario solo creando las instancias correspondientes en la ontología y añadiendo la biblioteca correspondiente al gestor en la aplicación.

Recomendaciones.

1. Hasta el momento en que se escribió este documento no se encontró un *corpus* gratuito lo suficientemente amplio para cumplir con las características mencionadas en el punto 1 de las conclusiones, por lo que la creación del mismo sería una aportación relevante para los trabajos que empleen el RV en español de México.
2. Es posible construir una herramienta gratuita que permita realizar los pasos para la construcción del corpus de manera automática, pues existen en el mercado instrumentos para este fin, sin embargo, los pasos deben ser realizados semiautomáticamente.
3. Actualmente se encuentra en versión pre-alfa la biblioteca de *CMU-Sphinx* 5, que cuenta con un módulo para el reconocimiento de voz mediante grabaciones de audio, es decir, leyendo archivos previamente grabados por el usuario. En un futuro cuando esta versión se encuentre libre de errores, será

posible implementar un módulo en la aplicación que permita grabar en audio las indicaciones de los médicos para posteriormente realizar el proceso de reconocimiento.

4. Otro punto es el uso de *CMU-Sphinx* en dispositivos móviles, para esto se cuenta actualmente con *PocketSphinx* y que está en fase de pruebas por la comunidad de desarrolladores en equipos con el sistema Android. Esta versión es otra oportunidad para hacer crecer la gama de dispositivos en los que se implemente el registro de notas médicas con apoyo del RV.
5. La creación de un algoritmo que permita adaptarse al usuario está propuesta en varias fuentes, por lo que la implementación del mismo es otra área de oportunidad para logar que la aplicación vaya aumentando su porcentaje de acierto mientras más se utiliza.
6. La base de datos usada para el caso de estudio usa el gestor *PostgreSQL*, sin embargo, es posible crear un módulo de comunicación con dispositivos que soporten el estándar HL7 cuando sea necesario. Cabe mencionar que también existen bibliotecas gratuitas para este fin.

Glosario de términos.

Corpus. Nombre que se le da a las bases de datos de audio que están diseñadas para el uso en entrenamiento de sistemas de reconocimiento de voz. Tienen la característica de contar con los enunciados textuales que contiene cada archivo de audio, proceso conocido como etiquetado.

Escala de Likert. Es una escala psicométrica comúnmente utilizada en cuestionarios y es la escala de uso más amplio en encuestas para la investigación.

Hertz. Unidad de frecuencia del sistema internacional de unidades.

JDBC (*Java Data Base Connectivity*) O conectividad de base de datos en Java es una *API* (Interfaz de programación de aplicaciones) que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede, siempre que se utilice un tipo de base de datos *SQL*.

Sprint. Nombre que recibe el bloque de tiempo de aproximadamente un mes en la metodología ágil *Scrum,* durante el cual se crea un incremento del producto utilizable y potencialmente desplegable.

Vocal tónica. Es la vocal que tiene mayor intensidad en una palabra.

WAV. Formato de archivo de audio sin compresión de datos, admite archivos mono y estéreo a diversas frecuencias y velocidades de muestreo.

Productos académicos.

Mario E. Villalobos-Gallegos, Beatriz A. Olivares-Zepahua, Celia Romero-Torres, Gustavo Peláez-Camarena, Luis A. Reyes-Hernández. “Generación de un prototipo como prueba de concepto para una aplicación de registro de información en el expediente clínico electrónico de un sanatorio mediante el reconocimiento de voz”. Artículo presentado en el Encuentro Nacional de Ciencias de la Computación ENC 2015, el día 7 de octubre del 2015.

Ing. Mario E. Villalobos Gallegos, M. C. Beatriz A. Olivares Zepahua, Dr. Abel Herrera Camacho. “Construcción de modelo acústico en español de México para el área pediátrica y su implementación con CMU-SPHINX”. Investigación y desarrollo en robótica y computación, memoria de congreso, mayo del 2016, pp. 228-232, ISBN 978-607-97128-2-2. Artículo presentado en el III Congreso Internacional de Robótica y Computación CIRC 2016, el día 4 de mayo del 2016.

Fuentes de información.

[1] T. Gruber. [En línea]. Available: http://tomgruber.org/writing/ontology-definition-2007.htm. [Último acceso: 23 febrero 2015].

[2] F. y. W. J. Faiez, Ontology theory, management and design: Advanced tools and models, New York: Information Science Reference, 2010. pp. 3-4.

[3] G. d. l. E. U. d. Norteamérica, «What is an electronic health record (EHR)?,» [En línea]. Available: http://www.healthit.gov/providers-professionals/faqs/what-electronic-health-record-ehr. [Último acceso: 15 diciembre 2014].

[4] G. d. México, «Secretaria de Economía,» [En línea]. Available: http://www.economia.gob.mx/comunidad-negocios/competitividad-normatividad/normalizacion/catalogo-mexicano-de-normas. [Último acceso: 15 diciembre 2014].

[5] N. O. M. NOM-004-SSA3-2012, «Diario Oficial de la Federación,» [En línea]. Available: http://dof.gob.mx/nota_detalle_popup.php?codigo=5272787. [Último acceso: 15 diciembre 2014].

[6] A. e.-H. Workgroup, «Speech Recognition in the Electronic Health Record,» [En línea]. Available: library.ahima.org/xpedio/groups/public/documents/ahima/bok1_022107.hcsp?dDocName=bok1_022107. [Último acceso: 16 diciembre 2014].

[7] M. Fernandez-López, «Overview of methodologies for building ontologies,» *Proceedings of the IJCAI-99 workshop on ontologies and problem-solving methods,* pp. 1-13, 1999.

[8] A. y. H. J. Dean, Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL, Elsevier, 2011.

[9] W. L. W. y. B. M. Wong, Ontology learning and knowledge discovery using the web: Challenges and recent advances, IGI Global, 2011.

[10] A. Adami, «Automatic speech recognition: From the beginning to the Portuguese language,» de *International Conference on Computational Processing of Portuguese*, Porto Alegre, Brasil, 2010. pp. 6.

[11] H. y. L. K. Xuedong, «On speaker-independent, speaker-dependent, and speaker-adaptive speech recognition,» *IEEE Transactions on speech and audio proceesing,* vol. 1, nº 2, pp. 150-157, 1993 DOI: 10.1109/89.222875.

[12] S. i. VÓCALI, «INVOX Medical Dictation,» [En línea]. Available: http://www.vocali.net/products/invox-medical-dictation. [Último acceso: 20 enero 2015].

[13] OpenClinical, «Ontologies,» [En línea]. Available: http://www.openclinical.org/ontologies.html. [Último acceso: 20 enero 2015].

[14] A. Shuman, «Speed EHR documentation with voice recognition software,» Contemporary Pediatrics, Junio 2014. [En línea]. Available: http://contemporarypediatrics.modernmedicine.com/contemporary-pediatrics/content/tags/contemporary-pediatrics-peds-v20/speed-ehr-documentation-voice-

?page=full. [Último acceso: 26 enero 2015].

[15] K. Terry, «Speech recognition booms as EHR adoption grows,» Information Week, 2013. [En línea]. Available: http://www.informationweek.com/healthcare/electronic-health-records/speech-recognition-booms-as-ehr-adoption-grows/d/d-id/1108842?. [Último acceso: 26 enero 2015].

[16] Nuance. [En línea]. Available: http://www.nuance.com/for-healthcare/by-solutions/speech-enabled-ehr/index.htm. [Último acceso: 25 enero 2015].

[17] Capterra, «Top medical transcription software,» [En línea]. Available: http://www.capterra.com/medical-transcription-software/. [Último acceso: 25 enero 2015].

[18] B. Lars-Erik, «Processing electronic medical records: Ontology-Driven information extraction and structuring in the clinical domain,» 2012. [En línea]. Available: https://www.duo.uio.no/handle/10852/34162. [Último acceso: 25 enero 2015].

[19] P. O. D. y. W. P. Plastiras, «An ontology-driven information model for interoperability of personal and electronic health records,» *The Sixth International Conference on eHealth, Telemedicine and Social Medicine,* pp. 130-133, 2014.

[20] W. e. a. Ceusters, «Ontology-based integration of medical coding systems and electronic patient records,» 2044. [En línea]. Available: http://ontology.buffalo.edu/medo/CodingAndEHCR.pdf. [Último acceso: 26 enero 2015].

[21] R. P. X. Lozano-Rubí, «OWLing clinical data repositories with the ontology web language,» JMIR medical informatics, 2014. [En línea]. [Último acceso: 26 enero 2015].

[22] M. Koivikko, «Improvement of report workflow and productivity using speech recognition: A follow-up study,» *Journal of digital imaging,* vol. 21, nº 4, pp. 378-382. DOI: 10.1007/s10278-008-9121-4, 2008.

[23] M. Nagy, «Voice-controlled data entry in dental electronic health record,» *Studies in health technology and informatics,* vol. 136, pp. 529-534, 2008.

[24] M. Ghazisaeedi, «The design of a reconstructive hand surgery text database based on speech recognition system in Iran,» *International Journal of Biomedical Science and Engineering,* vol. 2, nº 3, pp. 17-22. DOI: 10.11648/j.ijbse.20140203.11, 2014.

[25] R. y. Y. A. Hoyt, «Lessons learned from implementation of voice recognition for documentation in the military electronic health record system,» *Perspectives in health information management,* vol. 1, nº 7, pp. 1-9, 2010.

[26] T. y. K. R. Way, «Achieving acceptabe accuracy in a low-cost, assistive note-taking, speech transcription system,» *Proceedings of the IASTED International Conference on Telehealth and Assistive Technologies,* pp. 1-6, 2008.

[27] W. y. F. W. Xinxin, «The application of speech recognition in radiology information system,» *International Conference on Biomedical Engineering and Computer Sciences,* pp. 1-3. DOI: 10.1109/ICBECS.2010.5462425, 2010.

[28] I. e. a. Berges, «Semantic interoperability of clinical data,» *Proceedings of the First International Workshop on Model-Driven Interoperability,* pp. 10-14. DOI:10.1145/1866272.1866275, 2010.

[29] A. y. R. W. Sari, «Archetype sub-ontology: Improving constraint-based clinical knowledge model in electronic health records,» *Knowledge-Based Systems,* nº 26, pp. 75-85. DOI:10.1016/j.knosys.2011.07.004, 2012.

[30] e. a. Taweel, «Service and model-driven dynamic integration of health data,» *Proceedings of the first international workshop on managing interoperability and complexity in health systems,* pp. 11-17. DOI: 10.1145/2064747.2064752, 2011.

[31] U. d. Standford, «protégé,» Universidad de Stanford, [En línea]. Available: http://protege.stanford.edu/about.php. [Último acceso: 8 marzo 2016].

[32] K. y. S. J. Schwaber, «Scrum guides,» [En línea]. Available: http://www.scrumguides.org/. [Último acceso: 16 noviembre 2015].

[33] W3C, «World Wide Web Consortium,» 26 marzo 2013. [En línea]. Available: http://www.w3.org/TR/rdf-sparql-query/. [Último acceso: 15 noviembre 2015].

[34] C. M. University, «Robust Group Tutorial,» [En línea]. Available: http://www.speech.cs.cmu.edu/sphinx/tutorial.html. [Último acceso: 16 noviembre 2015].

[35] C. y. H. J. Mena, «Proyecto CIEMPIESS-UNAM,» UNAM, [En línea]. Available: http://www.ciempiess.org/about. [Último acceso: 16 noviembre 2015].

[36] M. y. A. K. Tschudy, Manual Harriet Lane de pediatría, Barcelona: Elsevier, 2013.

[37] A. Team, «Audacity,» [En línea]. Available: http://audacityteam.org/?lang=es. [Último acceso: 16 noviembre 2015].

[38] U. A. d. Barcelona, «Cconceptos básicos de fonética y fonología,» [En línea]. Available: http://liceu.uab.es/~joaquim/phonetics/fon_transcr/transcripcion_fonetica.html#definicion_transcripcion_fonetica. [Último acceso: 16 noviembre 2015].

[39] J. Cuétara-Priede, *Fonética de la ciudad de México. Aportaciones desde las tecnologías del habla.,* México: Universidad Nacional Autónoma de México, 2004.

[40] «LiClipse,» Brainwy Software Ltda., [En línea]. Available: http://www.liclipse.com/. [Último acceso: 4 febrero 2016].

[41] CMU-SPHINX, «sourceforge.net,» Carnegie Mellon University, [En línea]. Available: http://cmusphinx.sourceforge.net/wiki/tutorialam. [Último acceso: 3 febrero 2015].

[42] «SoX Sound eXchange,» [En línea]. Available: http://sox.sourceforge.net/. [Último acceso: 4 febrero 2016].

[43] e. a. Artín Olguin, «VoCMex: A voice corpus in Mexican Spanish for research in speaker recognition,» *INTERNATIONAL JOURNAL OF SPEECH TECHNOLOGY. DOI 10.1007/s10772-012-9183-z,* pp. 1-9, 2013.

[44] R. Srivastava, «BioPortal,» [En línea]. Available: http://bioportal.bioontology.org/ontologies/PEDTERM. [Último acceso: 18 abril 2016].