Universidad Politécnica del Estado de Morelos

Universidad Santo Tomás – Seccional Tunja

Sistema Experto para posibles portadores de la enfermedad de Chagas

TESINA

Que para obtener el título de:

INGENIERO DE SISTEMAS

P r e s e n t a

JOHN LEANDRO MEJÍA CASTRO

Directores de Tesina

Dra. Sandra Elizabeth León Sosa

Ing. Edgar Medina Sánchez

Jiutepec, Morelos Diciembre 2018

Oficio de autorización de impresión

Upemor
Universidad Politécnica

INGENIERÍA EN INFORMÁTICA

Jiutepec, Morelos a 12 de diciembre de 2018

AUTORIZACIÓN DE IMPRESIÓN DE TESINA

Los abajo firmantes, miembros del jurado para la evaluación del proyecto de estadía del alumno: **John Leandro Mejía Castro** manifiestan que después de haber revisado su tesina titulada **"Sistema experto para posibles portadores de la enfermedad de Chagas"**, realizada bajo la dirección del Dra. Sandra Elizabeth León Sosa y el Ing. Edgar Medina Sánchez de la Universidad Santo Tomás Colombia - Seccional Tunja, el trabajo se ACEPTA para proceder a su impresión.

ATENTAMENTE

Presidente

Dra. Sandra Elizabeth León Sosa
Cédula Profesional: 11186188

Secretario

Dr. Cornelio Morales Morales
Cédula Profesional: 9336584

Vocal I

Dra. Irma Yazmín Hernández Báez
Cédula Profesional: 8508209

Dr. Cornelio Morales Morales
Director Académico
Cédula Profesional: 9336584
Certificado que las firmas anteriores corresponden al jurado designado para la evaluación final de estadía.

Dedicatoria

Este trabajo está dedicado a Beatriz Robles de Mejía (03/01/1941 – 16/09/2017) mejor profesora del municipio de Chitaraque Boyacá en el año 2006, quien dedicó cuarenta y dos años de su vida a la docencia, convirtiéndose así en un ejemplo de vida para mí. Siendo apenas un niño me enseñó a sumar, restar y multiplicar, siempre le importó que sus nietos fueran personas cultas, queriendo verlos profesionales y animándonos siempre a ello.

Hoy dedico ésta tesina a una madre y abuela que con su ejemplo y actos me hacía sentir orgulloso, para quien la enseñanza era una de sus principales virtudes sin importarle tener grandes cantidades de dinero, únicamente importándole su esposo o patrón como solía llamarle y la educación de los diversos niños que pasaron por sus manos en los cuarenta y dos años de enseñanza, por esto dedico este trabajo y triunfo a esa mujer que le importaba la educación, quien sabía que lograría mi título como ingeniero y que se sentía orgullosa de ello, por lo anterior y mucho más mil gracias abuelita.

Agradecimientos

Existe un ser supremo que en mi camino ha sido como la luz de un faro el cual me guía para cumplir con mis diversas metas, por ello agradezco primeramente a Dios quien hace que los tiempos sean perfectos y que las cosas sucedan en el momento indicado, siempre cuidando de mi familia y mostrándome que me puedo superar en los retos que la vida me pone.

En segundo lugar agradezco a mis padres John Leandro Mejía Robles y Consuelo Castro Güiza quienes me inculcaron de niño principios que se están perdiendo en esta sociedad, quienes con trabajo, lágrimas y mucho esfuerzo me llevaron hasta este punto de mi vida dándome consejos desde la niñez los cuales en su momento parecían ser duros pero que al pasar el tiempo me formarían como persona. Gracias por dejar a un lado sus metas y por ayudarme a cumplir las mías, los amo.

En tercer lugar agradezco a mis abuelitos Reynaldo Mejía Barón, Beatriz Robles de Mejía, Cristo Castro León y Rosalba Gúiza viuda de Castro por darle vida a mis padres, también valores y principios que posteriormente pasarían a mí, son el eje central de mi familia pues con sus consejos y ejemplo me hacen ser cada vez mejor.

Resumen

En el presente proyecto se desarrolla un sistema web experto que pretende analizar desde una perspectiva social la problemática que existe con la enfermedad de Chagas pues la mejor forma de hacer un control a esta enfermedad es la prevención, por ende se construye un sistema que influya en generar una mayor prevención.

Para el desarrollo de este proyecto se hace fundamental aplicar un proceso que lleva tiempo y que puede ser complejo, este se conoce como proceso de extracción de conocimiento. En este proyecto se llevan a cabo todas las fases del proceso descrito anteriormente, fases que son detalladas y puestas por escrito en el presente documento, donde se evidencia cual es el grado de eficacia del algoritmo implementado para determinar la correcta clasificación de los posibles portadores de la enfermedad de Chagas.

El sistema experto es codificado en Java Web, utilizando la plataforma Weka para el análisis de la información, así mismo se hace uso de una API que implementa un algoritmo clasificador bajo la plataforma de Java Web y generar los reportes correspondientes a las zonas que el usuario del sistema experto desee analizar.

Tabla de contenido

Lista de figuras

Lista de tablas

Capítulo 1. INTRODUCCIÓN

En este capítulo, se presentarán los puntos importantes donde se establece cuál es el objetivo principal del desarrollo de la tesina, además de los alcances, el resultado esperado, el resultado obtenido, la metodología utilizada para el desarrollo y cualquier otra información relevante del proyecto.

1.1. Antecedentes

Los conocimientos actuales que se tienen sobre el Chagas se basan en los descubrimientos efectuados por Carlos Chagas(1909) y Salvador Mazza (1931), en 1909 en el Brasil Carlos Chagas define al germen causante del Chagas como Schizotrypanum cruzi, posteriormente Emmanuel Dias Profundizó en el estudio morfológico y biológico del germen causante y Rodolfo Talice y Cecilio Romaña terminaron por aportar conocimientos.

Para el año 1928 en el estado de Veracruz (México) Hoffman C. realiza el descubrimiento del insecto Triatoma dimitia, el cual es considerado uno de los vectores más importantes en la propagación de Chagas, en 1967 Cuartero observó y detectó en el estado de Jalisco las primeras evidencias de la enfermedad de Chagas en la sangre de 5 niños, debido a condiciones socioeconómicas, climatológicas y ecológicas México se convierte en uno de los principales focos de transmisión de esta enfermedad. Entre 1997 y el 2001 se lleva a cabo un estudio epidemiológico en nueve jurisdicciones el estado de Veracruz, en el cual sus condiciones permiten que se convierta en foco de esta enfermedad, En dicho estudio se procede a identificar el hábitat del insecto Triatoma dimitia donde se hace una clasificación de las viviendas que facilitan la infestación de este insecto. Los resultados del estudio demuestran que la infección se estaba transmitiendo de forma activa en las jurisdicciones de Panúco, Tuxpan, Córdoba, Veracruz y San Andrés Tuxtla donde se presentaron varios casos incluso en menores de 18 años de edad y se encontraron factores que facilitan la permanencia de los triatominos en las viviendas, tales como, la convivencia con los perros y los gatos los cuales representan una fuente de alimentación para los triatomas, el material de construcción de la vivienda principalmente en los techos; lámina de Zinc, carrizo/bambú y teja o lámina de cartón, para los muros los materiales principales fueron el barro, la madera y en el piso la tierra.

Después del paludismo la enfermedad de Chagas es el padecimiento más serio y extendido entre los habitantes de América, región donde se estimaba para el año 2010 causaba 45 mil muertes anuales.

1.2. Definición del problema

Actualmente, no se tiene un sistema que permita determinar qué sector poblacional puede ser portador de la infección del Chagas, esto conlleva a que el proceso de análisis y detección de los posibles portadores de esta infección sea tardío, ocasionando molestias en los pacientes y afectando otros procesos como por ejemplo la transfusión sanguínea; puesto que la norma Oficial Mexicana NOM 003-SSA-1993, en las pruebas de sangre no establece la prueba de tamizaje universal como obligatoria, convirtiéndose así en la segunda vía de transmisión más importante de dicha infección.

1.3. Panorama general del proyecto

El desarrollo de este sistema, permitirá mediante el uso de algoritmos determinar quién puede ser portador de la infección de Chagas, detectando a la población que cuente con las características apropiadas para ser candidatos a portar esta enfermedad. Para llevar a cabo el sistema, se tendrá en cuenta lo siguiente:

- Generar la base de conocimiento para los posibles portadores basándose en los datos reales recolectados por el censo del 19 de septiembre del 2017.
- Implementar algoritmos clasificadores que determinen los posibles portadores de la enfermedad de Chagas.
- Generar reportes de las zonas con índices de posibles portadores de Chagas.

1.4. Objetivos

Objetivo general

- Desarrollar un Sistema experto para posibles portadores de la enfermedad de Chagas.

Objetivo específicos

- Generar la base de conocimiento para los posibles portadores de la enfermedad de chagas.
- Implementar dos algoritmos clasificadores que determinen los posibles portadores de la enfermedad de Chagas.
- Generar reportes de las zonas con índices de posibles portadores de Chagas.

1.5. Justificación

Debido a la falta de existencia de un sistema que permita determinar qué sector poblacional puede ser portador de la infección del Chagas, el proceso de analizar quién puede portar esta infección se hace tardío, por ello al analizar este problema desde una perspectiva social, podrá realizarse una mayor prevención al desarrollar un sistema experto que permita mediante el uso de algoritmos determinar quién puede ser portador de esta infección. Para ello se debe contar con la base de conocimientos adecuada y desarrollarse en Java Web mediante el uso de una API que permita conectar este con la herramienta de Weka.

1.6. Alcances y limitaciones

En esta sección se definen los alcances y limitaciones del proyecto.

Alcances

- Generar la base de conocimiento basándose en los datos reales recolectados por el censo del 19 de septiembre del 2017 para analizar los posibles portadores.
- Implementar dos algoritmos clasificadores, que determinen los posibles portadores de la enfermedad de Chagas.
- Generar reportes respecto a la base de conocimiento real de zonas con índice de posibles portadores de Chagas.

Limitaciones

- El periodo de desarrollo es de máximo tres meses.
- La información proporcionada por parte del censo del 19 de septiembre de 2017 debe ser la adecuada para generar una buena base de conocimientos.
- El sistema experto solo dará un pronóstico de qué tan posible es que la persona esté contagiada con Chagas, pero para poder verificar esto deberá realizársele el correspondiente análisis médico.
- Sólo se podrá tener acceso al sistema mediante un navegador web.

1.7. Metodología

La metodología de desarrollo de software que se elige para llevar a cabo este sistema experto es la METODOLOGÍA INCREMENTAL. Esta metodología se elige por la forma en que se debe desarrollar el software pues se adapta al ciclo de vida de este, haciendo uso de diversas técnicas, herramientas y métodos para el desarrollo.

Debido a que una de las grandes limitaciones es el periodo de tiempo con el que se cuenta para el desarrollo del sistema, la metodología incremental se convierte en una gran herramienta pues mediante ésta se puede llevar a cabo el trabajo de forma rápida y en etapas tempranas, también es un modelo flexible lo que implica que se pueden llevar a cabo diversas iteraciones durante todo el proceso sin verse afectadas las etapas de desarrollo, permitiendo que sea más fácil gestionar los riesgos que se puedan presentar.

En la Figura 1 se muestra la metodología Incremental y en la Tabla 1 se presenta cuáles son los incrementos de esta metodología.

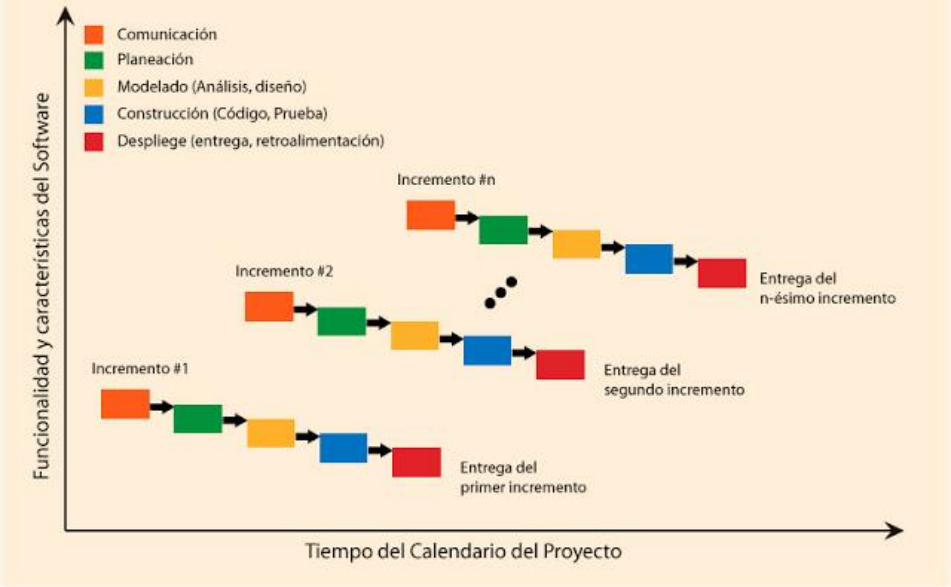

Figura 1. Metodología Incremental
Fuente: Pressman Roger S. (2005). Ingeniería del Software: Un enfoque Práctico.

Tabla 1. Metodología desarrollo de software - Metodología Incremental

Metodología Incremental	
Incremento	**Requisitos**
1	**FN.1 Base de Conocimientos.**
2	**FN.2 Presentación de datos almacenados.**
3	**FN.3 Generación de reporte.**
4	**FN.4 Comparación de los algoritmos clasificadores.**

Fuente: John Leandro Mejía Castro.

1.8. Organización de la tesina

En el siguiente apartado, se presentará una breve descripción de cada uno de los apartados que forman parte de esta tesina.

Capítulo 1. Introducción
En este capítulo se muestran las características principales del proyecto o lo que define a este como el panorama general, objetivo. Justificación, antecedentes, alcances, límites y metodología para el desarrollo del sistema.

Capítulo 2. Análisis de requisitos
Se detallan cada uno de los diferentes requisitos que se necesitan plantear para el desarrollo del sistema además de la definición de sus características como la importancia y validez.

Capítulo 3. Diseño del sistema
Se presenta el diseño de los requisitos funcionales del sistema, el diseño de las interfaces que se le mostrarán al usuario, el diseño de la base de datos, un diagrama entidad-relación obteniéndolo del capítulo anterior.

Capítulo 4. Implementación
Aquí se procede a mostrar la implementación de los diseños generados anteriormente, también la descripción de la arquitectura física, tecnologías empleadas, proceso de extracción de conocimiento y documentación del código fuente.

Capítulo 5. Pruebas
El objetivo principal de este capítulo es mostrar el correcto funcionamiento del sistema, esto se realizará de acuerdo a diferentes pruebas.

Capítulo 6. Conclusiones
Se realizan todas las conclusiones obtenidas de acuerdo a cómo se desarrolla este sistema, los problemas que se encontraron durante el desarrollo y planes futuros para el sistema.

Capítulo 2. ANÁLISIS DE REQUISITOS

En el siguiente capítulo, se muestra la descripción detallada de cada uno de los requisitos que necesita el Sistema experto para posibles portadores de la enfermedad de Chagas con el fin de definir la funcionalidad del proyecto.

2.1. Introducción

En la elicitación de requisitos del Sistema Experto para posibles portadores de la enfermedad de Chagas se hace fundamental aplicar las técnicas conocidas como *análisis de documentación* y *entrevistas*.

El análisis de documentación es una técnica fundamental para la elaboración del Sistema Experto, pues la información recolectada por parte del censo del 19 de septiembre del año 2017 es de carácter verídico, lo que permite llevar a cabo un análisis de la información que genere la base de conocimientos adecuada para el tratamiento de los datos.

La entrevista es otra técnica que se convierte en una herramienta fundamental para el desarrollo del Sistema Experto, brinda información detallada sobre los requisitos. La mayoría de las veces se hace entre dos personas, una de éstas es quien necesita adquirir la información detallada y precisa y la otra persona es quien puede suministrar dicha información. En esta técnica se hace importante que la persona que adquiere la información insista hasta comprender el requisito de quien suministra la información y tenga conocimientos generales sobre el sistema actual para la detección de posibles portadores de la enfermad de Chagas. Se optó por esta técnica para el desarrollo del presente documento ver Anexo I.

2.2. Definiciones y Acrónimos

En este apartado se incluirá una lista de las definiciones y una lista de los acrónimos utilizados en el documento.

Tabla 2. Definiciones y acrónimos.

Acrónimo	Definición
RD.	Restricción de diseño
FN.	Requisito funcional nominal
FF.	Requisito funcional no nominal
IN.	Requisitos de interfaz
CA.	Requisitos de calidad
EV.	Requisitos de evolución
PR.	Requisitos de Proyecto
OS.	Requisitos de soporte

Fuente: John Leandro Mejía Castro.

2.3. Restricciones de diseño

En esta sección, se presenta cada una de las restricciones de diseño que el sistema tendrá que son las que pueden llegar a limitar su desarrollo dependiendo de los requisitos.

RD.1 Sólo se podrá tener acceso al sistema mediante un navegador web**.**

RD.2 Mediante el uso de plantillas Bootstrap se llevan a cabo las interfaces de usuario.

2.4. Requisitos funcionales

Conjunto de requisitos que reflejan las funciones que debe prestar el sistema. Se clasifican en las siguientes subsecciones:

2.4.1. Requisitos funcionales nominales

FN.1 Base de Conocimientos.

Descripción: El sistema debe contar con los datos adecuados suministrados por el censo del 19 de septiembre del 2017 para tener una base de conocimientos verídica.

Importancia: Esencial.

Validez:

- Medible: Se debe depurar toda la información suministrada de los posibles portadores de Chagas, también crear una base de datos MySQL con las tablas y atributos necesarios, que permitan posteriormente realizar una migración a esta, teniendo en cuenta los datos del 19 de septiembre del 2017.
- Alcanzable: Se debe obtener un archivo.arff con el 80% de los datos depurados e igualmente un archivo .arff para el restante 20% de estos, también realizar la migración de estos datos a la base de datos MYSQL, también se debe implementar el proceso de extracción de conocimiento.
- Relevante: Al cumplir este requisito, se genera la base de conocimientos adecuada para poder hacer el análisis, aprendizaje y respectivas consultas de la información.

FN.2 Presentación de datos almacenados.

Descripción: El Sistema debe permitir al usuario conocer la información que será procesada para llevar a cabo el análisis de los posibles portadores de Chagas.

Importancia: Esencial.

Validez:

- Medible: El Sistema debe consultar la base de datos y de forma entendible e intuitiva presentar los datos que podrán ser analizados por el sistema.
- Alcanzable: El Sistema debe realizar la consulta a la base de datos para presentar en una tabla de forma ordenada la información a la que el usuario podría realizar el respectivo análisis.
- Relevante: Al cumplir este requisito, se presenta al usuario la información contenida en la base de datos de forma ordenada de tal manera que este tenga conocimiento de que información es la que desea analizar.

FN.3 Generación de reporte.

Descripción: Es necesario que el Sistema Experto genere reporte de los Municipios con posible índice de portadores de Chagas, haciendo uso de gráficas que representen quienes pueden ser portadores de esta enfermedad.

Importancia: Esencial.
Validez:

- Medible: El Sistema Experto debe hacer un análisis respecto a la información que desea consultar el usuario por municipio, generando

así un reporte con los datos respectivos de éstos y mediante el uso de gráficas representar el porcentaje de posibles portadores que puedan portar Chagas.

- Alcanzable: El sistema debe llevar a cabo el correspondiente análisis de acuerdo al municipio suministrado por el usuario, generando así estadísticas que le permitan a este conocer información de posibles portadores de Chagas.
- Relevante: Es fundamental la generación del reporte pues presenta al usuario de acuerdo al municipio que este seleccione el pronóstico de los posibles portadores de Chagas.

FN.4 Comparación de algoritmos.

Descripción: Es necesario efectuar una comparación de los algoritmos clasificadores, observando los modelos y la eficacia de entrenamiento que se generan con estos.

Importancia: Esencial.
Validez:

- Medible: Se deben observar los modelos o árboles que generan los algoritmos clasificadores viendo de qué manera clasifican estos la información.
- Alcanzable: Se deben comparar los algoritmos clasificadores J48 e ID3 observando los modelos o árboles que generan éstos para clasificar la información.
- Relevante: Es fundamental efectuar la comparación entre los dos algoritmos pues se debe seleccionar el que mejor eficacia tenga en cuanto a clasificación, de esta manera el sistema experto podrá tener una eficacia muy buena.

2.4.2. Requisitos funcionales no nominales

Requisitos para el funcionamiento del sistema en situaciones especiales o condiciones de error.

FF.1 Conexión a Weka.

Descripción: El aprendizaje de información del Sistema Experto solo se podrá llevar a cabo si se tiene conexión con la plataforma Weka.

Importancia: Esencial.

Validez:

- Medible: Al no existir conexión con la plataforma Weka el sistema no podrá tener un aprendizaje automático de la información.
- Alcanzable: Se debe validar que el Sistema cuente con la conexión apropiada a la plataforma Weka y de esta manera realice el aprendizaje de la información por medio del uso de algoritmos clasificadores.
- Relevante: Es importante la validación pues sin esta el Sistema no aprenderá y por ende no se podrá determinar a los posibles portadores de Chagas.

FF.2 Control de excepciones y error por operación.

Descripción: Los usuarios podrán consultar información desde la base de datos MySQL sólo si esta contiene registros almacenados.

Importancia: Esencial.

Validez:

- Medible: El sistema debe informar al usuario si no existen registros almacenados en la base de datos.
- Alcanzable: El sistema debe tener previamente la base de conocimientos migrada a su base de datos, para mostrar al usuario la información que el sistema debe analizar, de lo contrario debe mostrar al usuario que no existen registros almacenados en la base de datos.
- Relevante: Es importante pues si no se tiene previamente la base de conocimientos migrada a la base de datos el usuario no podrá conocer los datos que el sistema analizará para generar los respectivos reportes y si no existen estos registros el usuario debe conocer que no existen registros en la base de datos.

2.5. Requisitos de interfaz

Conjunto de requisitos que definen las necesidades de la interacción del sistema con otros sistemas y usuarios.

IN.1 Botones.

Descripción: Por pantalla debe existir un número determinado de botones que permita al usuario tener la certeza de que la acción que va a realizar es la correcta.

Importancia: Esencial.

Validez:

- Medible: El Sistema deberá proporcionar el nombre a los botones de acuerdo a las acciones que se puedan realizar con estos.
- Alcanzable: Los botones deben ser intuitivos y específicos según los estándares de la W3C, para que los usuarios puedan dar un correcto manejo al sistema.
- Relevante: Es importante porque el usuario tenga la certeza de que la acción que va a realizar es la correcta.

IN.2 Menús intuitivos.

Descripción: El sistema debe contar con menús intuitivos según los estándares de la W3C,que permitan al usuario acceder a los diferentes módulos y conocer donde se encuentra situado en el sistema.

Importancia: Esencial.

Validez:

- Medible: El Sistema deberá proporcionar al usuario la facilidad de moverse en este y tener conocimiento de donde se encuentra ubicado en este.
- Alcanzable: El sistema debe contar con un menú horizontal y vertical, como también un breadcrumb, los cuales permitan al usuario moverse fácilmente por los módulos de la aplicación y conocer a su vez donde se encuentra situado.
- Relevante: Es importante porque el usuario debe moverse fácilmente por los módulos de la aplicación y conocer a su vez donde se encuentra situado.

2.6. Requisitos de calidad

Exigencias en la calidad que se piden explícitamente para el producto. En esta categoría se engloban los requisitos de rendimiento, accesibilidad, facilidad de uso, etc.

CA.1 Diseño responsivo

Descripción: Las interfaces del Sistema deben adaptarse visualmente en cualquier dispositivo electrónico que cuente con un navegador Web.

Importancia: Esencial.

Validez:

- Medible: El sistema podrá adaptarse visualmente a diferentes dispositivos electrónicos tales como, computadoras, celulares, tabletas, entre otras.
- Alcanzable: Cada una de las partes de las interfaces del sistema deben ser configuradas para que se adapten a diversos dispositivos electrónicos.
- Relevante: Es importante puesto que el usuario no se limitará a tener a su alcance un solo dispositivo para usar el Sistema, sino que por el contrario podrá hacer uso de cualquier dispositivo electrónico que cuente con un navegador Web.

2.7. Requisitos de evolución

Requisitos para el diseño del producto con el objetivo de facilitar la adaptación a exigencias o condiciones que puedan surgir en el futuro.

EV.1 Escalabilidad.

Descripción: Con el paso del tiempo se hace importante realizar modificaciones al sistema, por ello debe desarrollarse de tal forma que en determinado momento puedan realizar estas.

Importancia: Esencial.

Validez:

- Medible: Se deben definir correctamente las actividades del sistema para diseñar los módulos de esta.
- Alcanzable: Se debe desarrollar el sistema mediante un diseño modular.
- Relevante: Al realizar modificaciones con el paso del tiempo, se hace importante que al hacerlas en una parte del Sistema éste no se vea afectado en su totalidad, por ello se debe desarrollar de forma modular, de tal manera que al modificar algo no se vea afectado todo el funcionamiento del sistema.

EV.2 Implementación.

Descripción: Se hace importante analizar la arquitectura del sistema pues este hace uso de otra plataforma.

Importancia: Esencial.

Validez:

- Medible: Implementar un tipo de arquitectura que permita la comunicación con la plataforma Weka.
- Alcanzable: Definir en el diseño del sistema cual será la arquitectura a implementar y como esta permitirá la conexión entre el sistema y Weka.
- Relevante: Es importante definir en el diseño del sistema de qué manera se comunicará este con la plataforma Weka y así no influya esto más adelante en otros procesos del desarrollo.

2.8. Requisitos de proyecto

Requisitos que afectan y condicionan el proceso de desarrollo del proyecto.

PR.1 Periodo de desarrollo.

Descripción: Para llevar a cabo el Sistema Experto se cuenta con tres meses, esto debido al proceso de intercambio que se realiza entre la Universidad UPEMOR y la Universidad Santo Tomás-Seccional Tunja.

Importancia: Esencial.

Validez:

- Medible: Se deben especificar las actividades y generar tiempos de trabajo para estas, cumpliendo así metas que permitan el desarrollo del Sistema Experto en el tiempo determinado.
- Alcanzable: Debe crearse una bitácora y cronograma donde se especifique que actividades se deben realizar, las fechas de entrega de estas y los temas abordados durante el trabajo diario que se realice.
- Relevante: Al ser un periodo de tiempo tan corto para el desarrollo del proyecto es fundamental organizar muy bien el tiempo pues esto permitirá cumplir metas a corto y largo plazo, de tal manera que se finalice el proyecto en el tiempo estimado.

2.9. Requisitos de soporte

Requisitos que deben ser cumplidos por el cliente (a diferencia de los anteriores).

OS.1 El usuario debe tener un navegador instalado.

Descripción: Para que el usuario pueda hacer uso del Sistema Experto debe hacer uso de un navegador Web.

Importancia: Esencial.

Validez:

- Medible: Para acceder al Sistema Experto el usuario debe de hacer uso de un navegador Web, por ello debe tener instalado éste en su dispositivo electrónico.
- Alcanzable: En caso de que el dispositivo electrónico con el cual se quiera acceder al sistema no cuente con un navegador Web, debe de instalarse uno en dicho dispositivo.
- Relevante: Al contar con un navegador Web el usuario puede hacer uso del sistema de lo contrario no lo puede hacer.

OS.2 Hacer uso de las herramientas tecnológicas.

Descripción: El usuario que desee hacer uso del Sistema Experto debe poseer los conocimientos tecnológicos básicos para poder gestionar éste, pues debe hacer uso de un navegador Web y por ende de un dispositivo electrónico.

Importancia: Esencial.

Validez:

- Medible: Para hacer uso del sistema el usuario debe tener conocimientos básicos sobre el dispositivo electrónico del que hará uso para acceder al sistema.
- Alcanzable: El sistema debe de proporcionar un módulo de ayuda al usuario, el cual contenga una guía de su correcto manejo y una descripción de las funcionalidades de éste.
- Relevante: Es importante que el usuario tenga conocimientos básicos de la herramienta tecnológica con la que accederá al sistema, pero de igual manera es fundamental que el sistema contenga una guía de ayuda al usuario donde describa el correcto manejo de este y se proporcione una descripción de sus funcionalidades.

OS.3 Tener conocimientos de estadística.

Descripción: El usuario que desee hacer uso del Sistema Experto debe poseer conocimientos básicos de estadística por medio de los cuales podrá entender los reportes del sistema.

Importancia: Esencial.
Validez:

- Medible: Para hacer uso del sistema el usuario debe tener conocimientos básicos sobre estadística pues los reportes contendrán gráficas que expliquen los datos de los posibles portadores de Chagas.
- Alcanzable: El sistema debe proporcionar una ayuda visual al usuario que contenga descripciones que expliquen los lineamientos de las gráficas presentadas.
- Relevante: Es importante que el usuario tenga conocimientos básicos de estadística pues los reportes contendrán gráficas que expliquen los datos de los posibles portadores de Chagas, también el sistema debe proporcionar las respectivas descripciones que expliquen las gráficas generadas.

Capítulo 3. DISEÑO DEL SISTEMA

El presente capítulo muestra el diseño que se implementó para llevar a cabo el Sistema Experto presentando la arquitectura, el diseño de los requisitos, interfaces y bases de datos.

3.1. Arquitectura del sistema

En esta sección, se explica qué arquitectura lógica utiliza el Sistema Experto para posibles portadores de la enfermedad de Chagas.

En la Figura 2, se aprecia un ejemplo gráfico de lo que es la arquitectura mediante la cual se lleva a cabo el desarrollo del sistema, donde se separa la lógica de negocio de la interfaz de usuario, facilitando así la evolución por separado del sistema e incrementando la flexibilidad y la reutilización en el código. Dicha arquitectura es conocida como Modelo Vista Controlador (MVC) en la cual:

- Modelo: Es la capa que contiene toda la información sobre los datos, cómo se debe acceder a estos, cual es el comportamiento de estos y la relación entre estos.
- Vista: Mediante esta capa se presenta al usuario las diferentes interfaces de las que puede hacer uso para gestionar el sistema.
- Controlador: Esta capa funciona como un puente entre las vistas y el modelo, así el sistema puede gestionar de manera efectiva las acciones que el usuario quiere llevar a cabo mediante la interfaz.

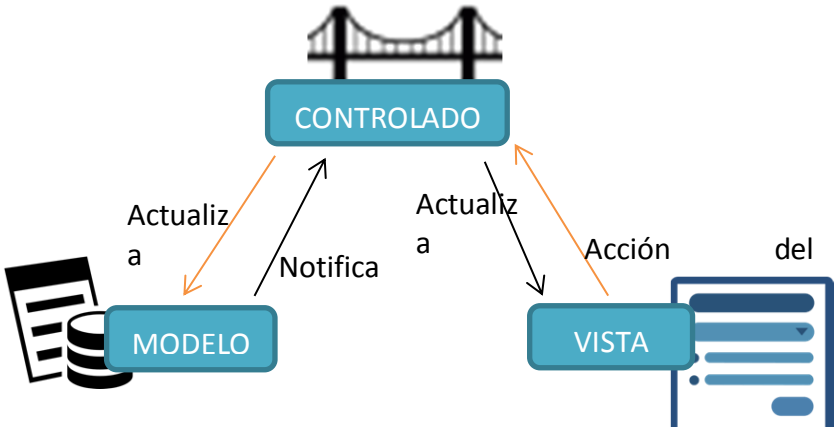

Figura 2. Representación gráfica del modelo vista controlador
Fuente: John Leandro Mejía Castro.

3.2. Diseño de requisitos funcionales

- *Figura 3*, se puede observar mediante un diagrama de secuencia el requisito funcional número uno el cual es importante porque es mediante este que se genera la base de conocimientos adecuada para poder realizar el análisis, aprendizaje y posterior consulta de la información.

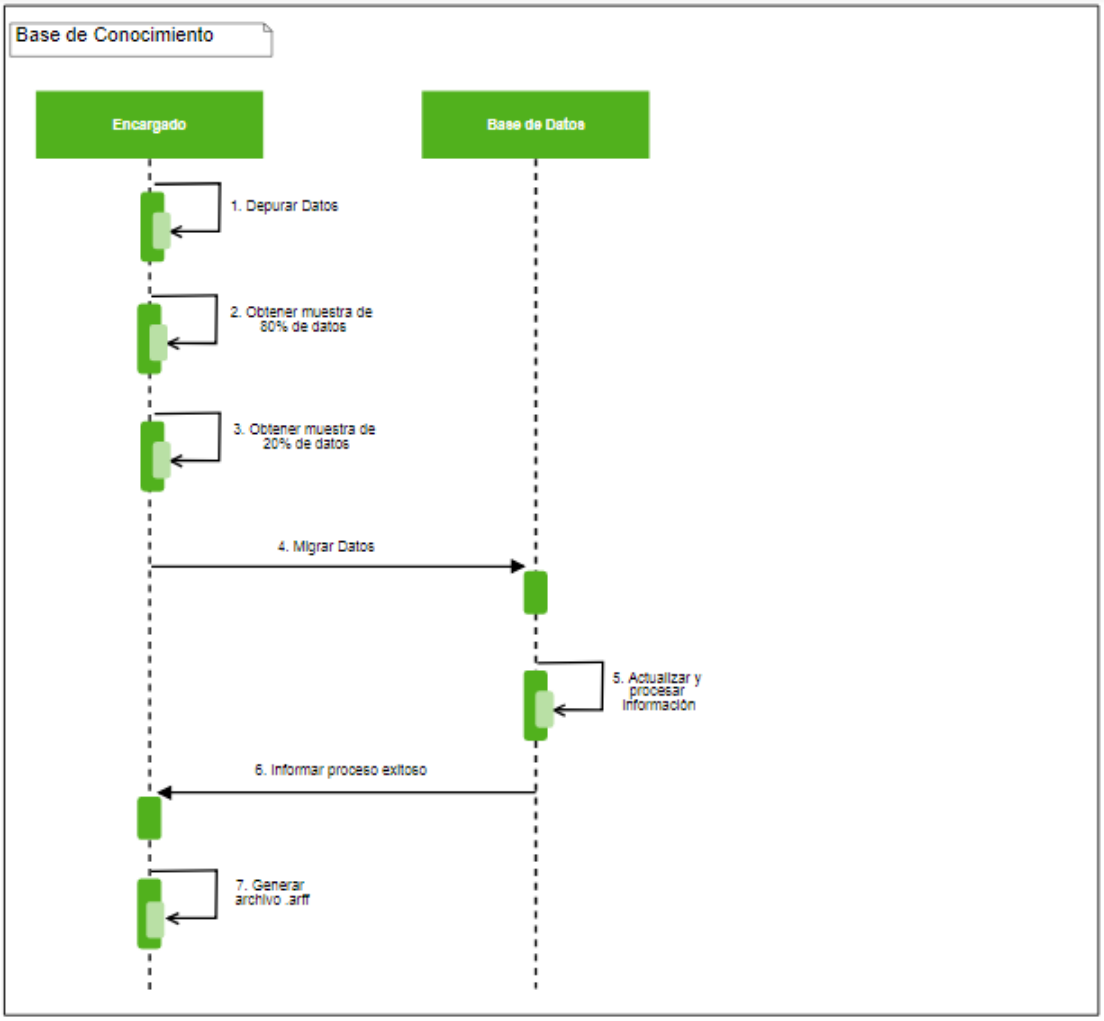

Figura 3. Diagrama de secuencia de base de conocimiento (FN1)
Fuente: John Leandro Mejía Castro.

- *Figura 4*, se presenta el diagrama de secuencia del requisito funcional nominal dos, mediante este requisito se pretende realizar la consulta a la base de datos y presentar de forma ordenada al usuario todos los datos que podrían llegar a ser analizados por el sistema para determinar los posibles portadores de Chagas.

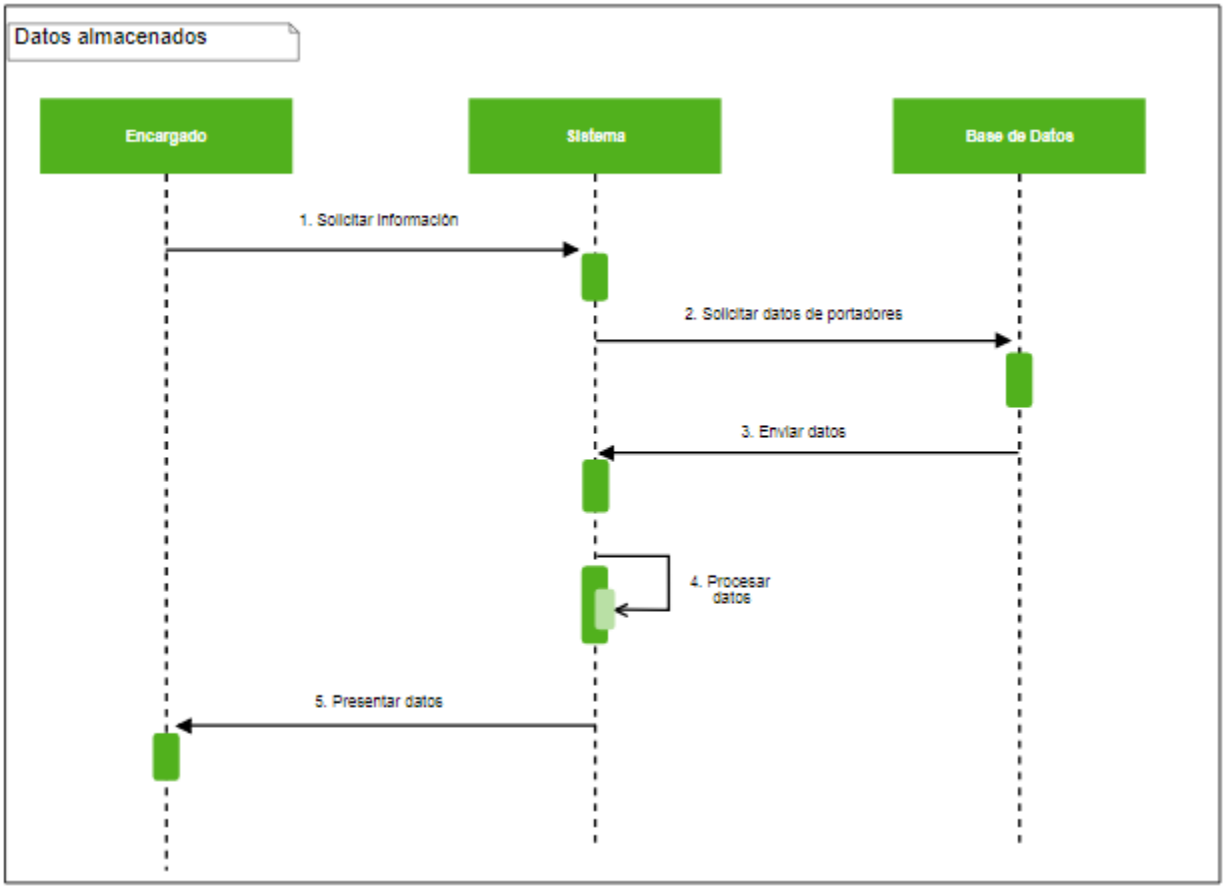

Figura 4. Diagrama de secuencia de presentación de datos almacenados (FN2)
Fuente: John Leandro Mejía Castro.

- *Figura 5*, se visualiza el diagrama de secuencia del requisito funcional nominal tres, lo que pretende este requisito es generar un reporte por municipio seleccionado por el usuario, con el respectivo análisis de los datos de posibles portadores de Chagas y así mediante el uso de estadística poder representar el porcentaje de posibles portadores que puedan tener esta enfermedad.

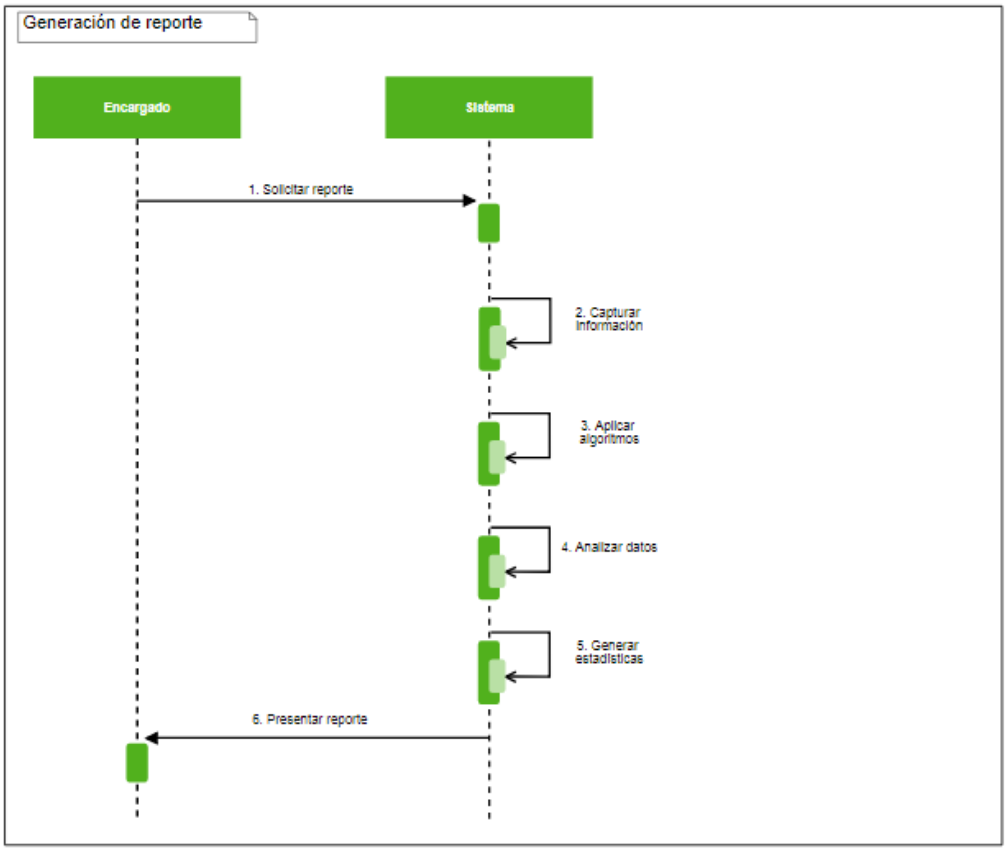

Figura 5. Diagrama de secuencia de generación de reporte (FN3)
Fuente: John Leandro Mejía Castro.

3.3. Diseño de la interfaz de usuario

En esta sección, se realiza una representación gráfica de lo que será el Sistema Experto, a continuación se ilustra:

Figura 6, se muestra la interfaz de Inicio del Sistema Experto, esta es la primera vista que tendrá el usuario del sistema, contiene un menú vertical fijo del cual podrá hacer uso para ir a los diferentes módulos del sistema, también contiene un breadcrumb que le permitirá al usuario saber en qué parte del sistema se encuentra ubicado facilitando así la navegación en este y un botón de información en el menú horizontal que le permitirá a este conocer la versión y los autores de la aplicación.

Figura 6. Interfaz de Inicio
Fuente: John Leandro Mejía Castro.

Figura 7, se observa la interfaz de Reporte del Sistema Experto, en ella el usuario podrá hacer uso de un combo que le permita seleccionar un municipio que se encuentre almacenado en la base de datos, también podrá hacer uso del botón ver reporte el cual le permitirá de acuerdo al municipio seleccionado obtener la información sobre los posibles portadores del Chagas en dicha zona.

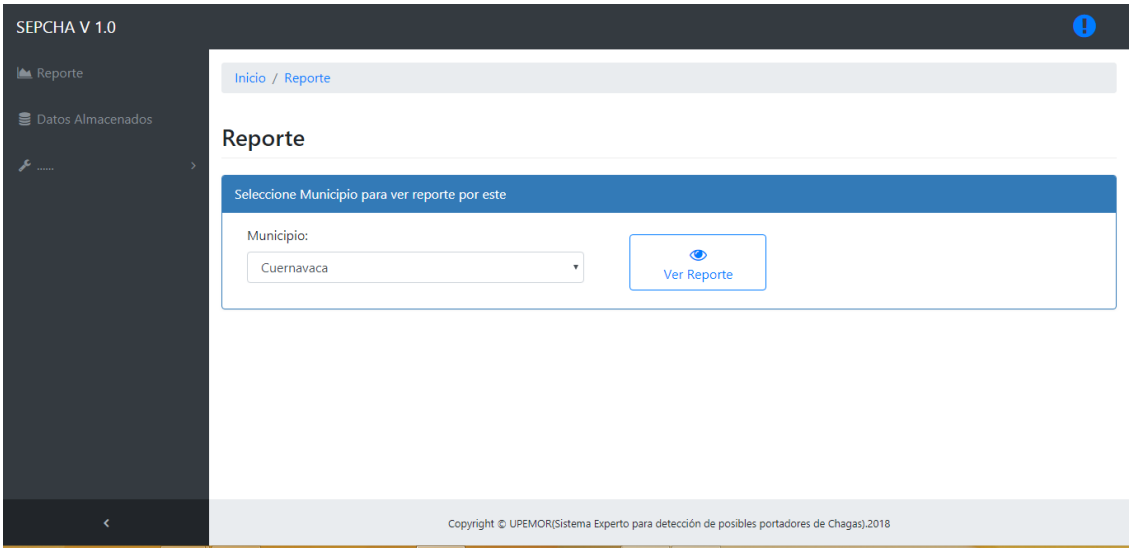

Figura 7. Interfaz reporte
Fuente: John Leandro Mejía Castro.

Figura 8, esta interfaz permite al usuario visualizar los registros que contiene la base de datos, también puede hacer uso de una caja de texto donde puede escribir la palabra a buscar en los registros la cual se actualizará en tiempo real.

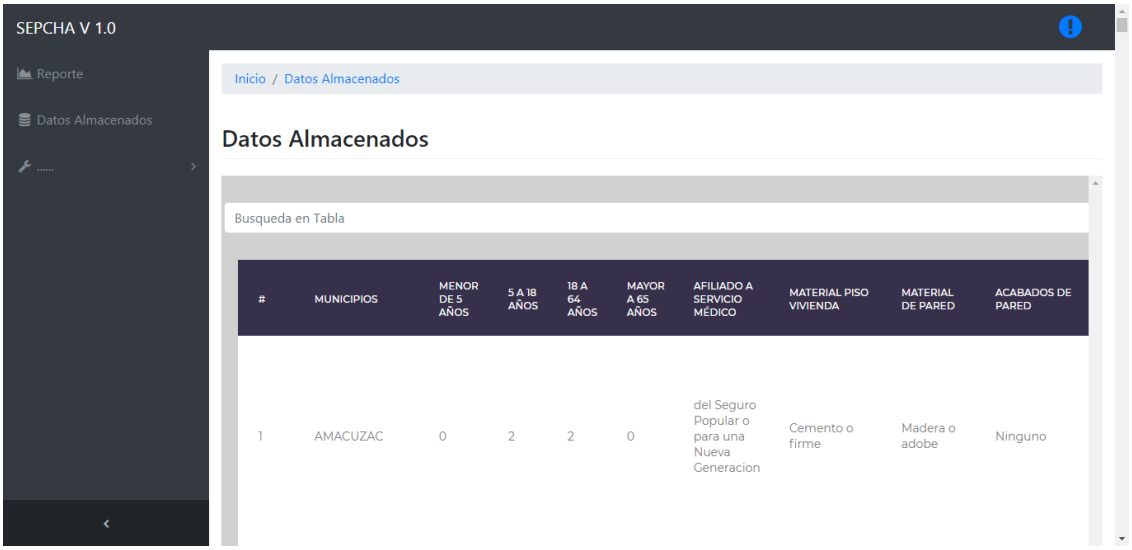

Figura 8. Interfaz datos almacenados
Fuente: John Leandro Mejía Castro.

3.4. Diseño de base de datos

A continuación se presentan los diagramas entidad relación y el diagrama físico efectuados en PowerDesigner los cuales se encuentran normalizados.

3.4.1. Diagrama Entidad-Relación

Figura 9, se muestra el diseño lógico de la base de datos que lleva por nombre "sistema_experto", esta contiene once tablas (municipio, vector, registro_vector, material_techo, material_piso_vivienda, servicio_agua_publica, material_acabado_pared, drenaje, material_pared, registro) las cuales contienen su respectivo grupo de atributos donde se almacena cada dato.

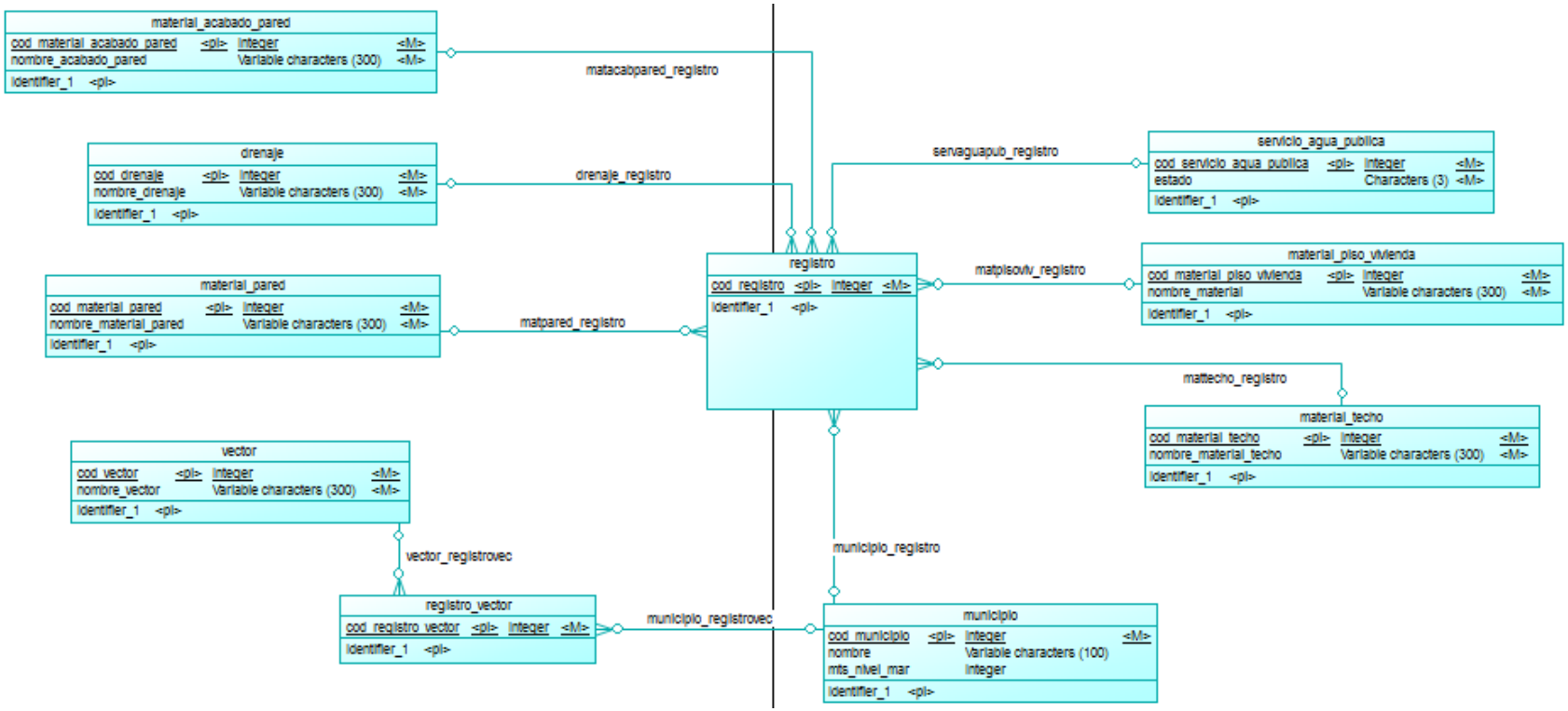

Figura 9. Diagrama lógico
Fuente: John Leandro Mejía Castro.

Figura 10, se presenta el diagrama físico de la base de datos "sistema_experto" en el cual se pueden observar las 11 tablas (municipio, vector, registro_vector, material_techo, material_piso_vivienda, servicio_agua_publica, material_acabado_pared, drenaje, material_pared, registro) sus atributos propios y las llaves foráneas que adquieren debido a sus relaciones.

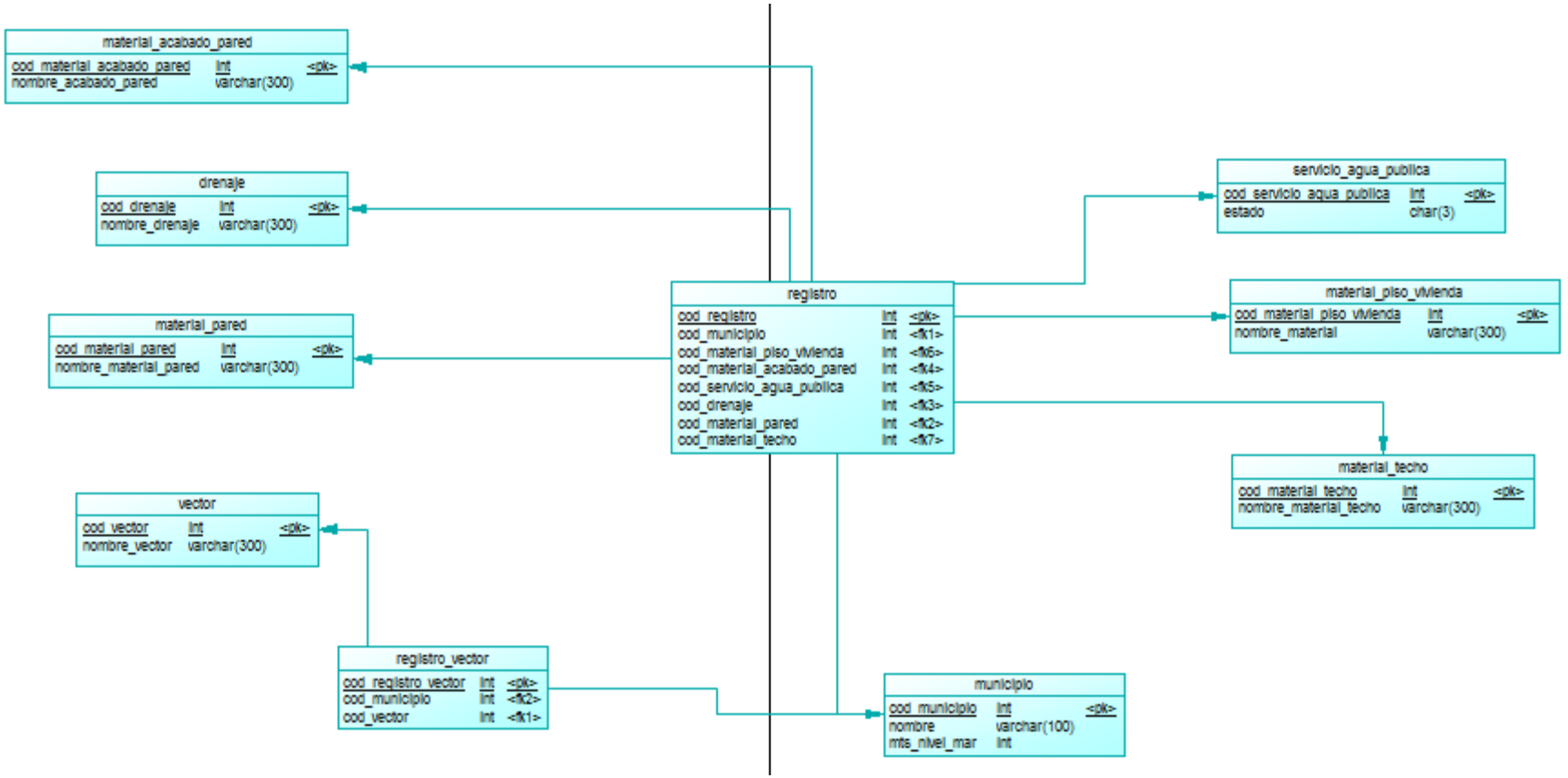

Figura 10. Diagrama físico
Fuente: John Leandro Mejía Castro.

3.4.2. Diccionario de datos

En esta sección, se mostrará la descripción de cada una de las tablas que contiene la base de datos con sus respectivos atributos y una descripción de estos.

Tabla 3, se muestra la tabla que lleva por nombre "vector", en la cual se pueden almacenar los nombres de los diversos vectores de los que se tiene conocimiento.

Tabla 3. Diccionario de datos, tabla "vector"

Dato	Tipo	Longitud	Descripción
cod_vector	Entero	11	Identificador único auto- incrementable que se crea para identificar un nombre de un vector.
nombre_vector	Cadena	300	Nombre con el que se identifica el vector.

Fuente: John Leandro Mejía Castro.

Tabla 4, se presenta la tabla "municipio", en la cual se pueden almacenar los nombres de los municipios que podrán ser analizados por el sistema.

Tabla 4. Diccionario de datos, tabla "municipio"

Dato	Tipo	Longitud	Descripción
cod_municipio	Entero	11	Identificador único auto-incrementable que se crea para identificar un nombre de un municipio.
nombre	Cadena	300	Nombre con el que se identifica el municipio.
mts_nivel_mar	Entero	11	Número que representa los metros sobre el nivel del mar a los que se encuentra el municipio.

Fuente: John Leandro Mejía Castro.

Tabla 5, se presenta la tabla "registro_vector", en ésta se pueden almacenar registros sobre los vectores que están presentes en los municipios.

Tabla 5. Diccionario de datos, tabla "registro_vector"

Dato	Tipo	Longitud	Descripción
cod_registro_vector	Entero	11	Identificador único auto-incrementable que se crea para identificar un registro de un vector que está presente en un municipio.
cod_municipio	Entero	11	Identificador foráneo de la tabla municipio con el que se identifica a un municipio.
cod_vector	Entero	11	Identificador foráneo de la tabla vector con el que se identifica a un vector.

Fuente: John Leandro Mejía Castro.

Tabla 6, se presenta la tabla "material_techo", en ésta se pueden almacenar un tipo de material de techo que puede tener una vivienda.

Tabla 6. Diccionario de datos, tabla "material_techo"

Dato	Tipo	Longitud	Descripción
cod_material_techo	Entero	11	Identificador único auto-incrementable que se crea para identificar un nombre de un material techo.
nombre_material_techo	Cadena	300	Nombre con el que se identifica un material de techo.

Fuente: John Leandro Mejía Castro.

Tabla 7, se observa la tabla "material_piso_vivienda", en ésta se puede almacenar un tipo de material del piso de la vivienda, con el que puede contar esta.

Tabla 7. Diccionario de datos, tabla "material_piso_vivienda"

Dato	Tipo	Longitud	Descripción
cod_material_piso_vivienda	Entero	11	Identificador único auto-incrementable que se crea para identificar un nombre de un material del piso de la vivienda.
nombre_material	Cadena	300	Nombre con el que se identifica un material de piso de vivienda.

Fuente: John Leandro Mejía Castro.

Tabla 8, se muestra la tabla "servicio_agua_publica", aquí se almacenan las opciones con las que podría contar una persona para hacer uso del servicio público.

Tabla 8. Diccionario de datos, tabla "servicio_agua_publica"

Dato	Tipo	Longitud	Descripción
cod_servicio_agua_publica	Entero	11	Identificador único auto-incrementable que se crea para identificar un nombre de un material del piso de la vivienda.
estado	Caracter	3	Estado u opción con la que cuenta la persona para hacer uso del agua como servicio público.

Fuente: John Leandro Mejía Castro.

Tabla 9, se muestra la tabla "material_acabado_pared", en ésta se puede almacenar un tipo de material del acabado de la pared de la vivienda.

Tabla 9. Diccionario de datos, tabla "material_acabado_pared"

Dato	Tipo	Longitud	Descripción
cod_material_acabado_pared	Entero	11	Identificador único auto-incrementable que se crea para identificar un nombre de un material del acabado de pared.
nombre_acabado_pared	Cadena	300	Nombre con el que se identifica un material del acabado de la pared.

Fuente: John Leandro Mejía Castro.

Tabla 10, se observa la tabla “drenaje”, aquí se almacenan los distintos tipos de drenaje con los que puede contar la vivienda.

Tabla 10. Diccionario de datos, tabla "drenaje"

Dato	Tipo	Longitud	Descripción
cod_drenaje	Entero	11	Identificador único auto-incrementable que se crea para identificar un nombre de un material del acabado de pared.
nombre_drenaje	Cadena	300	Nombre con el que se identifica un material del acabado de la pared.

Fuente: John Leandro Mejía Castro.

Tabla 11, se observa la tabla “material_pared”, en ella se almacenan los distintos tipos de material para las paredes con los que puede contar la vivienda.

Tabla 11. Diccionario de datos, tabla "material_pared"

Dato	Tipo	Longitud	Descripción
cod_material_pared	Entero	11	Identificador único auto-incrementable que se crea para identificar un tipo de material de la pared
nombre_material_pared	Cadena	300	Nombre con el que se identifica un tipo de material de la pared.

Fuente: John Leandro Mejía Castro.

Tabla 12, se observa la tabla “registro”, aquí se almacenan la información completa de una vivienda, conforme a los tipos de materiales, ubicación o servicios con que cuente ésta.

Tabla 12. Diccionario de datos, tabla "registro"

Dato	Tipo	Longitud	Descripción
cod_registro	Entero	11	Identificador único auto-incrementable que se crea para identificar un tipo de material de la pared
cod_municipio	Entero	11	Identificador foráneo de la tabla municipio con el que se identifica a un municipio.
cod_material_piso_vivienda	Entero	11	Identificador foráneo de la tabla material_piso_vivienda con el que se identifica a un tipo de material piso vivienda.
cod_material_acabado_pared	Entero	11	Identificador foráneo de la tabla material_acabado_pared con el que se identifica a un tipo de material de acabado de pared.
cod_servicio_agua_publica	Entero	11	Identificador foráneo de la tabla servicio_agua_publica con el que se identifica un

Dato	Tipo	Longitud	Descripción
			estado “si” o “no” de este servicio.
cod_drenaje	Entero	11	Identificador foráneo de la tabla drenaje con el que se identifica un tipo de drenaje.
cod_material_pared	Entero	11	Identificador foráneo de la tabla material_pared con el que se identifica a un tipo de material de pared.
cod_material_techo	Entero	11	Identificador foráneo de la tabla material_techo con el que se identifica a un tipo de material de techo.

Fuente: John Leandro Mejía Castro.

Capítulo 4. IMPLEMENTACIÓN

Este capítulo presenta la arquitectura física implementada en el Sistema Experto también las tecnologías implementadas y la documentación del código.

4.1. Arquitectura física

En esta sección, se muestra cómo está compuesta la arquitectura física del Sistema experto para posibles portadores de la enfermedad de Chagas.

En la Figura 11, se muestra el diagrama de componentes para el Sistema experto para posibles portadores de la enfermedad de Chagas el cual se compone de un servidor y el cliente. El cliente en este caso, es el que podrá acceder a la página del sistema y el servidor fungirá como almacenador de la información.

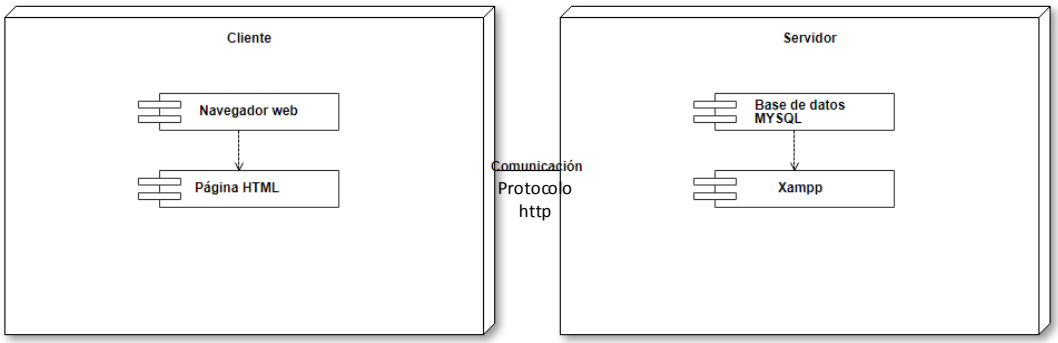

Figura 11. Arquitectura física Cliente-Servidor
Fuente: John Leandro Mejía Castro.

4.2. Tecnologías empleadas

En esta sección, se describirán cada una de las tecnologías utilizadas con sus características correspondientes para el desarrollo del sistema.

Para el desarrollo del sistema:

- HTML 5
  Descripción: Lenguaje de hiper texto utilizado para diseñar y estructurar páginas web. (Rodríguez, 2006)

Versión: 5
Justificación: Es fundamental hacer uso de este lenguaje pues se debe contar con una estructura para el sitio web experto.

- Java EE
  Descripción: Java Enterprise Edition plataforma de programación para desarrollar y ejecutar software de aplicaciones en el lenguaje Java, Java EE cuenta con varias especificaciones y tecnologías de servicio web. (IBM, 2011)
  Versión: 7.1
  Justificación: Es importante hacer uso de esta plataforma para desarrollar el sistema experto pues permite la conexión con la Api de Weka y generar los reportes web.

- Bootstrap
  Descripción: Es un framework que tiene como objetivo facilitar el diseño web, Permite crear páginas web de manera sencilla y con un diseño adaptable. (Maria, 2016)
  Versión: 4.0.0
  Justificación: El uso de un framework para el diseño web facilita el desarrollo del sistema experto pues evita tener que incurrir en tiempo y diseños que este proporciona con facilidad.

- XAMPP
  Descripción: Es un servidor multiplataforma de software libre el cual permite visualizar páginas web o algún otro desarrollo desde el ordenador sin necesidad de acceder a internet. (Jorge, 2017)
  Versión: 3.2.2
  Justificación: Gracias a este gestor de servicios se puede implementar el sistema web experto localmente evitando costos adicionales.

- JavaScript
  Descripción: Lenguaje de programación que permite realizar actividades complejas para el desarrollo de páginas web, creando contenido nuevo y dinámico. (bosspetta, 2018)
  Versión: ECEMAScript 6
  Justificación: El sistema necesita contenido dinámico por ende es muy importante que haga uso de este lenguaje.

- API Weka
  Descripción: Biblioteca que proporciona una gran colección de algoritmos de aprendizaje, implementados en Java. (Waikato, 2018)
  Versión: 3.8.2
  Justificación: La API de Weka permite implementar los algoritmos con los que el sistema experto clasificará los diversos registros.

- NetBeans
  Descripción: Es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. (NETBEANS, 2018)
  Versión: 3.8.1
  Justificación: Este entorno de desarrollo permite hacer uso de varios lenguajes y estructurar de forma fácil e intuitiva el proyecto del sistema experto por ello se hace fundamental su uso.

Para el desarrollo de elementos para el diseño:

- Axure RP
  Descripción: Herramienta de software de prototipado rápido, documentación y especificación destinada a aplicaciones web y de escritorio. (marketingscientec, 2016)
  Versión: 8.0.0.3293
  Justificación: Gracias a esta herramienta se elaboran los prototipos realistas de interfaces con los que va a contar el sistema experto.

- Cacoo
  Descripción: Plataforma web para diseñar y modelar diagramas de software. (CACOO, 2018)
  Versión: 2.8.0
  Justificación: Se deben modelar diversos diagramas de software, por ello esta plataforma es fundamental.

- Powerdesigner
  Descripción: Es una herramienta de modelado de colaboración empresarial usada para generar los modelos y el script de la base de datos. (POWERDESIGNER, 2015)
  Versión: 16.5
  Justificación: Para realizar los diagramas pertinentes de la base de datos es fundamental hacer uso de esta plataforma pues aparte de generar los diagramas genera los scripts para crear la base de datos.

4.3. Proceso de implementación del sistema experto

Para el desarrollo del sistema experto se lleva a cabo un proceso de extracción de conocimiento mediante el cual se pretende que el sistema con base a ciertas reglas esté en capacidad de diagnosticar las zonas con posibles incidencias de la enfermedad de Chagas, por ello este proceso se hace sumamente importante en el desarrollo del

proyecto, en la **Figura 12** se describe como se llevó a cabo éste y se explica cómo se implementan en el proyecto cada una de sus partes.

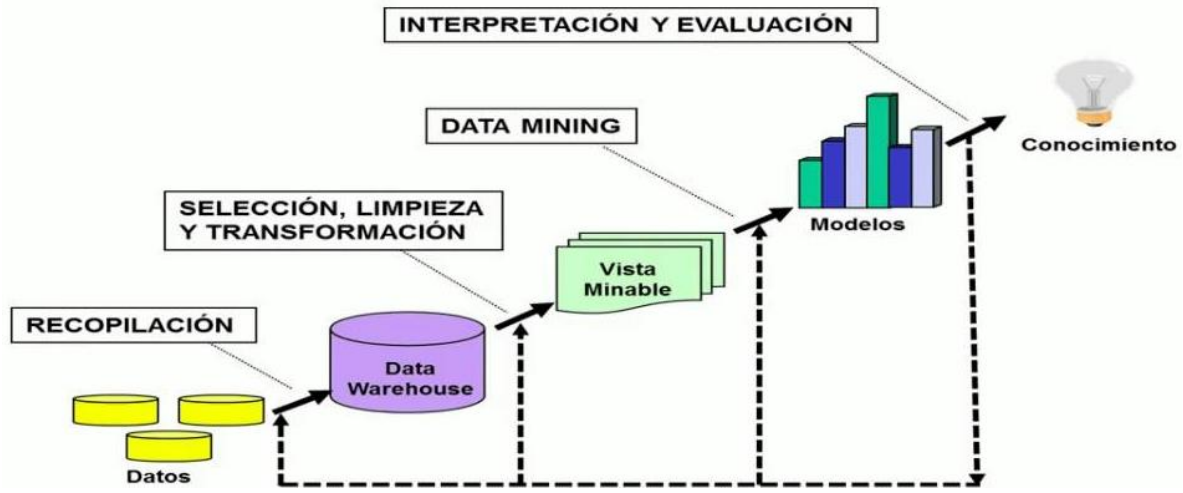

Figura 12. Proceso de extracción de conocimiento
Fuente: Juan Lara, 2014

4.3.1 Proceso de extracción de conocimientos – Datos, Recopilación

Este es el primer paso del proceso de extracción de conocimientos en el cual se logra obtener una base de datos por parte del censo del 19 de septiembre del año 2017, dicha base de datos contiene 22.810 registros con 57 entidades o factores a evaluar, de estos factores debe proceder a seleccionarse los adecuados para llevar a cabo el proceso de extracción de conocimientos, para ello se necesita tener un conocimiento experto. Gracias a un simposio Latinoamericano de Cardiopatía Chagásica llevado a cabo en la ciudad de Cuernavaca, Morelos, se logra generar un conocimiento experto mediante el cual se procede a hacer una recopilación de datos pertinentes e importantes que puedan ser analizados por el sistema, formando así un almacén de datos (Data Warehouse).

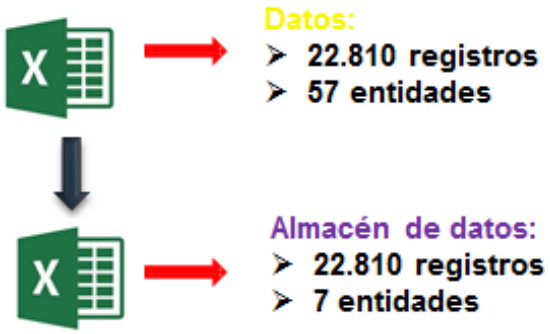

Figura 13. Proceso de extracción de conocimientos (datos, almacén de datos)
Fuente: John Leandro Mejía Castro

Como se puede observar en la Figura 13 la etapa de recopilación de los datos es muy importante puesto que se definen los factores que tendrán incidencia en el análisis del sistema experto. Primero se obtiene por parte de las instituciones un archivo .XLS el cual contiene 57 entidades o factores que deben pasar por un proceso de limpieza o recopilación que al llevarse a cabo genera un archivo .XLS con siete factores que tienen incidencia en el análisis, se determinan estos mediante los conocimientos de la persona experta en la enfermedad de Chagas, estos son:

- Municipio
- Material del piso de la vivienda
- Material de la pared de la vivienda
- Material del techo de la vivienda
- Material del acabado de pared
- Servicio de agua potable de la vivienda
- Servicio de drenaje de la vivienda

Cada uno de los 22.810 registros de los siete factores o entidades deben ser limpiados, para ello hay que eliminar las tildes de éstos y acotar las palabras, tal como se puede observar en la Figura 14 y Figura 15:

	municipiio	material_piso_vivienda	otro_material_pared	otro_acabado_pared	otro_material_techo	servicio_agua_public	drenaje
2	AMACUZAC	Cemento o firme	Madera o adobe	Ninguno	Carton, hule, tela, llantas, lamina de carton, carrizo, bambu, palm	No	Hoyo negro, pozo ciego o letri
3	AMACUZAC	Cemento o firme	Panel de concreto o tabla ro	Recubrimiento con pintura o tapiz	Tabique, ladrillo, tabicon, block, concreto monolitico, piedra o c	No	Excusado o sanitario con conex
4	AMACUZAC	Cemento o firme	Tabique, ladrillo, tabicon, blo	Cemento	Lamina metalica, fibra de vidrio, plastico, mica o asbesto	No	Excusado o sanitario sin conex
5	AMACUZAC	Cemento o firme	Madera o adobe	Ninguno	Lamina metalica, fibra de vidrio, plastico, mica o asbesto	No	Excusado o sanitario sin conex
6	AMACUZAC	Cemento o firme	Tabique, ladrillo, tabicon, blo	Recubrimiento con pintura o tapiz	Tabique, ladrillo, tabicon, block, concreto monolitico, piedra o c	No	Excusado o sanitario con conex
7	AMACUZAC	Cemento o firme	Tabique, ladrillo, tabicon, blo	Ninguno	Lamina metalica, fibra de vidrio, plastico, mica o asbesto	No	Hoyo negro, pozo ciego o letri
8	AMACUZAC	Cemento o firme	Madera o adobe	Cemento	Lamina metalica, fibra de vidrio, plastico, mica o asbesto	No	Excusado o sanitario con conex
9	AMACUZAC	Cemento o firme	Tabique, ladrillo, tabicon, blo	Recubrimiento (pasta, yeso) sin pintu	Lamina metalica, fibra de vidrio, plastico, mica o asbesto	Si	Excusado o sanitario sin conex

Figura 14. Registros a ser limpiados
Fuente: John Leandro Mejía Castro

	municipio	mat_piso_vivi	mat_pared	mat_techo	mat_acab_pared	serv_agua	drenaje
2	AMACUZAC	cemento-firme	madera-adobe	carton-hule	Ninguno	No	letrina
3	AMACUZAC	cemento-firme	panel-concreto	tabique-ladrillo	rec-pin-tapiz	No	sanita-con-conex
4	AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Cemento	No	sanita-sin-conex
5	AMACUZAC	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex
6	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	rec-pin-tapiz	No	sanita-con-conex
7	AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Ninguno	No	letrina

Figura 15. Registros limpiados
Fuente: John Leandro Mejía Castro

4.3.2 Proceso de extracción de conocimientos – Data Warehouse, limpieza, transformación y vista minable

Después de llevar a cabo la recopilación de datos se obtiene un archivo .XLS el cual contiene el almacén de datos (Data Warehouse) con cada uno de los 22.810 registros acotados y sin tildes. Posteriormente esos datos deben seguirse tratando, es por ello que se debe llevar a cabo una fase de limpieza y transformación de éstos para lo cual se aplican las siguientes técnicas:

- Eliminación de ruido
- Valores erróneos

Para llevar a cabo la técnica de eliminación de ruido se aplica la media, es decir del factor que se está observando se obtiene el promedio de cuál es el registro más usado y se procede a implementar, puesto que en diversos registros hay datos faltantes o erróneos. También hay datos erróneos que no concuerdan con otros factores o entidades y por tal motivo deben ser eliminados. Al limpiar y tratar los 22.810 registros se procede a obtener un archivo .XLS con un total de 16.440 registros eliminando así 6.370 registros, de esta manera se genera lo que en el proceso de extracción de conocimiento se conoce como vista minable.

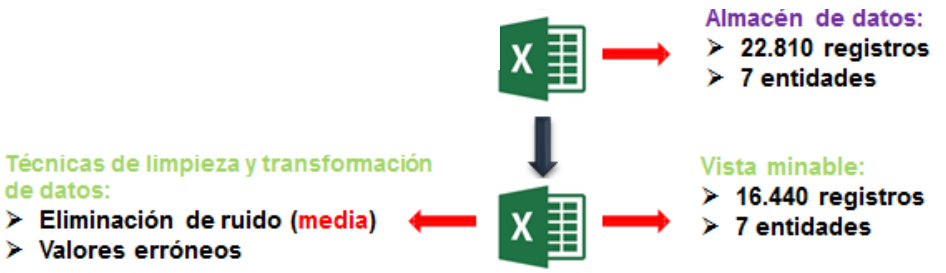

Figura 16. Proceso de extracción de conocimientos (almacén de datos, vista minable)
Fuente: John Leandro Mejía Castro

Al generar la vista minable se hace importante hacer una selección de datos para poder aplicar correctamente la minería de datos, pues los algoritmos a comparar y posterior mente a seleccionarse uno de éstos deben ser entrenados y probados, por ésto se procede a seleccionar el 80% de los datos para entrenamiento y el 20% de éstos para pruebas.

Figura 17. Proceso de extracción de conocimientos (selección de registros)
Fuente: John Leandro Mejía Castro

La selección descrita anteriormente procede a hacerse por muestra de población, es decir de cada uno de los 33 municipios del estado de Morelos que se encuentran en esa vista minable deben seleccionarse aleatoriamente un 80% para entrenamiento y un 20% para pruebas, de esta manera se obtiene un total de 13.152 registros (80%) para entrenamiento y 3.288 registros (20%) para pruebas. La selección implementada por muestra de población se describe en la tabla a continuación (tabla 13 "*Selección por muestra de población*").

Tabla 13. Selección por muestra de población

Municipio	Número de registros por municipio	Selección de registros 80%	Selección de registros 20%
Amacuzac	31	25	6
Atlatlahucan	75	60	15
Axochapan	350	280	70
Ayala	260	208	52
Coatlan del rio	151	121	30
Cuautla	311	249	62
Cuernavaca	671	537	134
Emilianozapata	357	286	71
Huitzilac	316	253	63
Jantetelco	1394	1115	279
Jiutepec	810	648	162
Jojutla	1588	1270	318
Jonaca tepec	176	141	35
Mazatepec	283	226	57
Miacatlan	304	243	61
Ocultuco	461	369	92
Puente de Ixtla	1136	909	227
Temixco	219	175	44
Temoac	97	78	19
Tepalcingo	1819	1455	364
Tepoztlan	368	294	74
Tetecala	214	171	43
Tetela del Volcan	475	380	95
Tlalneplanta	265	212	53
Tlaltizapan	611	489	122
Tlaltizapan de zapata	223	178	45
Tlalquiltenango	789	631	158
Tlayacapan	246	197	49
Totolapan	699	559	140
Xochitepec	17	14	3
Yautepec	471	377	94
Yecapixtla	261	209	52
Zacatepec	657	525	132

Municipio	Número de registros por municipio	Selección de registros 80%	Selección de registros 20%
Zacualpan de amilpas	335	268	67

Fuente: John Leandro Mejía Castro.

Al llevar a cabo la selección descrita anteriormente se generan dos archivos .XLS, uno donde se encuentran 13.152 registros (80%) para entrenamiento y otro archivo con 3.288 registros (20%) para pruebas.

El archivo .XLS con el 80% de registros y 7 entidades o factores importantes debe ser clasificado por una persona que tenga los conocimientos necesarios para determinar que registros de los 13.152 pueden ser por un lado positivos (P) y por otro lado negativos (N) a tener la enfermedad de Chagas, dicha clasificación se debe generar como otra entidad donde se defina la clasificación del registro si es positivo o negativo Figura 18.

municipio	mat_piso_vivi	mat_pared	mat_techo	mat_acab_pared	serv_agua	drenaje	clasificaci
AMACUZAC	cemento-firme	madera-adobe	carton-hule	Ninguno	No	letrina	P
AMACUZAC	cemento-firme	panel-concreto	tabique-ladrillo	rec-pin-tapiz	No	sanita-con-conex	N
AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Cemento	No	sanita-sin-conex	N
AMACUZAC	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	P
AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	rec-pin-tapiz	No	sanita-con-conex	N
AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Ninguno	No	letrina	N
AMACUZAC	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	No	sanita-con-conex	N
AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	rec-past-yes-snpin	Si	sanita-sin-conex	N

Figura 18. Clasificación experta a registros
Fuente: John Leandro Mejía Castro

Al llevar a cabo la anterior clasificación se tienen en cuenta diversos factores que generan el ambiente necesario o ideal para que el vector triatoma portador de la bacteria tripanosoma Cruzi pueda vivir allí. Las tablas presentadas a continuación presentan los diversos materiales o servicios que pueden influir en un buen hábitat para el vector triatoma y por ende la clasificación del registro como positiva, al ser posibles portadores de la enfermedad de Chagas.

Tabla 14. Materiales adecuados para un buen habitad del triatoma

Factor o entidad a analizar	Tipo de Material
Material del piso de la vivienda	**cemento-firme, madera-mosaico, tierra**
Material de la pared de la vivienda	**carton-hule, madera-adobe, otro.**
Material del techo de la vivienda	**Carton-hule, lámina metálica fibra de vidrio, madera-adobe, panel-concreto, tabique-ladrillo.**
Material acabado pared de vivienda	**Ninguno, recubrimiento pasta yeso sin**

Factor o entidad a analizar	Tipo de Material
	pintura, recubrimiento pintura tapiz.
Servicio agua potable de la vivienda	**Sí, no.**
Servicio drenaje de la vivienda	**Letrina, no tiene servicio sanitario, sanitario con conexión, sanitario sin conexión.**

Fuente: John Leandro Mejía Castro.

De los materiales descritos anteriormente los que tienen más influencia en un buen habitad para el vector triatoma son: material del piso de la vivienda, material de la pared de la vivienda y material del techo de la vivienda, ésto teniendo en cuenta la conferencia presentada por la Doctora Any Laura Flores Villegas con el nombre "Alternativas en el control de los triatomas" (Simposio Latinoamericano de Cardiopatía Chagásica, 2018).

4.3.3 Proceso de extracción de conocimientos – Minería de datos, modelos, interpretación y evaluación

En esta fase del proceso de extracción de conocimientos se tiene un archivo .XLS con el 80% de registros los cuales para este proyecto se usarán para entrenar y probar dos algoritmos: J48 e ID3, para hacer ésto se usan dos plataformas por un lado se emplea Weka, plataforma de software libre que sirve para minería de datos y por otro lado también se emplea RapidMiner Studio, plataforma que no es de software libre y necesita de licencia para poder usarse, para ello se utiliza una licencia de tipo estudiantil.

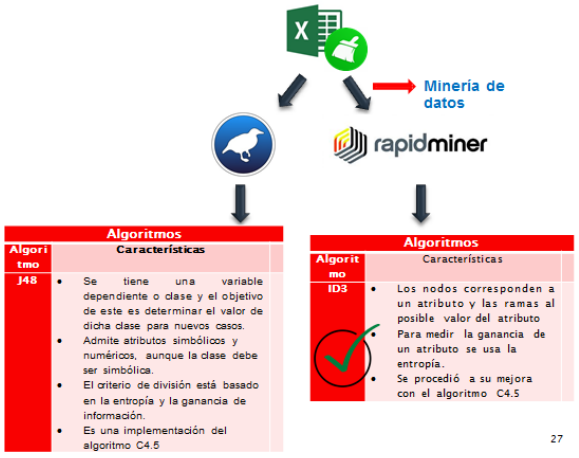

Figura 19. Proceso de extracción de conocimientos (minería de datos)
Fuente: John Leandro Mejía Castro

Para poder ejecutar el análisis de los registros con los algoritmos, por parte de la plataforma Weka el algoritmo J48 y por parte de la plataforma RapidMiner el algoritmo ID3, de dicho análisis se obtiene y selecciona el siguiente árbol o modelo generado por el algoritmo ID3:

- Árbol generado por algoritmo ID3 en plataforma RapidMiner:

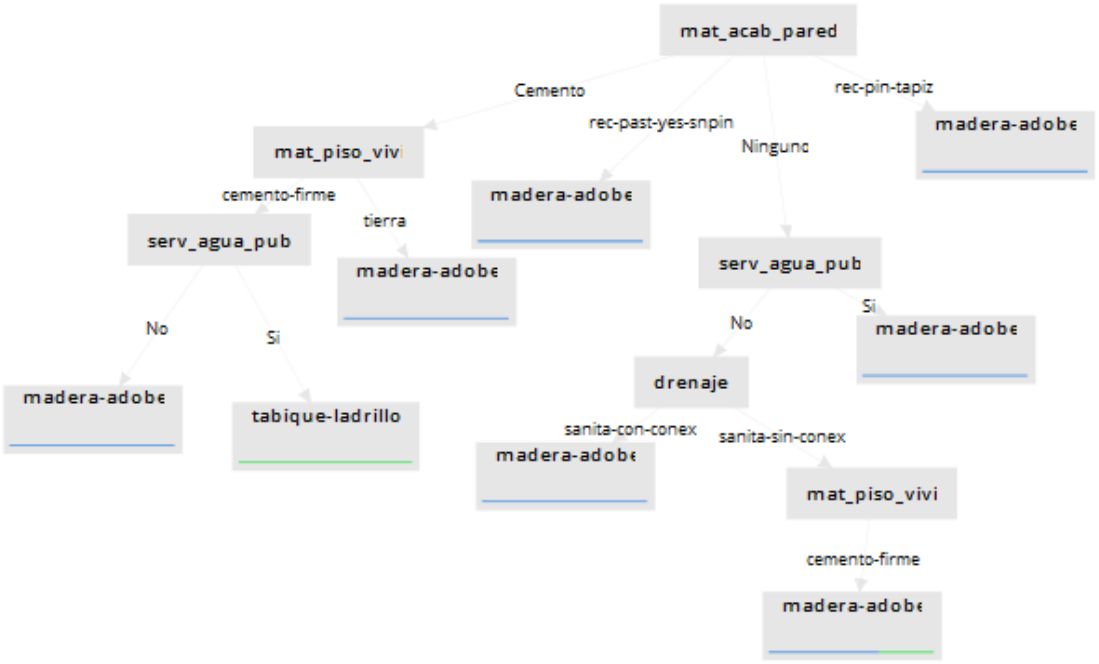

Figura 20. Árbol generado por algoritmo ID3
Fuente: John Leandro Mejía Castro

Para poder realizar el análisis mediante la plataforma RapidMiner se debe convertir el archivo .XLS a un archivo .CSV delimitado por comas y así poder generar el árbol mostrado en la figura anterior el cual tiene como nodo raíz el material pared de la vivienda. Se hace importante que para observar la correcta clasificación del algoritmo se entrene y pruebe, para ello se llevan a cabo los siguientes pasos:

a. Entrenar el algoritmo con el 80% de los datos.

El entrenamiento del algoritmo se lleva a cabo en el lenguaje de programación Java Enterprise Edition puesto que el sistema experto es una plataforma web, por ende se hace uso de una API de Weka para poder implementar y probar dicho algoritmo, como se puede ver en la siguiente imagen:

```
br = new BufferedReader(new FileReader("D:\\Mis Documentos\\Documents\\NetBeansProjects\\Test1_Weka\\build\\classes\\archivo\\seleccion_80_porc_prueba.arff"));
String line = br.readLine();
while (null != line) {
    String[] fields = line.split(SEPARADOR);
    anali.setValue(0, fields[0]);
    anali.setValue(1, fields[1]);
    anali.setValue(2, fields[2]);
    anali.setValue(3, fields[3]);
    anali.setValue(4, fields[4]);
    anali.setValue(5, fields[5]);
    anali.setValue(6, fields[6]);

    double probabilidad[] = alg.distributionForInstance(anali);
    System.out.println(fields[0] + "," + fields[1] + "," + fields[2] + "," + fields[3] + "," +
            fields[4] + "," + fields[5] + "," + fields[6]+ "," + probabilidad[0]);
```

Figura 21. Entrenamiento de algoritmo en Java EE, con el 80% de registros
Fuente: John Leandro Mejía Castro

Para que los registros puedan ser clasificados por la API deben estar en un formato .arff, de esta manera el algoritmo podrá clasificar los diferentes registros. En la imagen anterior se puede notar la ruta donde se encuentra el archivo .arff con el 80% de los datos los cuales son analizados mediante el algoritmo ID3 y clasificados con una probabilidad de: 1.0 si el registro es posiblemente positivo a la enfermedad de Chagas, o por el contrario 0.0 si el registro es negativo a la enfermedad. Cada uno de esos registros es clasificado como se puede observar a continuación:

```
Output - Test1_Weka (run)
run:
AMACUZAC,cemento-firme,madera-adobe,carton-hule,Ninguno,No,letrina,1.0
AMACUZAC,cemento-firme,panel-concreto,tabique-ladrillo,rec-pin-tapiz,No,sanita-con-conex,0.0
AMACUZAC,cemento-firme,tabique-ladrillo,lm-metalica-fbvid,Cemento,No,sanita-sin-conex,0.0
AMACUZAC,cemento-firme,madera-adobe,lm-metalica-fbvid,Ninguno,No,sanita-sin-conex,1.0
AMACUZAC,cemento-firme,tabique-ladrillo,tabique-ladrillo,rec-pin-tapiz,No,sanita-con-conex,0.0
AMACUZAC,cemento-firme,tabique-ladrillo,lm-metalica-fbvid,Ninguno,No,letrina,0.0
AMACUZAC,cemento-firme,madera-adobe,lm-metalica-fbvid,Cemento,No,sanita-con-conex,1.0
AMACUZAC,cemento-firme,tabique-ladrillo,lm-metalica-fbvid,rec-past-yes-snpin,Si,sanita-sin-conex,0.0
AMACUZAC,madera-mosaico,tabique-ladrillo,tabique-ladrillo,rec-pin-tapiz,No,sanita-con-conex,0.0
AMACUZAC,cemento-firme,tabique-ladrillo,tabique-ladrillo,rec-pin-tapiz,No,sanita-con-conex,0.0
```

Figura 22. Clasificación de registros de entrenamiento en Java EE
Fuente: John Leandro Mejía Castro

b. Determinar eficacia del entrenamiento.

Se hace muy importante ver la eficacia del entrenamiento del algoritmo con los datos de prueba es decir los 13.152 datos o 80%, puesto que al ver la clasificación de estos se puede observar el porcentaje de eficacia del entrenamiento y por ende se prevé la eficacia con el 20% de los datos o 3.288 registros.

Para poder determinar la eficacia descrita anteriormente se procede a tomar los datos obtenidos en la Figura 12 y se llevan a un archivo .XLS donde se compara la clasificación obtenida por el algoritmo ID3 con la clasificación hecha por la persona experta, tal como se puede observar en la Figura 23.

	Prueba ID3								REALIDAD							
2	municipio	mat_piso_vivi	mat_pared	mat_techo	mat_acab_pared	serv_agu	drenaje	clasificacio	municipio	mat_piso_vivi	mat_pared	mat_techo	mat_acab_pare	serv_a	drenaje	clasificacio
3	AMACUZAC	cemento-firme	madera-adobe	carton-hule	Ninguno	No	letrina	1.0	AMACUZAC	cemento-firme	madera-adobe	carton-hule	Ninguno	No	letrina	P
4	AMACUZAC	cemento-firme	panel-concreto	tabique-ladrillo	rec-pin-tapiz	No	sanita-con-conex	0.0	AMACUZAC	cemento-firme	panel-concreto	tabique-ladrillo	rec-pin-tapiz	No	sanita-con-cone	N
5	AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Cemento	No	sanita-sin-conex	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Cemento	No	sanita-sin-conex	N
6	AMACUZAC	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	1.0	AMACUZAC	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	P
7	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	rec-pin-tapiz	No	sanita-con-conex	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	rec-pin-tapiz	No	sanita-con-cone	N
8	AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Ninguno	No	letrina	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Ninguno	No	letrina	N
9	AMACUZAC	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	No	sanita-con-conex	1.0	AMACUZAC	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	No	sanita-con-cone	P
10	AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	rec-past-yes-snpin	Si	sanita-sin-conex	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	rec-past-yes-s	Si	sanita-sin-conex	N
11	AMACUZAC	madera-mosaico	tabique-ladrillo	tabique-ladrillo	rec-pin-tapiz	No	sanita-con-conex	0.0	AMACUZAC	madera-mosaico	tabique-ladrillo	tabique-ladrillo	rec-pin-tapiz	No	sanita-con-cone	N
12	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	rec-pin-tapiz	No	sanita-con-conex	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	rec-pin-tapiz	No	sanita-con-cone	N
13	AMACUZAC	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-pin-tapiz	No	sanita-con-conex	1.0	AMACUZAC	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-pin-tapiz	No	sanita-con-cone	P
14	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	Cemento	No	sanita-con-conex	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	Cemento	No	sanita-con-cone	N
15	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	Cemento	No	sanita-sin-conex	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	Cemento	No	sanita-sin-conex	N
16	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	Cemento	No	sanita-sin-conex	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	Cemento	No	sanita-sin-conex	N
17	AMACUZAC	tierra	tabique-ladrillo	lm-metalica-fbvid	Cemento	Si	sanita-sin-conex	1.0	AMACUZAC	tierra	tabique-ladrillo	lm-metalica-fbvid	Cemento	Si	sanita-sin-conex	P
18	AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	N
19	AMACUZAC	madera-mosaico	tabique-ladrillo	tabique-ladrillo	Cemento	No	sanita-sin-conex	0.0	AMACUZAC	madera-mosaico	tabique-ladrillo	tabique-ladrillo	Cemento	No	sanita-sin-conex	N
20	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	Ninguno	No	sanita-sin-conex	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	Ninguno	No	sanita-sin-conex	N
21	AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	N
22	AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	rec-pin-tapiz	No	sanita-sin-conex	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	rec-pin-tapiz	No	sanita-sin-conex	N
23	AMACUZAC	cemento-firme	madera-adobe	carton-hule	Cemento	No	sanita-sin-conex	1.0	AMACUZAC	cemento-firme	madera-adobe	carton-hule	Cemento	No	sanita-sin-conex	P
24	AMACUZAC	tierra	tabique-ladrillo	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	1.0	AMACUZAC	tierra	tabique-ladrillo	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	P
25	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	Cemento	No	sanita-sin-conex	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	Cemento	No	sanita-sin-conex	N
26	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	Cemento	No	sanita-con-conex	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	Cemento	No	sanita-con-cone	N
27	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	Cemento	No	sanita-sin-conex	0.0	AMACUZAC	cemento-firme	tabique-ladrillo	tabique-ladrillo	Cemento	No	sanita-sin-conex	N
28	ATLATLAHUCAN	madera-mosaico	madera-adobe	lm-metalica-fbvid	Cemento	Si	sanita-con-conex	1.0	ATLATLAHUCA	madera-mosaico	madera-adobe	lm-metalica-fbvid	Cemento	Si	sanita-con-cone	P
29	ATLATLAHUCAN	cemento-firme	madera-adobe	tabique-ladrillo	Cemento	No	sanita-sin-conex	1.0	ATLATLAHUCA	cemento-firme	madera-adobe	tabique-ladrillo	Cemento	No	sanita-sin-conex	P
30	ATLATLAHUCAN	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	Si	sanita-con-conex	1.0	ATLATLAHUCA	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	Si	sanita-con-cone	P
31	ATLATLAHUCAN	cemento-firme	madera-adobe	tabique-ladrillo	Cemento	No	sanita-sin-conex	1.0	ATLATLAHUCA	cemento-firme	madera-adobe	tabique-ladrillo	Cemento	No	sanita-sin-conex	P
32	ATLATLAHUCAN	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Ninguno	Si	letrina	0.0	ATLATLAHUCA	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Ninguno	Si	letrina	N
33	ATLATLAHUCAN	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	Si	sanita-sin-conex	1.0	ATLATLAHUCA	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	Si	sanita-sin-conex	P
34	ATLATLAHUCAN	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Cemento	No	letrina	0.0	ATLATLAHUCA	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Cemento	No	letrina	N
35	ATLATLAHUCAN	cemento-firme	tabique-ladrillo	tabique-ladrillo	rec-past-yes-snpin	No	sanita-sin-conex	0.0	ATLATLAHUCA	cemento-firme	tabique-ladrillo	tabique-ladrillo	rec-past-yes-s	No	sanita-sin-conex	N

Figura 23. Comparación de persona experta vs algoritmo de los datos de entrenamiento
Fuente: John Leandro Mejía Castro

Para poder hacer un correcto análisis de que tan eficaz es el entrenamiento se debe observar la clasificación, de tal forma que se obtenga el: número de registros que en la realidad si pueden ser posibles portadores de Chagas y en el sistema experto también sean posibles portadores, o que en la realidad puedan ser posibles portadores pero que el sistema experto no los considere como tal, o que por el contrario en la realidad o el concepto del experto sean negativos y el sistema los considere como positivos. Para lo descrito anteriormente se hace fundamental realizar una matriz de confusión y así obtener el porcentaje de eficacia del algoritmo y por ende que porcentaje de eficacia tendría posteriormente el sistema experto al analizar el restante 20% de registros, en la Figura 24 se presenta la matriz de confusión para los 13.152 (80%) registros.

		Sistema Experto		
	n = 13152	Positivos	Negativos	
Realidad	Positivos	100%	0%	
	4794	4794	0	4794
	Negativos	0%	100%	
	8358	0	8358	8358
		4794	8358	

Figura 24. Matriz de confusión del 80% de registros, entrenamiento
Fuente: John Leandro Mejía Castro

Al analizar la matriz de confusión que se encuentra en la Figura 24, se puede observar la eficacia del algoritmo puesto que el número de registros determinado por el algoritmo como positivos es el mismo que el determinado por la persona experta, 4794 registros, e igualmente el número de registros determinados como negativos por la persona experta es el mismo número de registros determinado por el algoritmo, 8353 registros. El porcentaje de eficacia del entrenamiento es del 100%, lo que genera la confianza para probar el modelo sobre el restante de los registros el 20%.

c. Probar registros faltantes.

El siguiente paso es probar los registros faltantes es decir los 3.288 registros que representan el 20% de los datos, esto se lleva a cabo en el lenguaje de programación Java Enterprise Edition, primero se debe entrenar el algoritmo ID3 y después proceder a probar el 20% de registros, tal como se puede observar en la Figura 25.

```
try {
    String ruta_80_poc = "D:\\Mis Documentos\\Documents\\NetBeansProjects\\"+
            "Test1_Weka\\build\\classes\\archivo\\seleccion_80_porc_entrenamiento.arff";
    ConverterUtils.DataSource source = new ConverterUtils.DataSource(ruta_80_poc);
    Instances inst = source.getDataSet();
    inst.setClassIndex(7);

    Id3 alg = new Id3();
    alg.buildClassifier(inst);

    Instance anali = new Instance(7);
    anali.setDataset(inst);

    String ruta_20_porc = "D:\\Mis Documentos\\Documents\\NetBeansProjects\\"
            + "Test1_Weka\\build\\classes\\archivo\\seleccion_20_porc_pruebas.arff";
    br = new BufferedReader(new FileReader(ruta_20_porc));
    String line = br.readLine();
    while (null != line) {
        String[] fields = line.split(SEPARADOR);
        anali.setValue(0, fields[0]);
        anali.setValue(1, fields[1]);
        anali.setValue(2, fields[2]);
        anali.setValue(3, fields[3]);
        anali.setValue(4, fields[4]);
        anali.setValue(5, fields[5]);
        anali.setValue(6, fields[6]);

        double probabilidad[] = alg.distributionForInstance(anali);
        System.out.println(fields[0] + "," + fields[1] + "," + fields[2] + "," + fields[3] +
                "," + fields[4] + "," + fields[5] + "," + fields[6]+ "," + probabilidad[0]);
```

Figura 25. Prueba de registros faltantes, 20%
Fuente: John Leandro Mejía Castro

Para realizar la clasificación de los registros mediante la API de Weka es necesario que los archivos del 80% y 20% de los datos se encuentren en formato .arff de esta manera se puede seleccionar primero el archivo del 80% de los datos para entrenar el algoritmo, esto mediante la ruta donde se encuentra dicho archivo, tal como se puede observar en la Figura 25, posteriormente se debe seleccionar el archivo .arff con el 20% de los datos para proceder a clasificar cada uno de estos registros, tal como se puede observar a continuación:

```
Output - Test1_Weka (run)
YECAPIXTLA,cemento-firme,tabique-ladrillo,lm-metalica-fbvid,Ninguno,No,letrina,0.0
OCUITUCO,madera-mosaico,madera-adobe,lm-metalica-fbvid,rec-past-yes-snpin,No,sanita-sin-conex,1.0
OCUITUCO,cemento-firme,madera-adobe,lm-metalica-fbvid,Cemento,No,sanita-sin-conex,1.0
YECAPIXTLA,cemento-firme,madera-adobe,lm-metalica-fbvid,Ninguno,No,letrina,1.0
```

Figura 26. Clasificación de los registros de prueba, 20%
Fuente: John Leandro Mejía Castro

Se hace muy importante ver la eficacia de la clasificación a los registros de prueba, para ello se procede a tomar los datos obtenidos en la Figura 26 y se llevan a un archivo .XLS donde se debe proceder a realizar el siguiente paso que es construir la matriz de confusión.

d. Construir la matriz de confusión

La clasificación de los 3.288 registros de prueba en el archivo .XLS, tal como se muestra en la Figura 27 , se procede a realizar la matriz de confusión de tal forma que se obtenga el; número de registros que en la realidad si pueden ser posibles portadores de Chagas y en el sistema experto también sean posibles portadores, o que en la realidad puedan ser posibles portadores pero que el sistema experto no los considere como tal, o que por el contrario en la realidad, entiéndase como el concepto del experto sean negativos y el sistema los considere como positivos, dicha matriz se puede observar en la Figura 28.

	Prueba ID3								REALIDAD							
2	municipio	mat_piso_vivi	mat_pared	mat_techo	mat_acab_pared	serv_agua	drenaje	Clasificacio	municipio	mat_piso_vivi	mat_pared	mat_techo	mat_acab_pared	serv_agua	drenaje	Clasificacio
3	YECAPIXTLA	tierra	madera-adobe	lm-metalica-fbvid	Ninguno	No	letrina	1.0	YECAPIXTLA	tierra	madera-adobe	lm-metalica-fbvid	Ninguno	No	letrina	P
4	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	letrina	1.0	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	letrina	P
5	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	No	letrina	1.0	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	No	letrina	P
6	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	1.0	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	P
7	YECAPIXTLA	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	0.0	YECAPIXTLA	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	N
8	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-pin-tapiz	No	sanita-sin-conex	1.0	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-pin-tapiz	No	sanita-sin-conex	P
9	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	letrina	1.0	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	letrina	P
10	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-pin-tapiz	No	letrina	1.0	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-pin-tapiz	No	letrina	P
11	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	sanita-con-conex	1.0	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	sanita-con-cone	P
12	OCUITUCO	tierra	carton-hule	carton-hule	Ninguno	No	letrina	1.0	OCUITUCO	tierra	carton-hule	carton-hule	Ninguno	No	letrina	P
13	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	Si	letrina	1.0	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	Si	letrina	P
14	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-past-yes-snp	No	letrina	1.0	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-past-yes-snpir	No	letrina	P
15	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	1.0	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	sanita-sin-conex	P
16	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-pin-tapiz	No	sanita-sin-conex	1.0	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-pin-tapiz	No	sanita-sin-conex	P
17	YECAPIXTLA	tierra	madera-adobe	lm-metalica-fbvid	Ninguno	No	letrina	1.0	YECAPIXTLA	tierra	madera-adobe	lm-metalica-fbvid	Ninguno	No	letrina	P
18	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-past-yes-snp	No	letrina	1.0	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-past-yes-snpir	No	letrina	P
19	OCUITUCO	cemento-firme	madera-adobe	madera-adobe	Ninguno	Si	letrina	1.0	OCUITUCO	cemento-firme	madera-adobe	madera-adobe	Ninguno	Si	letrina	P
20	YECAPIXTLA	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Ninguno	No	letrina	0.0	YECAPIXTLA	cemento-firme	tabique-ladrillo	lm-metalica-fbvid	Ninguno	No	letrina	N
21	OCUITUCO	madera-mosaico	madera-adobe	lm-metalica-fbvid	rec-past-yes-snp	No	sanita-sin-conex	1.0	OCUITUCO	madera-mosaico	madera-adobe	lm-metalica-fbvid	rec-past-yes-snpir	No	sanita-sin-conex	P
22	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	No	sanita-sin-conex	1.0	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	No	sanita-sin-conex	P
23	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	letrina	1.0	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	letrina	P
24	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-pin-tapiz	Si	sanita-con-conex	1.0	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-pin-tapiz	Si	sanita-con-cone	P
25	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	No	sanita-sin-conex	1.0	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	No	sanita-sin-conex	P
26	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	letrina	1.0	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	letrina	P
27	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	Si	letrina	1.0	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	Si	letrina	P
28	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	No	sanita-sin-conex	1.0	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	No	sanita-sin-conex	P
29	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-past-yes-snp	No	letrina	1.0	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-past-yes-snpir	No	letrina	P
30	YECAPIXTLA	cemento-firme	tabique-ladrillo	tabique-ladrillo	rec-past-yes-snp	No	letrina	0.0	YECAPIXTLA	cemento-firme	tabique-ladrillo	tabique-ladrillo	rec-past-yes-snpir	No	letrina	N
31	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-pin-tapiz	No	letrina	1.0	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-pin-tapiz	No	letrina	P
32	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	letrina	1.0	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	Ninguno	No	letrina	P
33	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-past-yes-snp	No	letrina	1.0	YECAPIXTLA	cemento-firme	madera-adobe	lm-metalica-fbvid	rec-past-yes-snpir	No	letrina	P
34	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	No	letrina	1.0	OCUITUCO	cemento-firme	madera-adobe	lm-metalica-fbvid	Cemento	No	letrina	P

Figura 27. Comparación de persona experta vs algoritmo de los registros de prueba
Fuente: John Leandro Mejía Castro

		Sistema Experto		
	n = 3288	Positivos	Negativos	
Realidad	Positivos	100%	0%	
	1143	1143	0	1143
	Negativos	0%	100%	
	2145	0	2145	2145
		1143	2145	

Figura 28. Matriz de confusión del 20% de registros, prueba
Fuente: John Leandro Mejía Castro

e. Analizar y concluir

Al analizar la matriz de confusión que se encuentra en la Figura 28, se puede observar la eficacia del algoritmo ID3 puesto que el número de registros determinado por el algoritmo como positivos es el mismo que el determinado por la persona experta, 1143 registros, e igualmente el número de registros determinados como negativos por la persona experta es el mismo número de registros determinado por el algoritmo, 2145 registros para un total de 3.288 registros que son los registros totales de prueba, es importante notar que el sistema experto no clasifica ningún registro diferente a como lo determina la persona experta es decir hay un 100% de eficacia en la clasificación de los registros de prueba. De esta forma se determina que el proceso de extracción de conocimiento ha sido exitoso para este proyecto.

4.4. Documentación del código

El desarrollo del sistema experto para posibles portadores de Chagas se implementó mediante la plataforma de programación Java web usando como arquitectura del sistema la arquitectura Modelo Vista Controlador (MVC). El proyecto es creado como un proyecto Web Aplication y lleva por nombre *SepchaV1.0.*

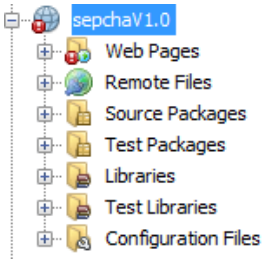

Figura 29. Creación y nombre del sistema experto para posibles portadores de la enfermedad de Chagas
Fuente: John Leandro Mejía Castro

Debido a la arquitectura del sistema MVC se crean diferentes paquetes donde se tengan por separado las vistas los datos o modelos y los controladores, a continuación se explican cada una de estas capas:

4.4.1 Documentación del código – Capa Vista

Para crear la capa Vista en la cual se encuentran las vistas del Sistema experto para posibles portadores de Chagas se usan archivos .jsp los cuales se encuentran almacenados dentro de una carpeta que tiene por nombre "vistas" y que contienen las etiquetas HTML y propiedades Bootstrap necesarias para crear interfaces intuitivas, en la Figura 30 se puede observar la estructuración de esta capa en el proyecto *SepchaV1.0.*

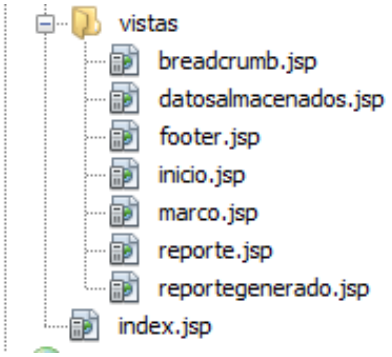

Figura 30. Estructura de capa vista en proyecto SepchaV1.0
Fuente: John Leandro Mejía Castro

En el directorio raíz se encuentra un archivo .jsp llamado "index.jsp", este archivo contiene lo presentado en la Figura 31.

```
<%--
    Document   : index
    Created on : 18/09/2018, 02:43:03 PM
    Author     : USUARIO
--%>

<%  int index=1;
    response.sendRedirect("vistas/marco.jsp?index=" +index);     %>
```

Figura 31. Archivo "index.jsp"
Fuente: John Leandro Mejía Castro

Cuando el usuario ingresa a la URL del sistema experto apunta directamente al archivo "index.jsp" en el directorio raíz del proyecto, en este archivo se implementa se tiene un método que remite directamente a la vista principal del sistema experto. Para poder hacer un correcto manejo de las vistas y tener una estructura que permita tener modularidad en cada una de las interfaces de los módulos del sistema se crea un marco el cual será siempre único y llamará a las demás interfaces del sistema, dicho marco en el proyecto lleva por nombre "marco.jsp" el cual se encuentra dentro de la carpeta vistas y es el archivo al cual apunta el archivo "index.jsp" cuando el usuario ingresa a la URL principal del sistema.

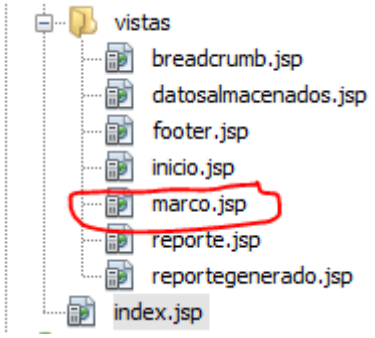

Figura 32. Estructura de capa vista, archivo "marco.jsp"
Fuente: John Leandro Mejía Castro

Para que el archivo "marco.jsp" pueda llamar a las demás interfaces del sistema debe crearse un identificador de estas, dicho indicador permitirá mostrar al usuario la interfaz que desea visualizar, en la Tabla 15 Identificadores de vistas (interfaces).

Tabla 15. Identificadores de vistas (interfaces)

Archivo Vistas	Identificador
inicio.jsp	1
reporte.jsp	2
datosalmacenados.jsp	3
reportegenerado.jsp	4

Fuente: John Leandro Mejía Castro.

Para que el archivo "marco.jsp" llame a las demás interfaces se usan los identificadores como se presenta en la Figura 33.

```
<div class="content-wrapper">
    <div class="container-fluid">
        <jsp:include page="breadcrumb.jsp" />
        <div class="row">
            <div class="col-12">

                <% if (index.equals("1")) {%>
                <jsp:include page="inicio.jsp" />

                <% } %>

                <% if (index.equals("2")) {%>
                <jsp:include page="reporte.jsp"/>
                <% }%>

                <% if (index.equals("3")) {%>
                <jsp:include page="datosalmacenados.jsp"/>
                <% }%>
                <% if (index.equals("4")) {%>
                <jsp:include page="reportegenerado.jsp"/>
                <% }%>
```

Figura 33. Uso de identificadores para mostrar interfaces en "marco.jsp"
Fuente: John Leandro Mejía Castro

La interfaz de inicio del sistema experto se encuentra en el archivo “inicio.jsp” la cual contiene la presentación de bienvenida al sistema experto, el nombre del sistema y el nombre de la institución que implementa el proyecto, tal como se puede observar en la Figura 34.

```
<div class="col-xs-12">
    <div class="headerh1">
        <h2 class="titulo1">BIENVENIDO</h2>
    </div>
</div>

<div class="col-xs-12">
    <div class="headerh1">
        <h2 class="titulo2">SEPCHA</h2>
        <h3 class="titulo3">(Sistema experto para posibles portadores de Chagas)</h3>

    </div>
</div>

<div class="col-xs-12">
    <div class="centerImage">
        <img alt="Logo Upemor" src="../imgs/inicioUpemor.png" class="logoUpemor"/>
    </div>
    <h3 class="titulo4">Universidad Politécnica del Estado de Morelos</h3>
</div>
```

Figura 34. Archivo "inicio.jsp"
Fuente: John Leandro Mejía Castro

Para que el usuario tenga una correcta navegabilidad en la aplicación se implementa un menú breadcrumb el cual le permita a este conocer su ubicación en el sistema experto, para esto se crea el archivo "breadcrum.jsp" el cual mediante el uso de los identificadores de las interfaces, mostrará al usuario su ubicación en el sistema experto, tal como se puede observar en la Figura 35.

```
<% String index = request.getParameter("index"); %>

<% if ( index.equals("1") ) {%>

<ol class="breadcrumb">
    <li class="breadcrumb-item active">
        <a href="marco.jsp?index=<% out.print(1); %>">Inicio</a>
    </li>
</ol>

<% } %>

<% if ( index.equals("2") ) {%>

<ol class="breadcrumb">
    <li class="breadcrumb-item active"><a href="marco.jsp?index=<% out.print(1); %>">Inicio</a></li>
    <li class="breadcrumb-item breadcrumfont"><a href="marco.jsp?index=<% out.print(2); %>">Reporte</a></li>
</ol>

<% }%>

<% if ( index.equals("3") ) {%>

<ol class="breadcrumb">
    <li class="breadcrumb-item active"><a href="marco.jsp?index=<% out.print(1); %>">Inicio</a></li>
    <li class="breadcrumb-item breadcrumfont"><a href="marco.jsp?index=<% out.print(3); %>">Datos Almacenados</a></li>
</ol>

<% }%>

<% if ( index.equals("4") ) {%>
```

Figura 35. Archivo "breadcrumb.jsp"
Fuente: John Leandro Mejía Castro

Para que el usuario del sistema experto pueda conocer los datos almacenados en la base de datos se deben presentar de forma intuitiva para ello se crea el archivo "datosalmacenados.jsp" donde se tienen las etiquetas necesarias para presentar los datos almacenados en una tabla, también se hace uso de JavaScript para poder mostrar estos registros, lo anterior se presenta en la Figura 36.

```
<div class="row">
    <div class="col-lg-12">
        <div class="headerh2">
            <h3 class="page-header">Datos Almacenados</h3>
        </div>

        <div class="panel-body">

            <table id="registros" class="display" style="width:100%">
                <thead>
                    <tr>

                        <th>Num</th>
                        <th>Municipio</th>
                        <th>Mat. piso vivienda</th>
                        <th>Mat. de pared</th>
                        <th>Mat. techo</th>
                        <th>Acabado pared</th>
                        <th>Servicio agua potable</th>
                        <th>Servicio drenaje</th>
                    </tr>
                </thead>
```

Figura 36. Archivo "datosalmacenados.jsp"
Fuente: John Leandro Mejía Castro

```
    <script type="text/javascript">
        $(document).ready(function () {

            var tabla = $("#registros").DataTable({
                ajax: {
                    method: "POST",
                    url: "CargarDatosTabla",
                    dataSrc: "datos"
                },
                columns: [
                    {"data": "Num"},
                    {"data": "Municipio"},
                    {"data": "Matepisovivienda"},
                    {"data": "Matedepared"},
                    {"data": "Matetecho"},
                    {"data": "Acabadopared"},
                    {"data": "Servaguapotable"},
                    {"data": "Servdrenaje"}

                ]

            });
        });
    </script>
```

Figura 37. Presentación de registros en tabla de archivo "datosalmacenados.jsp"
Fuente: John Leandro Mejía Castro

El usuario del sistema experto puede seleccionar la zona o municipio a la cual desee hacer el análisis respectivo de los posibles portadores del Chagas esto lo puede hacer por medio del archivo “reporte.jsp” donde se crea un objeto listamunicipo donde se encuentra almacenados los diferentes municipios a seleccionarse en un combo para proceder a que el sistema experto entregue el reporte pertinente.

```
<div class="col-lg-12 formSeleccion">
    <div class="input-group">
        <label for="zona">Municipio:</label>
        <div class="col-lg-4 col-md-4">

            <div class="form-group">

                <select class="form-control" id="sel1" name="zona">

                    <%
                        CrudReporte municipio = new CrudReporte();
                        LinkedList<Municipio> listamunicipio = municipio.readMunicipio();

                        if (listamunicipio.size() == 0) {
                    %><option>-------------</option><%
                    } else {

                        for (int i = 0; i < listamunicipio.size(); i++) {
                    %><option value="<% out.println(listamunicipio.get(i).getNombre_municipio()); %>"><% out.println(listamunicipio.get(i).getNombre_mu
                            }
                        }%>

                </select>
            </div>

        </div>

        <button type="submit" class="btn btn-outline-primary" id="boton1" >
```

Figura 38. Archivo "reporte.jsp"
Fuente: John Leandro Mejía Castro

Después de que el usuario selecciona la zona o municipio de la cual desea obtener el reporte del sistema experto se le presenta un reporte intuitivo con gráficas y tablas las cuales le presentan la información de la clasificación efectuada, para en el archivo "reportegenerado.jsp" se reciben ciertos datos los cuales se almacenan en variables que posteriormente serán mostradas en el reporte de forma intuitiva, tal como se observa en la Figura 39.

```
<%
    String muestraPoblacional = request.getParameter("muestrap");
    String numeroregistroEntrenamiento = request.getParameter("numregistentre");
    String conteoRegistros = request.getParameter("contregistro");
    String registrosPositivos = request.getParameter("registposit");
    String registrosNegativos = request.getParameter("registnegat");
    String registropisocemfirme = request.getParameter("pisocemfirme");
    String registropisomadmosaico = request.getParameter("pisomadmosaico");
    String registropisotierra = request.getParameter("pisotierra");
    String registroparedmadadobe = request.getParameter("paredmadadobe");
    String registroparedpanconc = request.getParameter("paredpanconc");
    String registroparedtabladri = request.getParameter("paredtabladri");
    String registroparedcartonhule = request.getParameter("paredcartonhule");
    String registroparedlmmetfbvid = request.getParameter("paredlmmetfbvid");
    String registroparedotro =  request.getParameter("paredotro");
    String registrotechocarthule = request.getParameter("techocarthule");
    String registrotechotabladrillo = request.getParameter("techotabladrillo");
    String registrotecholmmetfbvid = request.getParameter("techolmmetfbvid");
    String registrotechopancon = request.getParameter("techopancon");
    String registrotechomadadob = request.getParameter("techomadadob");
    String registroacabparedninguno = request.getParameter("acabparedninguno");
    String registroacabparedrecpintapiz = request.getParameter("acabparedrecpintapiz");
    String registroacabparedcem = request.getParameter("acabparedcem");
    String registroacabparedrecpastyessnpin = request.getParameter("acabparedrecpastyessnpin");
    String registroservaguano = request.getParameter("servaguano");
    String registroservaguasi = request.getParameter("servaguasi");
    String registroservdrenletrina = request.getParameter("servdrenletrina");
    String registroservdrensanconconex = request.getParameter("servdrensanconconex");
    String registroservdrensansinconex = request.getParameter("servdrensansinconex");
    String registroservdrennoservsan = request.getParameter("servdrennoservsan");
%>
```

Figura 39. Archivo "reportegenerado.jsp" recibir datos en variables
Fuente: John Leandro Mejía Castro

De los datos almacenados en las variables mostradas en la Figura 39, se proceden a crear una serie de gráficas las cuales se muestran como imágenes al usuario del sistema experto, al igual que tablas con los datos de estas variables, tal como se puede observar en la Figura 40.

```
<div class="panel-heading">
    Portadores de la enfermedad del Chagas
</div>
<div class="panel-body">
    <div class="tablaPortadores col-lg-8">
        <img alt="Gráfica portadores Chagas" src="../imgs/portadoresdechagas_torta.jpg" width="510" height="405"/>
    </div>

    <div class="col-lg-8">
        <div class="table100 ver1 m-b-110">
            <table data-vertable="ver1" id="dev-table">
                <thead>
                    <tr class="row100 head">
                        <th class="column100 column2" data-column="column1"><center>Estado</center></th>
                <th class="column100 column2" data-column="column2"><center>Cantidad</center></th>

                </tr>
                </thead>
                <tbody>

                    <tr class="row100">
                        <td class="column100 column2" data-column="column1">Positivos</td>
                        <td class="column100 column2 filatabla" data-column="column2"><center><% out.println(registrosPositivos); %></center></td>

                </tr>
```

Figura 40.Archivo "reportegenerado.jsp" presentación de gráficas y tablas
Fuente: John Leandro Mejía Castro

4.4.2 Documentación del código – Capa Modelo

Para crear la capa Modelo en el proyecto *sepchaV1.0* se crean los paquetes *Model*, *Model_crud, Conexionbd, BDRespaldo,* los cuales contienen la información sobre los datos. El paquete Model contiene las clases; Municipio, Registro y Reporte, el paquete Model_crud contiene las clases; CrudRegistro y CrudReporte, el paquete Conexionbd contiene la clase Conexión y la clase prueba y el paquete BDRespaldo contiene la clase Respaldobd, tal como se muestra en la Figura 41.

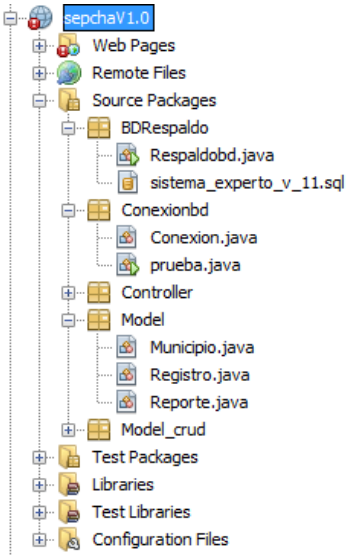

Figura 41. Estructura de capa modelo en proyecto SepchaV1.0
Fuente: John Leandro Mejía Castro

La clase Municipio contiene los datos referentes al nombre del municipio o zona y un código para distinguirlos de todos los demás ésta contiene un constructor que recibe dos parámetros que son los descritos anteriormente y también tiene los get y los set de los atributos descritos anteriormente, tal como se ve en la Figura 42.

```
public class Municipio {

    int cod_municipio;
    String nombre_municipio;

    public Municipio() {
    }

    public Municipio(int cod_municipio, String nombre_municipio) {
        this.cod_municipio = cod_municipio;
        this.nombre_municipio = nombre_municipio;
    }

    public int getCod_municipio() {
        return cod_municipio;
    }

    public void setCod_municipio(int cod_municipio) {
        this.cod_municipio = cod_municipio;
    }

    public String getNombre_municipio() {
        return nombre_municipio;
    }

    public void setNombre_municipio(String nombre_municipio) {
        this.nombre_municipio = nombre_municipio;
    }
```

Figura 42. Clase "Municipio"
Fuente: John Leandro Mejía Castro

La clase Registro contiene los datos referentes al código del registro, nombre del municipio, material del piso de la vivienda, material del acabado de la pared, servicio de agua potable, servicio de drenaje, material de la pared y material del techo de la vivienda, la clase contiene un constructor vacío y otro que recibe los parámetros descritos anteriormente y también tiene los get y los set de los respectivos de esto, tal como se puede observar en la Figura 43.

```
public class Registro {

    public int cod_registro;
    public String nombre_municipio;
    public String nombre_material_piso_vivienda;
    public String nombre_material_acabado_pared;
    public String nombre_servicio_agua_publica;
    public String nombre_drenaje;
    public String nombre_servicio_medico;
    public String nombre_material_pared;
    public String nombre_material_techo;
    public int menor_cinco_anios;
    public int cinco_dieciocho_anios;
    public int dieciocho_sesenta_cuatro_anios;
    public int mayor_sesenta_cinco_anios;

    public Registro(){

    }

    public Registro(int cod_registro, String nombre_municipio, String nombre_material_piso_vivienda, String nombre_material_acabado_pared,
        this.cod_registro = cod_registro;
        this.nombre_municipio = nombre_municipio;
        this.nombre_material_piso_vivienda = nombre_material_piso_vivienda;
        this.nombre_material_acabado_pared = nombre_material_acabado_pared;
        this.nombre_servicio_agua_publica = nombre_servicio_agua_publica;
        this.nombre_drenaje = nombre_drenaje;
        this.nombre_servicio_medico = nombre_servicio_medico;
        this.nombre_material_pared = nombre_material_pared;
        this.nombre_material_techo = nombre_material_techo;
        this.menor_cinco_anios = menor_cinco_anios;
        this.cinco_dieciocho_anios = cinco_dieciocho_anios;
        this.dieciocho_sesenta_cuatro_anios = dieciocho_sesenta_cuatro_anios;
        this.mayor_sesenta_cinco_anios = mayor_sesenta_cinco_anios;
    }

    public int getCod_registro() {
        return cod_registro;
    }

    public void setCod_registro(int cod_registro) {
```

Figura 43. Clase "Registro"
Fuente: John Leandro Mejía Castro

La clase Reporte contiene los datos necesarios para poder generar las gráficas es decir los textos que van a llevar estas, los valores a graficar y el nombre del archivo con el que será llamado la imagen de la gráfica que realice el sistema, para esto la clase contiene un constructor por default que recibe por parámetros los datos descritos anteriormente, también posee tres métodos; uno que es el encargado de crear el gráfico de torta, otro que es el encargado de dar el estilo al gráfico y guardarlo como un archivo JPEG, tal como se muestra en la Figura 44.

```
/**
 * Constructor por default
 *
 * @param textos Textos separados por ;
 * @param valores Valores separados por ;
 * @param nombreArchivo Nombre del archivo
 *
 */
public Reporte(String textos, String valores, String nombreArchivo) {
    this.textosArray = textos.split(";");
    this.valoresArray = valores.split(";");
    this.nombreArchivo = nombreArchivo;

}
```

Figura 44. Clase "Reporte", constructor por defecto
Fuente: John Leandro Mejía Castro

```
/**
 * Metodo que crea en si el grafico de torta
 *
 *
 */
private PieDataset createPieDataset(String[] textos, String[] valores) {
    final double[][] data = new double[valores.length][valores.length];

    for (int i = 0; i < valores.length; i++) {
        for (int j = 0; j < valores.length; j++) {
            data[i][j] = new Double(valores[i]).doubleValue();
        }
    }

    String[] args0 = textos;
    String[] args1 = textos;

    return DatasetUtilities.createPieDatasetForColumn(DatasetUtilities.createCategoryDataset(args0, args1, data), 0);

}
```

Figura 45. Clase "Reporte", método que crea el gráfico de torta
Fuente: John Leandro Mejía Castro

```
/**
 * Metodo que guarda en un archivo el grafico JPEG de tortas
 */
private void generarGraficoDeTorta() throws Exception {
    final PieSectionLabelGenerator labelGenerator = new StandardPieSectionLabelGenerator("{0} = {2}");
    final PieDataset data = createPieDataset(this.textosArray, this.valoresArray);
    final JFreeChart chartTorta = ChartFactory.createPieChart("", data, true, true, true);


    chartTorta.setBackgroundPaint(Color.WHITE);
    chartTorta.setBorderPaint(Color.WHITE);
    PiePlot plot = (PiePlot) chartTorta.getPlot();
    plot.setLabelGenerator(labelGenerator);
    plot.setNoDataMessage("No hay datos disponibles");
    plot.setCircular(false);
    plot.setBackgroundPaint(Color.WHITE);

    try {
        ChartRenderingInfo info = new ChartRenderingInfo(new StandardEntityCollection());

        String path = "C:\\xampp\\htdocs\\sepchaV1.0\\web\\imgs\\" + nombreArchivo + "_torta.jpg";
        ChartUtilities.saveChartAsJPEG(new File(path), 1, chartTorta, 600, 400);

    } catch (Exception e) {

        throw new Exception("Error al generar el archivo JPG");
    }
```

Figura 46. Clase "Reporte", método que da el estilo y guarda el gráfico en formato JPEG
Fuente: John Leandro Mejía Castro

Los datos que deben ser traídos de la base de datos para presentarse al usuario son leídos desde el paquete *Model_crud,* el cual contiene dos clases: CrudRegistro y CrudReporte.

La clase CrudRegistro contiene un método *readRegistro()* el cual retorna una lista con los registros almacenados en la base de datos, para lo cual hace una consulta a la base de datos para obtener los diversos registros a ser presentados al usuario, para lo cual se crea una lista de tipo Registro clase que se describió anteriormente y se añaden los diversos registros a esta, tal como se presenta en la Figura 47.

```
blic LinkedList<Registro> readRegistro() {


  PreparedStatement p_statement = null;

  ResultSet r_set = null;
  ResultSet r_set_nombres = null;


  int cod_municipio;
  int cod_material_piso_vivienda;
  int cod_material_acabado_pared;
  int cod_servicio_agua_publica;
  int cod_drenaje;
  int cod_servicio_medico;
  int cod_material_pared;
  int cod_material_techo;

  LinkedList<Registro> listaRegistro = new LinkedList<>();

  try {

      String query_cons_registro = "SELECT * FROM registro";
      p_statement = conectar().prepareStatement(query_cons_registro);
      r_set = p_statement.executeQuery();

      while (r_set.next()) {

          Registro registro = new Registro();

          registro.setCod_registro(r_set.getInt("COD_REGISTRO"));
          cod_municipio = r_set.getInt("COD_MUNICIPIO");
```

```
      String query_cons_nombres = "SELECT NOMBRE, NOMBRE_MATERIAL, NOMBRE_ACABADO_PARED, ESTADO, NOMBRE_DRENAJE,  NOMBRE_MATERIAL
      p_statement = conectar().prepareStatement(query_cons_nombres);
      r_set_nombres = p_statement.executeQuery();

      while (r_set_nombres.next()) {
          registro.setNombre_municipio(r_set_nombres.getString("NOMBRE").trim());
          registro.setNombre_material_piso_vivienda(r_set_nombres.getString("NOMBRE_MATERIAL").trim());
          registro.setNombre_material_acabado_pared(r_set_nombres.getString("NOMBRE_ACABADO_PARED").trim());
          registro.setNombre_servicio_agua_publica(r_set_nombres.getString("ESTADO").trim());
          registro.setNombre_drenaje(r_set_nombres.getString("NOMBRE_DRENAJE").trim());
          registro.setNombre_material_pared(r_set_nombres.getString("NOMBRE_MATERIAL_PARED").trim());
          registro.setNombre_material_techo(r_set_nombres.getString("NOMBRE_MATERIAL_TECHO").trim());

      }
      r_set_nombres.close();
      p_statement.close();


      listaRegistro.add(registro);
  }
  r_set.close();
  p_statement.close();

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

return listaRegistro;
```

Figura 47. Clase "CrudRegistro", método readRegistro()
Fuente: John Leandro Mejía Castro

La clase CrudReporte contiene un método *readMunicipio()* el cual retorna una lista con los registros almacenados en la base de datos, para ello hace una consulta a la base de datos para obtener los diversos registros a ser presentados al usuario, para lo cual se

crea una lista de tipo Municipio clase que se describió anteriormente y se añaden los diversos registros a esta, tal como se presenta en la Figura 48.

```
public LinkedList<Municipio> readMunicipio(){

    PreparedStatement p_statement = null;
    ResultSet r_set = null;



    LinkedList<Municipio> listaMunicipio = new LinkedList<>();

    try {

        String query_cons_cod_municipio = "SELECT COD_MUNICIPIO, NOMBRE FROM municipio";
        p_statement = conectar().prepareStatement(query_cons_cod_municipio);
        r_set = p_statement.executeQuery();

        while(r_set.next()){

            Municipio municipio = new Municipio();

            municipio.setCod_municipio(r_set.getInt("COD_MUNICIPIO"));
            municipio.setNombre_municipio(r_set.getString("NOMBRE"));

            listaMunicipio.add(municipio);
        }
        r_set.close();
        p_statement.close();


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

    return listaMunicipio;
```

Figura 48. Clase "CrudReporte", método readMunicipio()
Fuente: John Leandro Mejía Castro

Para lograr traer los datos de la base de datos debe realizarse una conexión del sistema con la base de datos MYSQL para lo cual se crea el paquete *Conexionbd*, también el sistema debe realizar backups de la información que se encuentra en esa base de datos por lo que se crea el paquete BDRespaldo.

La clase Conexion que se encuentra en el paquete *Conexionbd* contiene los datos y métodos necesarios para realizar la conexión del sistema con la base de datos MYSQL, tal como se presenta en la Figura 49.

```
public class Conexion {
    private  String servidor = "localhost";
    private  String bd = "sistema_experto_v_11";
    private  String usuario = "john";
    private  String password =  "john";
    private  String classname="com.mysql.jdbc.Driver";
    private  String url = "jdbc:mysql://" + servidor + "/" + bd;
    private  Connection conexion;


    /** Creación del constructor que permite realizar la conexion con la base
     * de datos
     *
     */

    public Conexion() {
        try{
            Class.forName(classname);
            conexion= DriverManager.getConnection(url, usuario, password);
        }catch(ClassNotFoundException e){
            System.out.println("Error"+e);
        }catch(SQLException e){
            System.out.println("Error"+e);
        }
    }


    /**
     *
     * @return conexion - Retorna la conexion con la base de datos
     */

    public Connection conectar(){
        return conexion;
```

Figura 49. Clase "Conexion"
Fuente: John Leandro Mejía Castro

La clase Prueba que se encuentra en el paquete *Conexionbd* contiene los datos y métodos necesarios para realizar la conexión del sistema con la base de datos MYSQL permitiendo ejecutarse y determinar si la conexión con la base de datos es exitosa o por el contrario no hay conexión con esta, tal como se presenta en la Figura 50.

```java
public static Connection getConexion() {

    String servidor = "localhost";
    String puerto = "3306";
    String bd = "sistema_experto_v_11";
    String usuario = "john";
    String password = "john";
    String classname = "com.mysql.jdbc.Driver";
    String url = "jdbc:mysql://" + servidor + "/" + bd;
    Connection conexion = null;

    try {
        Class.forName(classname);
        conexion = DriverManager.getConnection(url, usuario, password);
    } catch (ClassNotFoundException e) {
        System.out.println("Error" + e);
    } catch (SQLException e) {
        System.out.println("Error" + e);
    }

    return conexion;
}

/**
 * metodo que permite ejecutar la conexion con la base de datos
 * @param args
 */
public static void main(String[] args) {
    prueba.getConexion();
}
```

Figura 50. Clase "Prueba"
Fuente: John Leandro Mejía Castro

La clase Respaldobd que se encuentra en el paquete *BDRespaldo* contiene los datos y métodos necesarios para realizar un backup de grado total de la información contenida en la base de datos MYSQL, tal como se presenta en la Figura 51.

```java
public class Respaldobd {

    /**
     *metodo que permite ejecutar el respaldo de la base de datos del sistema.
     * @param args the command line arguments
     */
    public static void main(String[] args) throws IOException {
        Process p = Runtime.getRuntime().exec("mysqldump -u root sistema_experto_v_11");
        InputStream is = p.getInputStream();
        FileOutputStream fos = new FileOutputStream("sistema_experto_v_11.sql");
        byte[] buffer = new byte[10000];
        int leido = is.read(buffer);
        while (leido > 0) {
            fos.write(buffer, 0, leido);
            leido = is.read(buffer);
        }
        fos.close();
    }
```

Figura 51. Clase "Respaldobd"
Fuente: John Leandro Mejía Castro

4.4.3 Documentación del código – Capa Controlador

Para crear la capa del controlador en el proyecto *sepchaV1.0* se crea el paquete *Controller,* el cual gestiona las acciones que desea realizar el usuario en la interfaces con la capa de los datos es decir el Modelo, este paquete contiene dos Servlets; *ServletCargarDatosTabla* y *ServletClasificacion* tal como se observa en la Figura 52.

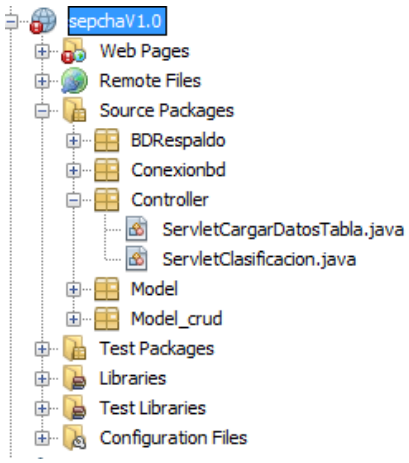

Figura 52. Estructura de capa controlador en proyecto SepchaV1.0
Fuente: John Leandro Mejía Castro

El Servlet *ServletCargarDatosTabla* que se encuentra en el paquete *Controller* hace uso de la lista de tipo Registro que se describió anteriormente, generando un archivo Json con esos datos y mediante un plugin de jQuery llamado DataTable poder cargar todos los registros y presentarlos de forma ordenada e intuitiva en una tabla Bootstrap en el archivo "datosalmacenados.jsp", tal como se presenta en la Figura 53.

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("application/json;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {

        CrudRegistro registro = new CrudRegistro();
        LinkedList<Registro> lista = registro.readRegistro();

        com.google.gson.JsonObject gson = new JsonObject();
        JsonArray array = new JsonArray();

        if(!lista.isEmpty()){
            String datos= "";
            int totallista= lista.size();
            int index=1;


            for (int i = 0; i < lista.size(); i++) {
               JsonObject item = new JsonObject();
               item.addProperty("Num",lista.get(i).getCod_registro());
               item.addProperty("Municipio",lista.get(i).getNombre_municipio());
               item.addProperty("Matepisovivienda",lista.get(i).getNombre_material_piso_vivienda());
               item.addProperty("Matedepared",lista.get(i).getNombre_material_pared());
               item.addProperty("Matetecho",lista.get(i).getNombre_material_techo());
               item.addProperty("Acabadopared",lista.get(i).getNombre_material_acabado_pared());
               item.addProperty("Servaguapotable",lista.get(i).getNombre_servicio_agua_publica());
               item.addProperty("Servdrenaje",lista.get(i).getNombre_drenaje());

               array.add(item);
            }

            gson.add("datos", array);

            out.println(gson.toString());
```

Figura 53. Servlet "ServletCargarDatosTabla"
Fuente: John Leandro Mejía Castro

El Servlet *ServletClasificacion* que se encuentra en el paquete *Controller* permite realizar la clasificación de los datos mediante el algoritmo ID3 por tanto debe leer los archivos tanto para entrenar este como para proceder a probar los nuevos registros y realizar la clasificación de estos generando así el reporte correspondiente sobre los posibles portadores de la enfermedad de Chagas, también se crean variables únicas para cada material a ser analizado por el algoritmo y así obtener las estadísticas respectivas para generar las gráficas por ende se hace uso de la clase Reporte la cual se explicó anteriormente, como también se pasan los datos de estas variables a la interfaz de “reportegenerado.jsp” para generar las tablas que acompañan a las gráficas en el reporte generado por el sistema experto.

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException, Exception {
    response.setContentType("text/html;charset=UTF-8");

    String zona = request.getParameter("zona").trim();
    String municipio = "";
    int numeroregistrosentrenamiento = 0;
    int contadorregistro = 0;
    int contadorregistropositivo = 0;
    String contadorregistroPositivo = "";
    int contadorregistronegativo = 0;
    String contadorresgistroNegativo = "";

    // variables de piso vivivienda
    int contadorregistro_pisoviv_cementoFirme = 0;
    String contadorregistro_Pisoviv_CementoFirme = "";
    int contadorregistro_pisoviv_maderaMosaico = 0;
    String contadorregistro_Pisoviv_MaderaMosaico = "";
    int contadorregistro_pisoviv_tierra = 0;
    String contadorregistro_Pisoviv_Tierra = "";

    // variables de material de pared;  madera-adobe,panel-concreto,tabique-ladrillo,carton-hule,lm-metalica-fbvid,Otro
    int contadoreregistro_matpared_maderaAdobe = 0;
    String contadorregistro_Matpared_MaderaAdobe = "";
    int contadorregistro_matpared_panelConcreto = 0;
    String contadorregistro_Matpared_PanelConcreto = "";
    int contadortabiqueLadrillo = 0;
    String contadorregistro_Matpared_TabiqueLadrillo = "";
    int contadoreregistro_matpared_cartonHule = 0;
    String contadorregistro_Matpared_CartonHule = "";
    int contadoreregistro_matpared_lmmetalicaFbvid = 0;
    String contadorregistro_Matpared_LmMetalicaFbvid = "";
    int contadoreregistro_matpared_otro = 0;
    String contadorregistro_Matpared_Otro = "";
```

Figura 54. Servlet "ServletClasificacion", variables para generar estadísticas de reporte
Fuente: John Leandro Mejía Castro

```
ConverterUtils.DataSource source = new ConverterUtils.DataSource("C:\\xampp\\htdocs\\sepchaV1.0\\web\\archivos\\seleccion_80_porc_entrenamiento.arff");
Instances inst = source.getDataSet();
inst.setClassIndex(7);
numeroregistrosentrenamiento = inst.numInstances();

Id3 alg = new Id3();
alg.buildClassifier(inst);

Instance anali = new Instance(7);
anali.setDataset(inst);

br = new BufferedReader(new FileReader("C:\\xampp\\htdocs\\sepchaV1.0\\web\\archivos\\seleccion_20_porc_pruebas.arff"));
String line = br.readLine();
while (null != line) {
    String[] fields = line.split(SEPARADOR);
    municipio = fields[0].trim();

    if (zona.equals(municipio)) {
        anali.setValue(0, fields[0]);
        anali.setValue(1, fields[1]);
        anali.setValue(2, fields[2]);
        anali.setValue(3, fields[3]);
        anali.setValue(4, fields[4]);
        anali.setValue(5, fields[5]);
        anali.setValue(6, fields[6]);

        double probabilidad[] = alg.distributionForInstance(anali);
        System.out.println(fields[0] + "," + fields[1] + "," + fields[2] + "," + fields[3] + "," + fields[4] + "," + fields[5] + "," + fields[6] + "-->probab

        String tabique = "tabique-ladrillo";
        System.out.println(fields[2].length() + " VS " + tabique.length());
```

Figura 55. Servlet "ServletClasificacion", entrenamiento y prueba de algoritmo ID3
Fuente: John Leandro Mejía Castro

```
// Variables para generar estadisticas

contadorregistro++;
if (probabilidad[0] == 1.0) {
    contadorregistropositivo++;

    // contadores del material del piso; cemento-firme,madera-mosaico,tierra
    if (fields[1].equals("cemento-firme")) {
        contadorregistro_pisoviv_cementoFirme++;
    }
    if (fields[1].equals("madera-mosaico")) {
        contadorregistro_pisoviv_maderaMosaico++;
    }
    if (fields[1].equals("tierra")) {
        contadorregistro_pisoviv_tierra++;
    }
```

Figura 56. Servlet "ServletClasificacion", estadísticas por material si la clasificación es positiva
Fuente: John Leandro Mejía Castro

```
// Generar grafica de portadores de chagas
String textosPortadoresChagas = "Positivo;Negativo";
contadorregistroPositivo = String.valueOf(contadorregistropositivo);
contadorresgistroNegativo = String.valueOf(contadorregistronegativo);
String valoresPortadoresChagas = contadorregistroPositivo + ";" + contadorresgistroNegativo;
String nombreArchivo = "portadoresdechagas";
System.out.println(valoresPortadoresChagas);
Reporte generarReporte = new Reporte(textosPortadoresChagas, valoresPortadoresChagas, nombreArchivo);

generarReporte.Reporte();
```

Figura 57. Servlet "ServletClasificacion", generar gráficas con datos obtenidos
Fuente: John Leandro Mejía Castro

```
RequestDispatcher rd = request.getRequestDispatcher("/vistas/marco.jsp?index=4&muestrap="+zona+
        "&numregistentre="+numeroregistrosentrenamiento+"&contregistro="+contadorregistro+
        "&registposit="+contadorregistropositivo+"&registnegat="+contadorregistronegativo+
        "&pisocemfirme="+contadorregistro_pisoviv_cementoFirme+"&pisomadmosaico="+contadorregistro_pisoviv_maderaMosaico+
        "&pisotierra="+contadorregistro_pisoviv_tierra+"&paredmadadobe="+contadoreregistro_matpared_maderaAdobe+
        "&paredpanconc="+contadorregistro_matpared_panelConcreto+"&paredtabladri="+contadortabiqueLadrillo+
        "&paredcartonhule="+contadoreregistro_matpared_cartonHule+"&paredlmmetfbvid="+contadoreregistro_matpared_lmmetalicaFbvid+
        "&paredotro="+contadoreregistro_matpared_otro+"&techocarthule="+contadoreregistro_mattecho_cartonHule+
        "&techotabladrillo="+contadoreregistro_mattecho_tabiqueLadrillo+"&techolmmetfbvid="+contadoreregistro_mattecho_lmmetalicaFbvid+
        "&techopancon="+contadorregistro_mattecho_panelConcreto+"&techomadadob="+contadoreregistro_mattecho_maderaAdobe+
        "&acabpareddninguno="+contadoreregistro_matacabpared_ninguno+"&acabparedrecpintapis="+contadoreregistro_matacabpared_recpinTapis+
        "&acabparedcem="+contadoreregistro_matacabpared_cemento+"&acabparedrecpastyessnpin="+contadoreregistro_matacabpared_RecPastYesSnPin+
        "&servaguano="+contadoreregistro_servaguapub_no+"&servaguasi="+contadoreregistro_servaguapub_si+
        "&servdrenletrina="+contadoreregistro_servdrenaje_letrina+"&servdrensanconconex="+contadoreregistro_servdrenaje_sanitaconconex+
        "&servdrensansinconex="+contadoreregistro_servdrenaje_sanitasinconex+"&servdrennoservsan="+contadoreregistro_servdrenaje_noservsanita);
rd.forward(request, response);
```

Figura 58. Servlet "ServletClasificacion", paso de datos estadísticos para tablas en archivo "reportegenerado.jsp"
Fuente: John Leandro Mejía Castro

Capítulo 5. PRUEBAS

El presente capítulo muestra las diversas pruebas efectuadas al Sistema Experto para determinar su correcto funcionamiento.

5.1. Ambiente de pruebas

En esta sección se presentan las características de los dispositivos electrónicos utilizados para realizar las pruebas al sistema.

HARDWARE:

- Laptop, RAM: 8GB, Procesador: AMD A6
- Número de usuarios: 1 usuarios.

SOFTWARE:

- Sistema operativo Windows 8.1
- Servidor Xampp versión 3.2.2
- Navegador de internet: Chrome, Mozilla.

5.2. Prueba de base de datos

En esta sección se presentan los registros almacenados en la base de datos MYSQL *sistema_experto_v_11,* en sus 10 tablas, presentando en phpMyAdmin como gestor los diferentes registros almacenados correctamente en estas tablas.

En la Figura 59 se presenta la prueba de correcto almacenamiento en la base de datos en la tabla “drenaje”.

COD_DRENAJE	NOMBRE_DRENAJE
1	sanita-con-conex
2	sanita-sin-conex
3	letrina
4	no-serv-sanita

Figura 59. Prueba de almacenamiento de datos a tabla "drenaje"
Fuente: John Leandro Mejía Castro

En la Figura 60 se muestra la prueba de correcto almacenamiento en la base de datos en la tabla “material_acabado_pared”.

COD_MATERIAL_ACABADO_PARED	NOMBRE_ACABADO_PARED
1	cemento
2	ninguno
3	rec-past-yes-snpin
4	rec-pin-tapiz

Figura 60. Prueba de almacenamiento de datos a tabla "material_acabado_pared"
Fuente: John Leandro Mejía Castro

En la Figura 61 se observa la prueba de correcto almacenamiento en la base de datos en la tabla “material_pared”.

COD_MATERIAL_PARED	NOMBRE_MATERIAL_PARED
1	carton-hule
2	madera-adobe
3	tabique-ladrillo
4	lm-metalica-fbvid
5	otro
6	panel-concreto

Figura 61. Prueba de almacenamiento de datos a tabla "material_pared"
Fuente: John Leandro Mejía Castro

En la Figura 62 se plasma la prueba de correcto almacenamiento en la base de datos en la tabla “material_piso_vivienda”.

COD_MATERIAL_PISO_VIVIENDA	NOMBRE_MATERIAL
1	cemento-firme
2	madera-mosaico
3	tierra

Figura 62. Prueba de almacenamiento de datos a tabla "material_piso_vivienda"
Fuente: John Leandro Mejía Castro

En la Figura 63 se muestra la prueba de correcto almacenamiento en la base de datos en la tabla “material_techo”.

COD_MATERIAL_TECHO	NOMBRE_MATERIAL_TECHO
1	carton-hule
2	lm-metalica-fbvid
3	madera-adobe
4	panel-concreto
5	tabique-ladrillo

Figura 63. Prueba de almacenamiento de datos a tabla "material_techo"
Fuente: John Leandro Mejía Castro

En la Figura 64 se presenta la prueba de correcto almacenamiento en la base de datos en la tabla “municipio”.

COD_MUNICIPIO	NOMBRE	MTS_NIVEL_MAR
1	AMACUZAC	982
2	ATLATLAHUCAN	1644
3	AXOCHIAPAN	1030
4	AYALA	1220
5	COATLAN-RIO	1010
6	CUAUTLA	1330
7	CUERNAVACA	1510
8	EMILIANO-ZAPATA	1250
9	HUITZILAC	2500
10	JANTETELCO	1160

Figura 64. Prueba de almacenamiento de datos a tabla "municipio"
Fuente: John Leandro Mejía Castro

En la Figura 65 se observa la prueba de correcto almacenamiento en la base de datos en la tabla “registro”.

COD_REGISTRO	COD_MUNICIPIO	COD_MATERIAL_PISO_VIVIENDA	COD_MATERIAL_ACABADO_PARED	COD_SERVICIO_AGUA_PUBLICA	COD_DRENAJE	COD_MATERIAL_PARED	COD_MATERIAL_TECHO
1	31	3	2	1	3	2	2
2	31	1	2	1	3	2	2
3	31	1	1	1	3	2	2
4	31	1	2	1	2	2	2
5	31	1	2	1	2	3	2
6	16	1	4	1	2	2	2
7	31	1	2	1	3	2	2
8	16	1	4	1	3	2	2
9	16	1	2	1	1	2	2
10	16	3	2	1	3	1	1
11	31	1	1	2	3	2	2
12	31	1	3	1	3	2	2
13	16	1	2	1	2	2	2
14	16	1	4	1	2	2	2
15	31	3	2	1	3	2	2
16	31	1	3	1	3	2	2

Figura 65.Prueba de almacenamiento de datos a tabla "registro"
Fuente: John Leandro Mejía Castro

En la Figura 66 se plasma la prueba de correcto almacenamiento en la base de datos en la tabla “registro_vector”.

COD_REGISTRO_VECTOR	COD_MUNICIPIO	COD_VECTOR
1	1	1
2	1	2
3	2	1
4	2	2
5	3	1
6	3	2
7	4	1
8	4	2
9	5	1

Figura 66. Prueba de almacenamiento de datos a tabla "registro_vector"
Fuente: John Leandro Mejía Castro

En la Figura 67 se ve la prueba de correcto almacenamiento en la base de datos en la tabla “servicio_agua_publica”.

COD_SERVICIO_AGUA_PUBLICA	ESTADO
1	No
2	Si

Figura 67. Prueba de almacenamiento de datos a tabla "servicio_agua_publica"
Fuente: John Leandro Mejía Castro

En la Figura 68 se observa la prueba de correcto almacenamiento en la base de datos en la tabla “vector”.

COD_VECTOR	NOMBRE_VECTOR
1	Triatoma barberi Usinger
2	Triatoma pallidipennis
3	Triatoma dimidiata Latreille

Figura 68. Prueba de almacenamiento de datos a tabla "vector"
Fuente: John Leandro Mejía Castro

5.3. Prueba de respaldo de base de datos

Tal como se muestra en la documentación del código la capa Modelo posee una clase que permite hacer un backup de la base de datos para el cual se necesita un esquema el cual se puede observar en la Figura 69.

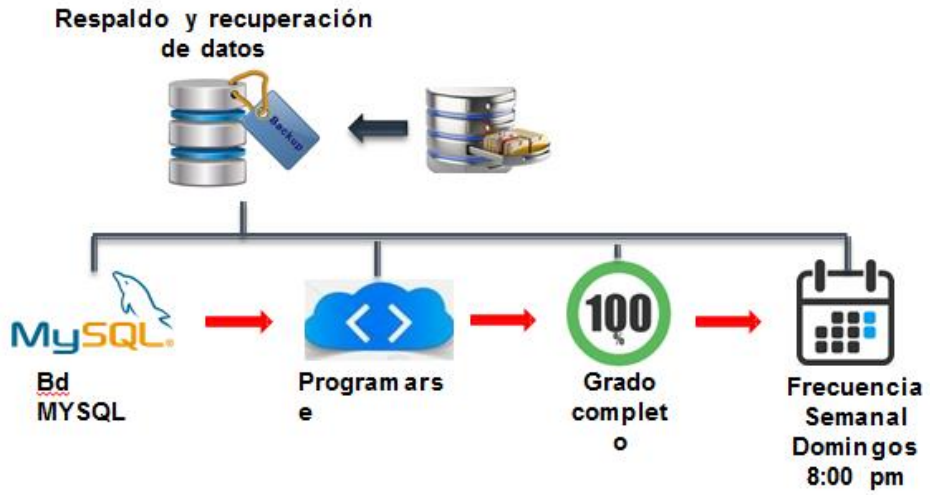

Figura 69. Prueba de respaldo de base de datos, esquema
Fuente: John Leandro Mejía Castro

La clase “Respaldobd” permite mediante codificación en la cual se debe tener un usuario con todos los permisos para la base de datos *sistema_experto_v_11* realizar un backup completo de la información de esta base de datos con una frecuencia semanal de los días domingos a las 8:00 pm, tal como se puede observar en la Figura 70.

```
-- Tiempo de generación: 18-11-2018 a las 08:09:57


SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";


--
-- Base de datos: `sistema_experto_v_11`
--

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `drenaje`
--

CREATE TABLE `drenaje` (
  `COD_DRENAJE` int(11) NOT NULL,
  `NOMBRE_DRENAJE` varchar(300) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Volcado de datos para la tabla `drenaje`
--

INSERT INTO `drenaje` (`COD_DRENAJE`, `NOMBRE_DRENAJE`) VALUES
```

Figura 70. Prueba de respaldo de base de datos, archivo .sql generado al ejecutar clase "Respaldobd"
Fuente: John Leandro Mejía Castro

5.4. Ejecución de pruebas de requisitos funcionales nominales (FN)

En la Tabla16 se presenta la documentación de la ejecución del requisito funcional número 1 en la cual se obtiene el resultado esperado de este requisito.

Tabla 16. Ejecución de prueba - FN1. Base de conocimientos

FN1. BASE DE CONOCIMIENTOS	
DATOS GENERALES DE LA PRUEBA	
Requisito a probar: FN1. Base de conocimientos	Ejecutor: John Leandro Mejía Castro
Fecha y Hora: 16/11/2018, 8:00 pm	Evaluador: Dra. Sandra Elizabeth León Sosa
DESARROLLO	
Objetivo: Comprobar que la clasificación del Sistema experto para posibles portadores de la enfermedad de Chagas sea la suministrada por la persona experta.	

Condiciones de ejecución: Se debe comparar la clasificación del Sistema experto versus la clasificación de la persona experta en el tema de Chagas, esto observando la matriz de confusión para obtener porcentajes de esto.		
Entradas: Archivo con el 80% de datos para entrenamiento de la red. Archivo con el 20% de datos para probar la red		
Acciones: 1. Ejecutar aplicación Java que genere por consola la clasificación del algoritmo ID3. 2. Recolectar clasificación obtenida por consola y comparar esta con la clasificación realizada por la persona experta. 3. Generar la matriz de confusión que permita determinar el grado de acierto en la clasificación de los registros por parte del Sistema Experto.	Resultados esperados: 1. En consola obtener cada uno de los registros analizados y el resultado de su clasificación. 2. En archivo .xls generar la matriz de confusión. 3. La matriz de confusión debe arrojar un 100% en los datos que el experto clasificó como positivos y que el sistema experto también califica como positivos, a su vez debe generar un 100% en los datos que el experto clasifica como negativos y que el sistema experto también debe clasificar como negativos.	Resultados obtenidos: Se obtuvo en la matriz de confusión un 100% de datos correctos clasificados como positivos por la persona experta y el sistema experto y también se obtuvo un 100% de datos clasificados correctamente como negativos por el experto y también por el sistema experto.
Evaluación de la prueba: La prueba se realizó de manera satisfactoria.		
Acciones correctivas: Ninguna.		

Fuente: John Leandro Mejía Castro

En la Figura 71 se presenta el resultado de la prueba presentada en la Tabla 16 (Ejecución de prueba – FN1. Base de conocimientos)

		Sistema Experto		
	n = 3288	Positivos	Negativos	
Realidad	Positivos	100%	0%	
	1143	1143	0	1143
	Negativos	0%	100%	
	2145	0	2145	2145
		1143	2145	

Figura 71. Resultado de prueba - FN1. Base de conocimientos
Fuente: John Leandro Mejía Castro

En la Figura 71, se puede observar la eficacia del algoritmo ID3 puesto que el número de registros determinado por el algoritmo como positivos es el mismo que el

determinado por la persona experta, 1143 registros, e igualmente el número de registros determinados como negativos por la persona experta es el mismo número de registros determinado por el algoritmo, 2145 registros para un total de 3.288 registros que son los registros totales de prueba, es importante notar que el sistema experto no clasifica ningún registro diferente a como lo determina la persona experta es decir hay un 100% de eficacia en la clasificación de los registros de prueba. De esta forma se determina que el requisito FN1 ha sido exitoso.

En la Tabla 17, se presenta la documentación de la ejecución del requisito funcional nominal número 2 en el cual se obtiene el resultado esperado de este requisito.

Tabla 17. Ejecución de prueba - FN2. Presentación de datos almacenados

FN2. PRESENTACIÓN DE DATOS ALMACENADOS		
DATOS GENERALES DE LA PRUEBA		
Requisito a probar: FN2. Presentación de datos almacenados	Ejecutor: John Leandro Mejía Castro	
Fecha y Hora: 20/11/2018, 5:00 pm	Evaluador: Dra. Sandra Elizabeth León Sosa	
DESARROLLO		
Objetivo: Probar la correcta presentación de los registros almacenados en la base de datos a ser clasificados por el algoritmo ID3.		
Condiciones de ejecución: Deben mostrarse los 3.288 registros de forma que sea intuitivo al usuario y no le genere molestia en el momento de encontrar un registro.		
Entradas: Consulta a la base de datos *sistema_experto_v_11* Tabla "registro"		
Acciones: 1. Ejecutar el sistema web en un navegador. 2. Ingresar a la interfaz Datos almacenados. 3. Presentar los 3.288 registros al usuario del sistema experto 4. El usuario puede introducir en un campo de búsqueda el registro a encontrar.	Resultados esperados: 1. Aparece pantalla de bienvenida al usuario del sistema experto. 2. El usuario debe hacer clic en el botón del menú vertical desplegable que tiene por nombre "**Datos almacenados**". 3. Mostrar tabla indexada permitiendo al usuario seleccionar cuantos registros desea observar. 4. Campo que permita buscar en los 3.288 registros el	Resultados obtenidos: El sistema experto muestra en una tabla la cual contiene un combo que permite seleccionar el número de registros a ver por vista y un campo de texto que muestra el registro filtrado por el usuario.

	especificado por el usuario.	
Evaluación de la prueba: La prueba se realizó de manera satisfactoria.		
Acciones correctivas: Ninguna.		

Fuente: John Leandro Mejía Castro

En la Figura 72 se presenta el resultado de la prueba presentada en la Tabla 17 (Ejecución de prueba – FN2. Presentación de datos almacenados)

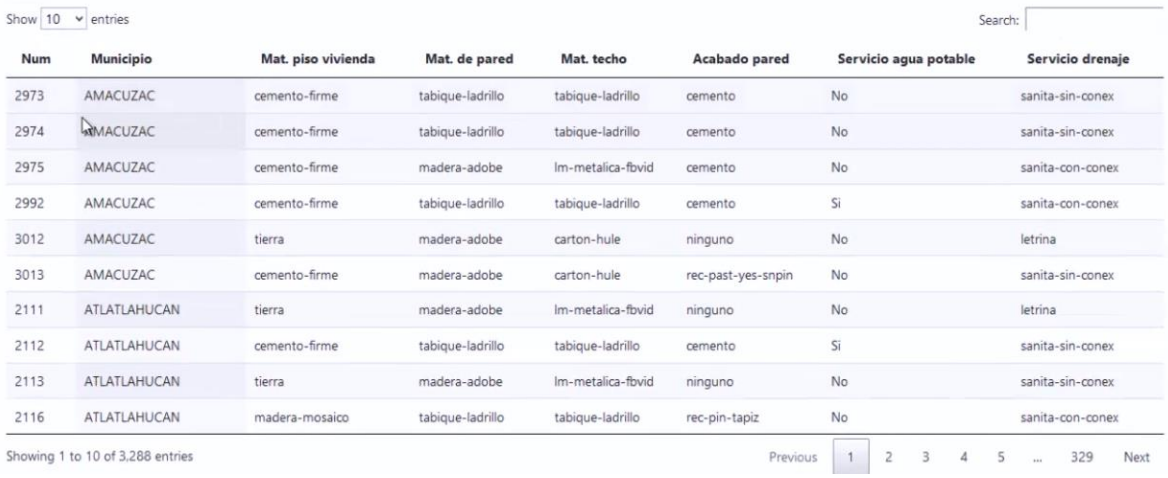

Figura 72. Resultado de prueba - FN2. Presentación de datos almacenados
Fuente: John Leandro Mejía Castro

En la Figura 72, se muestra cómo se desarrolla en su totalidad el requisito FN2, pues se presentan al usuario los registros que serán clasificados por el Sistema Experto.

En la *Tabla 18*, se muestra la documentación de la ejecución del requisito funcional nominal 3 del sistema experto, en la cual se obtiene el resultado esperado.

Tabla 18. Ejecución de prueba - FN3. Generación de reporte

FN3 GENERACIÓN DE REPORTE	
DATOS GENERALES DE LA PRUEBA	
Requisito a probar: FN3. Generación de reporte	Ejecutor: John Leandro Mejía Castro
Fecha y Hora:	Evaluador: Dra. Sandra Elizabeth León Sosa

20/11/2018, 03:15pm		
DESARROLLO		
Objetivo: Generar Gráficos y tablas de acuerdo a la clasificación realizada por el sistema experto.		
Condiciones de ejecución: El usuario debe seleccionar la zona o municipio a la cual se le realizará la clasificación.		
Entradas: Nombre de Zona o municipio a clasificarse.		
Acciones: 1. Ejecutar el sistema web en un navegador. 2. El usuario debe hacer clic en el botón del menú vertical desplegable que tiene por nombre "**Reporte**". 3. Seleccionar del combo el municipio del cual desea obtener el reporte**.** 4. Dar clic en botón "**Ver Reporte**".	Resultados esperados: 1. Aparece pantalla de bienvenida al usuario del sistema experto. 2. El Sistema debe presentar la interfaz Reporte. 3. En la interfaz Reporte el usuario puede hacer uso de un combo para seleccionar zona. 4. El sistema experto debe mostrar al usuario el reporte con las gráficas y las tablas necesarias de la clasificación realizada.	Resultados obtenidos: El sistema experto muestra al usuario el reporte con las gráficas y las tablas de los datos obtenidos de la clasificación realizada.
Evaluación de la prueba: La prueba se realizó de manera satisfactoria.		
Acciones correctivas: Ninguna.		

Fuente: John Leandro Mejía Castro

En la Figura 73, Figura 74 y Figura 75 se presentan los resultados de la prueba presentada en la Tabla 18 (Ejecución de prueba – FN3. Generación de reporte)

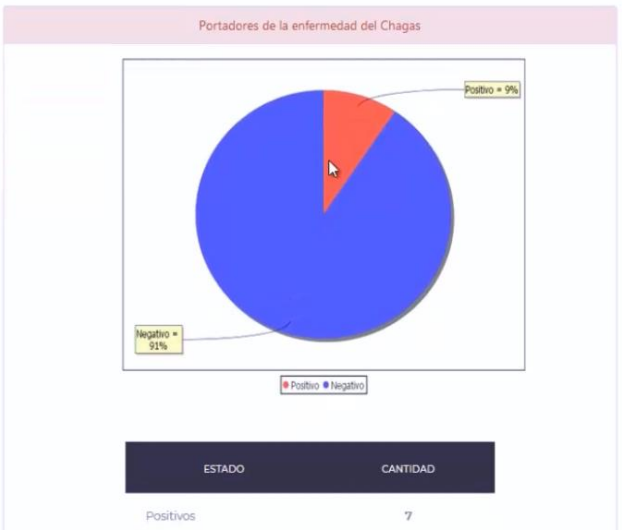

Figura 73. Resultado de prueba - FN2. Generación de reporte
Fuente: John Leandro Mejía Castro

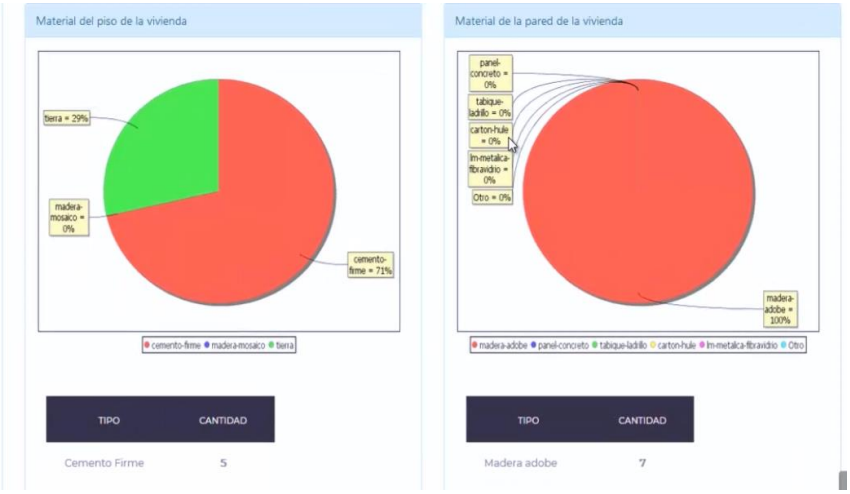

Figura 74. Resultado de prueba - FN2. Generación de reporte
Fuente: John Leandro Mejía Castro

Figura 75. Resultado de prueba - FN2. Generación de reporte
Fuente: John Leandro Mejía Castro

En la *Tabla 19*, se muestra la documentación de la ejecución del requisito funcional nominal 4 del sistema experto, en la cual se obtiene el resultado esperado.

Tabla 19. Ejecución de prueba - FN4. Comparación de algoritmos

FN4. COMPARACIÓN DE ALGORITMOS		
DATOS GENERALES DE LA PRUEBA		
Requisito a probar: FN3. Comparación de algoritmos	Ejecutor: John Leandro Mejía Castro	
Fecha y Hora: 7/11/2018, 03:00pm	Evaluador: Dra. Sandra Elizabeth León Sosa	
DESARROLLO		
Objetivo: Comprobar que el modelo generado por el algoritmo clasificador sea el preciso para lograr una correcta eficacia en el momento de que el sistema experto realice la clasificación de los registros.		
Condiciones de ejecución: Se deben comparar los modelos de los algoritmos J48 e ID3		
Entradas: Archivo con 80% de registros a clasificar.		
Acciones: 1. Generar un archivo .arff con el 80% de los registros a clasificar y un archivo .csv con estos mismos registros.	Resultados esperados: 1. Archivo .arff que contenga el 80% de registros a clasificar.	Resultados obtenidos: Se generan los árboles correspondientes a la clasificación de los registros por el algoritmo J48 e ID3, con lo cual se proceden a

2. Cargar en la plataforma Weka el archivo .arff con el 80% de los registros. 3. Clasificar con el algoritmo J48 el archivo .arff y generar el árbol de clasificación. 4. Cargar en la plataforma Rapid Miner el archivo .csv con el 80% de los registros. 5. Clasificar con el algoritmo ID3 el archivo .csv y generar el árbol de clasificación.	2. Visualizar en la plataforma las entidades y las diferentes gráficas que se generan del 80% de los registros. 3. Obtener el respectivo análisis del archivo .arff, como también el árbol que genera la clasificación con el algoritmo. 4. Visualizar en la plataforma las entidades y los diferentes registros que se generan del 80% de datos. 5. Obtener el respectivo análisis del archivo .csv como también el árbol que genera la clasificación con el	comparar, seleccionando así el algoritmo ID3 por su clasificación.
Evaluación de la prueba: La prueba se realizó de manera satisfactoria.		
Acciones correctivas: Ninguna.		

Fuente: John Leandro Mejía Castro

En la Figura 76 y Figura 77 se presentan los árboles obtenidos por la clasificación de los algoritmos J48 e ID3 de la ejecución de prueba presentada en la Tabla 19 (Ejecución de prueba – FN1. Generación de reporte)

Figura 76. . Resultado de prueba - FN4. Comparación de algoritmos - algoritmo ID3 - RapidMiner
Fuente: John Leandro Mejía Castro

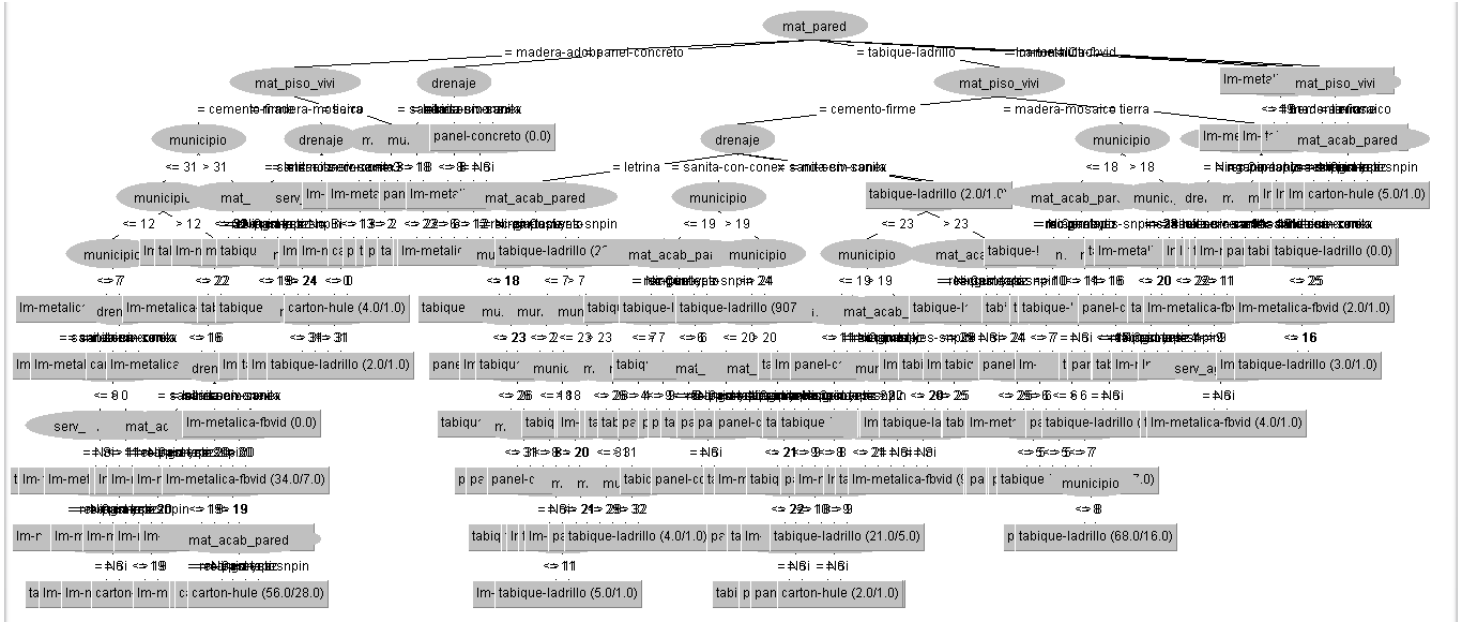

Figura 77. Resultado de prueba - FN4. Comparación de algoritmos - algoritmo J48 – Weka
Fuente: John Leandro Mejía Castro

Tabla 20. Ejecución de prueba - Comparación de algoritmos J48 - ID3

Algoritmo J48	Algoritmo ID3
Se obtiene un 88.57% de clasificación correcta de las instancias.	Se obtiene un 98.98% de clasificación correcta de las instancias.
El árbol generado por este algoritmo al hacer un análisis de su clasificación no cumple en un 100% el resultado generado por el humano experto en Chagas.	El árbol generado por este algoritmo al hacer un análisis de su clasificación sí cumple en un 100% el resultado generado por el humano experto en Chagas.

Fuente: John Leandro Mejía Castro

Capítulo 6. CONCLUSIONES

En el presente capítulo se plasman las conclusiones pertinentes al desarrollo del Sistema Experto, mostrando como se efectuaron en su totalidad los diversos requerimientos de éste.

6.1. Conclusiones

- Se generó la base de conocimientos adecuada para el sistema experto, para llevar a cabo esto fue fundamental realizar el proceso de KDD o extracción de conocimiento, convirtiéndose en la fase más compleja y larga del proyecto pues en cada sub fase de esta hubo que retornar a otras en más de 3 ocasiones, generando que tiempo y procesos realizados se perdieran pero generando al final lo más importante para poder desarrollar el proyecto; la base de conocimientos.

- El Sistema Experto permite al usuario del sistema experto ver la totalidad de los registros de prueba. El proceso de mostrar los datos almacenados necesitó de uso de plugins de JQuery conocido como "**DataTable**", también de JavaScript pues al ser tantos registros debe de ser indexada la tabla que se presenta al usuario, de tal forma que este pueda seleccionar el número de registros que desea ver por interfaz y a su vez le permita filtrar estos.

- Para que el Sistema Experto realice los reportes. Se necesitó hacer uso de una librería Java conocida como jfreechart, versión 1.0.19, por medio de la cual se realizan gráficos intuitivos los cuales permitan presentar el reporte pertinente al usuario de los posibles portadores de la enfermedad de Chagas, facilitando la implementación del módulo de reportes en el sistema experto.

- Al comparar los algoritmos clasificadores; J48 e ID3 se observó una mejor clasificación por parte del algoritmo ID3, motivo por el cual se seleccionó para hacer la clasificación de los registros de prueba, con lo cual se obtuvo una eficacia del cien por ciento como se puede observar en la matriz de confusión de los registros de prueba (Figura 28), cumpliendo de esta manera con la totalidad del requisito.

- Los requerimientos funcionales nominales como los funcionales no nominales se llevaron a cabo en su totalidad permitiendo así que se analice el problema de la enfermedad de Chagas desde una perspectiva social, pues tal como dijo la Doctora Any Laura Flores Villegas en el *Simposio Latinoamericano de Cardiopatía Chagásica* "La mejor forma de hacer un control vectorial es la prevención", y que mejor manera que determinando de acuerdo al estado actual de la vivienda si las personas pueden ser portadores de esta enfermedad también conocida como la enfermedad de la pobreza.

- Dentro del sistema se genera un reporte por municipio donde visualiza los datos analizados dependiendo del municipio a seleccionar, detallado por el material de vivienda, indicando además el algoritmo que se utilizó con el número de registros que son procesados ver Anexo II .

6.2. Trabajo futuro

El Sistema Experto para posibles portadores de la enfermedad de Chagas (SEPPCHA) fue desarrollado de tal manera que permita ser escalable, por tal motivo deben crearse formularios los cuales permitan ingresar información pertinente a la base de datos generada la cual también se creó teniendo en cuanta factores que se están estudiando y de los cuales se esperan obtener los estudios pertinentes para influir en la clasificación de esta enfermedad. Tales factores deberán tenerse en cuenta en un futuro para que el sistema realice la clasificación correcta.

Debido a que todos los días se hacen estudios y se conocen nuevas cosas sobre la enfermedad de Chagas el Sistema Experto puede seguir complementándose con cada uno de estos estudios, cumpliendo así con el objeto de analizar el problema desde una perspectiva social.

Referencias Bibliográficas

- Alvarado, D. M. (2018). Simposium Latinoamericano de Cardiopatía Chagásica. *Diagnóstico clínico y serológico: Fases Aguda y Crónica de la enfermedad de Chagas.* Cuernavaca, Morelos.
- Flores, D. J. (2018). Simposium Latinoamericano de Cardiopatía Chagásica. *Enfoque clínico actual del paciente con cardiopatía Chagásica Crónica.* Cuernavaca, Morelos.
- Gutiérrez, D. E. (2018). Simposium Latinoamericano de Cardiopatía Chagásica. *Ciclo biológico del Trypsnosoma cruzi, agente causal de la enfermedad de Chagas.* Cuernavaca, Morelos.
- M.Sc. Ingrid Wilford Rivera, D. A. (s.f.). *Revista Cubana de Informática Médica.* Recuperado el 13 de 09 de 2018, de Revista Cubana de Informática Médica: http://www.rcim.sld.cu/revista_18/articulos_htm/mineriadatos.htm#Conclu
- MADRID, U. A. (24 de 07 de 2014). *YouTube*. Recuperado el 26 de 10 de 2018, de YouTube: https://www.youtube.com/watch?v=4VvlgkL25Ks
- Perucho, D. E. (2018). Simposium Latinoamericano de Cardiopatía Chagásica. *Morbi-mortalidad de la cardiopatía Chagásicaen latinoamérica.* Cuernavaca, Morelos.
- Ramos, D. C. (2018). Simposium Latinoamericano de Cardiopatía Chagásica. *Enfermedad de Chagas en México: compromisos pendientes.* Cuernavaca, Morelos.
- Rivas, D. R. (2018). Simposium Latinoamericano de Cardiopatía Chagásica. *Tamizajeserológico de la enfermesas de Chagas en donadores de sangre.* Cuernavaca, Morelos.
- Schettino, D. P. (2018). Simposium Latinoamericano de Cardiopatía Chagásica. *Miocardiopatía Chagásica infantil en México.* Cuernavaca, Morelos.
- Villegas, D. A. (2018). Simposium Latinoamericano de Cardiopatía Chagásica. *Alternativas en el cotrol de triatominos.* Cuernavaca, Morelos.
- bosspetta, a. J.-c.-d. (13 de 09 de 2018). *MDN WEB DOCS*. Recuperado el 21 de 11 de 2018, de MDN WEB DOCS:

https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/Qu%C3%A9_es_JavaScript

- CACOO. (2018). *CACOO*. Recuperado el 21 de 11 de 2018, de CACOO: https://support.cacoo.com/hc/es-419

- Flores, D. J. (2018). Simposium Latinoamericano de Cardiopatía Chagásica. *Enfoque clínico actual del paciente con cardiopatía Chagásica Crónica.* Cuernavaca, Morelos.

- Gutiérrez, D. E. (2018). Simposium Latinoamericano de Cardiopatía Chagásica. *Ciclo biológico del Trypsnosoma cruzi, agente causal de la enfermedad de Chagas.* Cuernavaca, Morelos.

- IBM. (2011). *IBM KNOWLEDGE CENTER*. Recuperado el 21 de 11 de 2018, de IBM KNOWLEDGE CENTER: https://www.ibm.com/support/knowledgecenter/es/SSQP76_7.5.0/com.ibm.dserver.res/Content/Business_Rules/_pubskel/Infocenter_Primary/ps_DS_Rule_Execution_Server1444.html

- Jorge. (12 de 07 de 2017). *CODIGO BINARIO*. Recuperado el 21 de 11 de 2018, de CODIGO BINARIO: http://www.codigo-binario.es/xampp-herramienta-para-dev-web/

- M.Sc. Ingrid Wilford Rivera, D. A. (s.f.). *Revista Cubana de Informática Médica.* Recuperado el 13 de 09 de 2018, de Revista Cubana de Informática Médica: http://www.rcim.sld.cu/revista_18/articulos_htm/mineriadatos.htm#Conclu

- MADRID, U. A. (24 de 07 de 2014). *YouTube*. Recuperado el 26 de 10 de 2018, de YouTube: https://www.youtube.com/watch?v=4VvlgkL25Ks

- Maria. (03 de 08 de 2016). *PUNTO ABIERTO*. Recuperado el 21 de 11 de 2018, de PUNTO ABIERTO: https://puntoabierto.net/blog/que-es-bootstrap-y-cuales-son-sus-ventajas

- marketingscientec. (17 de 10 de 2016). *scientec*. Recuperado el 21 de 11 de 2018, de scientec: https://www.scientec.com.mx/axure-rp/

- NETBEANS. (2018). *NETBEANS*. Recuperado el 21 de 11 de 2018, de NETBEANS: https://netbeans.org/index_es.html

- Perucho, D. E. (2018). Simposium Latinoamericano de Cardiopatía Chagásica. *Morbi-mortalidad de la cardiopatía Chagásicaen latinoamérica.* Cuernavaca, Morelos.

- POWERDESIGNER. (2015). *POWERDESIGNER*. Recuperado el 21 de 11 de 2018, de POWERDESIGNER: https://www.powerdesigner.biz/ES/powerdesigner/powerdesigner-features.html

- Ramos, D. C. (2018). Simposium Latinoamericano de Cardiopatía Chagásica. *Enfermedad de Chagas en México: compromisos pendientes.* Cuernavaca, Morelos.
- Rivas, D. R. (2018). Simposium Latinoamericano de Cardiopatía Chagásica. *Tamizajeserológico de la enfermesas de Chagas en donadores de sangre.* Cuernavaca, Morelos.
- Rodríguez, E. (31 de 08 de 2006). *MAESTROS DEL WEB*. Recuperado el 21 de 11 de 2018, de MAESTROS DEL WEB: http://www.maestrosdelweb.com/htmlhis/

Anexo I.

En este apartado se especifican las preguntas que se realizaron para dar inicio con el proyecto.

Preguntas de la entrevista.

1.- ¿Cuál es el objetivo primordial del proyecto a realizar?

2.- ¿Conocer los factores que determinan la enfermedad de Chagas?

3.- ¿Cómo se van obtener los datos para poder clasificarlos?

4.- ¿Qué tipo de algoritmos serán utilizados para el análisis de los datos?

5.- ¿Herramientas que serán utilizadas para procesar la información clasificada?

6.- ¿El tiempo del desarrollo del proyecto?

7.- ¿El lenguaje para desarrollar el sistema?

Anexo II.

En la siguiente Figura 78, se muestra la selección del municipio que se requiere analizar, son municipios que fueron obtenidos de la base de datos del censo del 19 de septiembre del 2017.

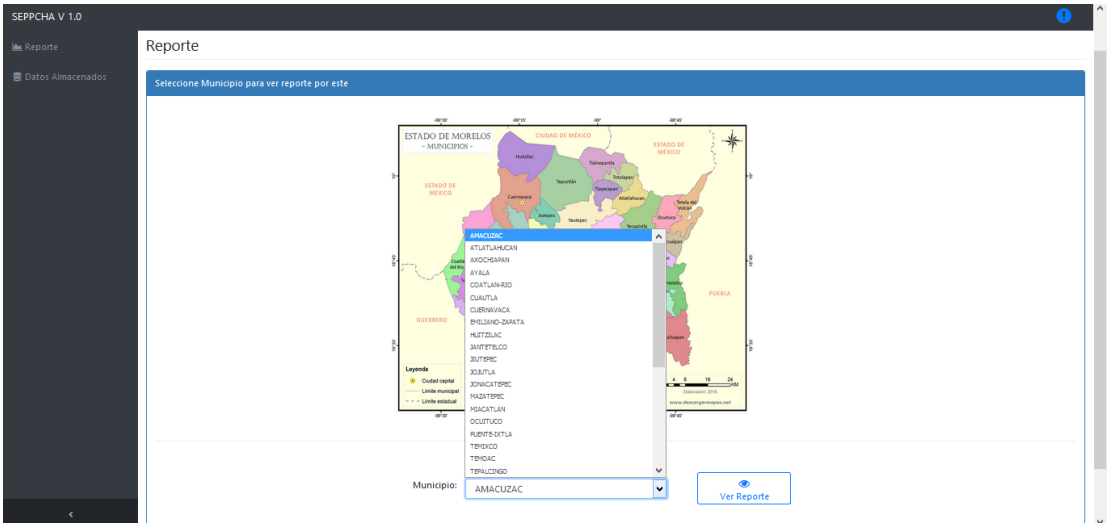

Figura 78. Anexo II - Selección del municipio que se debe analizar
Fuente: John Leandro Mejía Castro

En la siguiente Figura 79, indica el reporte con el número de registros con los que cuenta el municipio y el algoritmo con el que se ejecuta.

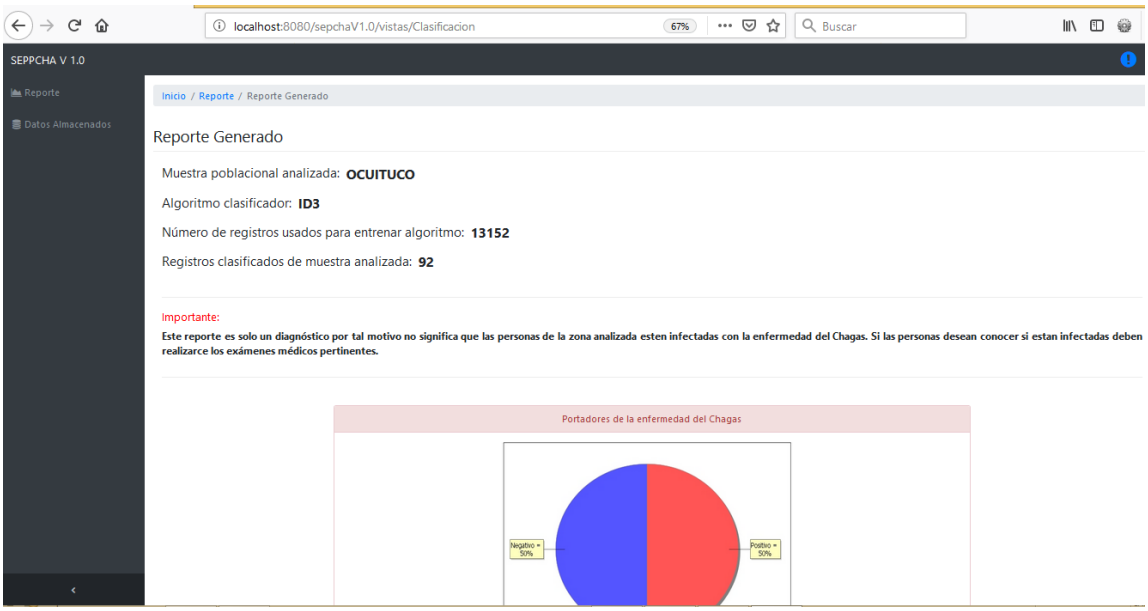

Figura 79. Anexo II - Detalle de reporte (número de registros del municipio, algoritmo que se ejecuta)
Fuente: John Leandro Mejía Castro