



Universidad de las Ciencias Informáticas

Facultad 2

Trabajo de Diploma para optar por el Título de
Ingeniero en Ciencias Informáticas

**PlaguiTox: Software para la gestión de información de los casos
de intoxicaciones por plaguicidas en el Cenatox**

Autores:

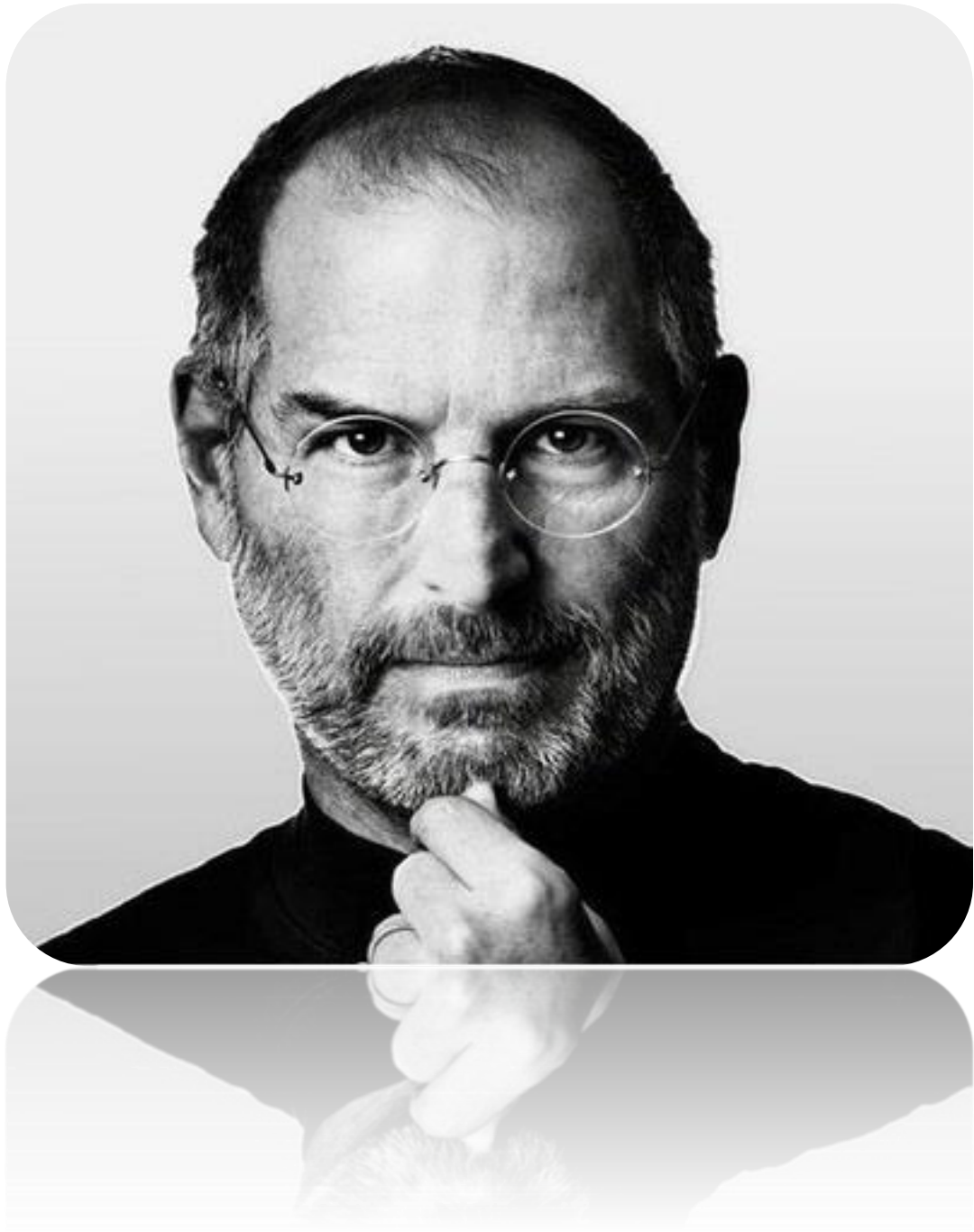
Víctor Alexis Mesa Cisneros

Manuel Alejandro Miranda Hernández

Tutor:

DrC. Arturo Orellana García

La Habana, junio de 2018



” El único modo de hacer un gran trabajo es amar lo que haces”

Steve Jobs



Declaración de autoría

Declaramos ser autores de la presente tesis que tiene por título: PlaguiTox: Software para la gestión de información de los casos de intoxicaciones por plaguicidas en el Cenatox; y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente a los ____ días del mes de junio del año 2018.

Manuel Alejandro Miranda Hernández

Autor

Víctor Alexis Mesa Cisneros

Autor

Dr. C. Arturo Orellana García

Tutor



Datos de contacto

Datos del autor:

Manuel Alejandro Miranda Hernández.

Universidad de las Ciencias Informáticas, La Habana, Cuba.

Email: mamiranda@estudiantes.uci.cu

Datos del autor:

Víctor Alexis Mesas Cisneros.

Universidad de las Ciencias Informáticas, La Habana, Cuba.

Email: vamesa@estudiantes.uci.cu

Datos del tutor:

Dr. C. Arturo Orellana García (aorellana@uci.cu): se desempeña como líder del Grupo de Investigación de Minería de procesos y Asesor de Capacitación, Desarrollo e Investigación del Centro de Soluciones de Informática Médica. Ha liderado proyectos I+D+i de desarrollo de componentes de software a partir de minería de procesos para el análisis de procesos de negocio del entorno hospitalario. Investiga la Ingeniería de comportamiento, la medicina de precisión y el procesamiento de imágenes médicas. Tutora varias tesis de grado, maestrías y doctorados enfocados al análisis de procesos de negocio, la informática médica y otras áreas del conocimiento. Obtuvo el grado de Máster en Informática Aplicada en 2015 desarrollando una herramienta informática basada en técnicas de minería de procesos para identificar problemas en la ejecución de procesos de negocio. Doctor en Ciencias Técnicas desde 2016 presentando un modelo computacional para la detección de variabilidad en procesos de negocio del entorno sanitario aplicando minería de procesos.



Dedicatoria

De Manuel Alejandro Miranda Hernández:

Este trabajo y todos sus resultados se los dedico a mis padres, de los que me siento orgulloso, que fueron la mejor escuela, los mejores amigos. Gracias por el amor, la entrega, la paciencia y la confianza que siempre me han dedicado. Nunca podré llegar a agradecerles lo suficiente por todos los sacrificios. Gracias por creer en mí, por soñar conmigo esta carrera. Si hoy estoy aquí es por ustedes. Los amo.

De Víctor Alexis Mesas Cisneros:

Le dedico esta tesis a mi familia:

A mi esposa y mi niña, mi mamá, mi papá y mi hermano por estar ahí en todo momento, brindándome su apoyo incondicional, por todos los sacrificios que han hecho para que yo pudiera estar hoy aquí, por todas las cosas que me han enseñado y por haberme ayudado a convertirme en la persona que hoy soy. En general a todas aquellas personas que de una forma u otra han contribuido con mi formación, este resultado también es de ustedes.

Alex



Agradecimientos

De Manuel Alejandro Miranda Hernández:

Quiero agradecer a todas aquellas personas que han hecho posible que este sueño se realizara.

A mis padres Odalys y Paco, a mis abuelos, tíos, primos y en general a toda la familia por tener su apoyo incondicional aun cuando las cosas no marchaban bien, porque cada uno de ellos siempre creyó en mí.

A mi novia por apoyarme en cada momento que lo necesite, por su amor, por su apoyo incondicional, por su atención, por alegrar cada uno de mis días, por poder contar con su presencia. Te amo.

Agradecerle al nuevo integrante de mi familia, mi compañero de tesis, por ser un excelente hermano, amigo y compañero. Por estar ahí cuando fue necesario, en las buenas y malas.

A mis compañeros de aula, por darme la oportunidad de vivir junto a ellos la más hermosa experiencia, la universidad, por solo estar ahí, en las fiestas, maldades, en los estudios en general por porque todos formaron una pequeña parte de mi camino.

A mis compañeros de apartamento, mis buenos amigos, por compartir fiestas cumpleaños, disgustos, alegría y tristeza, por su preocupación por hacer sentirme a gusto en la universidad.

A mi tutor, Arturo por toda la ayuda brindada, por estar siempre pendiente de que todo estuviera en tiempo, por soportarnos durante este tiempo y por su preocupación.

A el tribunal, por toda su ayuda, por los consejos y críticas constructivas que hicieron posible el desarrollo de este trabajo, por ayudarme a aprender.

A todos los profesores, que de una forma u otra durante los cinco años de la carrera contribuyeron en mi formación como profesional.

Especialmente agradecer a Mildrey, pues de no ser por ella, en un momento de mi estancia en la universidad lo hubiese dejado todo. Por brindarme su apoyo y amistad desinteresada.

A todos los que se me quedan y que saben que estuvieron ahí, gracias.



De Víctor Alexis Mesas Cisneros:

Quiero agradecer a todas las personas que han hecho posible este sueño.

- **En primer lugar a mi familia, en especial: mi niña y mi esposa, mi mamá, mi papá y mi hermano**, no tengo palabras para poder expresar todo lo agradecido que estoy de ustedes.
- **A mis abuelas, mis tías, mis primos, a papito, al babel** por siempre estar ahí y apoyarme a cumplir con cada una de mis metas.
- **A mis amigos y hermanos** que he hecho en estos 5 años de convivir con ustedes, no importa el lugar en que estemos el día de mañana, siempre los recordaré.
- **A los profe** que de una forma u otra han contribuido con este resultado.
- **A mi tutor** por haber confiado en mí y por todo lo que aprendí de usted.
- **A mi compañero de tesis** por la gran amistad que me has brindado, eres como un hermano.
- **A todos** los que se me quedan y que saben que estuvieron ahí, gracias.

Alex.

Resumen

El Centro Nacional de Toxicología (Cenatox) supervisa y controla la información de eventos atribuibles a la inmunización, vacunación e intoxicaciones con medicamentos y plaguicidas. Los casos que llegan al Centro se justifican mayormente por el uso de plaguicidas que tienen un alto nivel de toxicidad y riesgo de muerte. Es por ello que los especialistas en toxicología requieren tener facilidad de revisión de las hojas de seguridad, el listado oficial de plaguicidas autorizados en Cuba y los casos anteriores que, a partir de la experiencia, permitan analizar y emitir un diagnóstico que salve la vida del afectado.

En este trabajo de diploma se desarrolla un sistema que permite el diagnóstico de un paciente a partir de determinar el plaguicida que lo intoxicó. El sistema se soporta sobre el análisis de información de los casos de intoxicaciones y plaguicidas registrados en Cuba, utilizando razonamiento basado en casos.

El desarrollo se guio por la metodología de software Extreme Programming, modelando sobre la herramienta CASE Visual Paradgm 8.0 y lenguaje UML 2.0. Para la programación se utilizó Java con NetBeans 8.0.2 y como gestor de base de datos PostgreSQL 9.3.

Como resultados se obtuvo un marco teórico relacionado con la toxicología en Cuba y los sistemas que manejan información relacionada con plaguicidas y diagnósticos. Se desarrolló una base de casos de los síntomas, plaguicida y diagnóstico por plaguicida. Los especialistas de Cenatox cuentan actualmente con una herramienta de apoyo a la toma de decisiones reduciendo, en gran medida, la ocurrencia de errores humanos.

Palabras claves: gestión de casos, plaguicida, razonamiento basado en casos, sistema de gestión, síntomas toxicológicos.

Abstract

The National Toxicology Center (Cenatox) monitors and controls information on events attributable to immunization, vaccination, and drug and pesticide poisoning. The cases that arrive at the Center are mostly justified by the use of pesticides that have a high level of toxicity and risk of death. For this reason, toxicology specialists need to be able to easily review the safety data sheets, the official list of pesticides authorized in Cuba and the previous cases that, based on experience, allow for the analysis and diagnosis of the affected person's life.

In this diploma work a system is developed that allows the diagnosis of a patient from determining the pesticide that intoxicated him. The system is based on the analysis of information on cases of poisoning and pesticides registered in Cuba, using case based reasoning.

The development was guided by the Extreme Programming software methodology, modeling on the CASE Visual Paradgm 8.0 tool and UML 2.0 language. Java was used for programming with NetBeans 8.0.2 and as PostgreSQL 9.3 database manager.

The results were a theoretical framework related to toxicology in Cuba and the systems that manage information related to pesticides and diagnostics. A case base of symptoms, pesticide and pesticide diagnosis was developed. Cenatox specialists now have a decision support tool that greatly reduces the occurrence of human errors.

Keywords: case management, case based reasoning, management system, pesticide toxicological symptoms.

Índice

Introducción	1
Capítulo 1. Fundamentación teórica referente a la toxicología derivada al uso de plaguicidas	4
1.1. La toxicología en Cuba	4
1.2. Análisis del estado del arte	7
1.3. Sistemas basados en conocimientos.....	13
1.3.1 Razonamiento basado en casos.....	17
1.3.2 Técnicas de adquisición del conocimiento	18
1.4. Metodología de desarrollo	18
1.4.1. Metodología Programación Extrema (XP).....	19
1.5. Tecnologías y herramientas.....	22
1.5.1. Lenguajes de programación.....	22
1.5.2. Herramienta CASE: Visual Paradigm 8.0.....	23
1.5.3. Sistema gestor de base de datos.....	23
Conclusiones del capítulo.....	24
Capítulo 2. Características del sistema de gestión y análisis de casos de intoxicación por plaguicida	25
2.1. Modelo de negocio	25
2.2 Análisis de la solución.....	26
Funcionalidades del software PlaguiTox.....	26
Requisitos no funcionales del software PlaguiTox	26
2.3 Planeación usando historias de usuario.....	27
2.4 Patrones de diseño.....	32
2.5 Diseño del software PlaguiTox.....	35
2.6 Diseño de la base de casos	40
Conclusiones del capítulo.....	42
Capítulo 3. Implementación y validación del sistema	44
3.1 Proceso de extracción del conocimiento	44
Etapa de recuperación.....	44

Etapa de reutilización.....	45
Etapa de revisión o adaptación	45
Almacenamiento o aprendizaje	45
3.2 Fase de codificación	46
3.3 Pruebas de software	48
3.3.1 Niveles de prueba.....	48
Conclusiones del capítulo.....	61
Conclusiones	62
Recomendaciones	63
Referencias Bibliográficas.....	64
Glosario de términos.....	67
Bibliografía.....	68
Anexos.....	70

Índice de figuras

Figura 1: Interfaces de la Herramienta DiagnosMD. Fuente: (DiagnosMD, 2017)	9
Figura 2: Interfaces del software que muestra un listado de los posibles plaguicidas dado un caso. Fuente: (Ortiz Gálvez, 2012).	10
Figura 3: Interfaces del software que muestra los plaguicidas sugeridos según cultivo. Fuente: (Ortiz Gálvez, 2012).	11
Figura 4: Interfaces Grafica del software Rectox. Fuente: (Rectox, 2017).....	12
Figura 5: Ciclo del razonamiento basado en caso. Fuente: (Martí Pérez, 2012)	15
Figura 6: Etapas de la adquisición de conocimiento. Fuente: (Martí Pérez, 2012).	17
Figura 7: Ilustra el proceso XP. Fuente: (Pressman, 2007)	21
Figura 8: Ejemplo de patrón controlador en la implementación de la herramienta. Fuente: elaboración propia.....	33
Figura 9: Ejemplo de patrón creador en la implementación de la herramienta. Fuente: elaboración propia.	34
Figura 10: Ejemplo de patrón alta cohesión en la implementación de la herramienta. Fuente: elaboración propia.....	34
Figura 11: Ejemplo de patrón experto en la implementación de la herramienta. Fuente: elaboración propia.....	35
Figura 12: Arquitectura MVC. Fuente: elaboración propia.....	36
Figura 13: Patrón arquitectónico aplicado a la investigación. Fuente: elaboración propia.	37
Figura 14: Modelo de datos relacional para el software PlaguiTox. Fuente: elaboración propia.....	38
Figura 15: Caso de prueba del método GetId_Evolucion de la clase Caso. Fuente: elaboración propia.	56
Figura 16: Caso de prueba del método GetId_Paciente de la clase Paciente. Fuente: elaboración propia.	56
Figura 17: Caso de prueba del método GetId_Usuario de la clase Usuario. Fuente: elaboración propia.	57
Figura 18: Aval del resultado científico técnico. Fuente: Centro de informática médica.....	74
Figura 19: Solicitud de cooperación. Fuente: Cenatox.	75

Índice de tablas

Tabla 1: Clasificación de plaguicidas según riesgo recomendada por la OMS.....	7
Tabla 2: Comparación de las herramientas y software estudiados.....	12
Tabla 3: Personas que intervienen en el negocio.....	25
Tabla 4: Descripción de la HU 2 Gestión de Plaguicidas.....	28
Tabla 5: Descripción de la HU 3 Gestión de Casos.....	29
Tabla 6: Estimación del esfuerzo por HU.	30

Tabla 7: Plan de Iteraciones.	31
Tabla 8: Plan de Entrega.	31
Tabla 9: Tarjeta CRC clase Datos.....	39
Tabla 10: Tarjeta CRC clase PrincipalController.	39
Tabla 11: Tarjeta CRC clase Recuperar_Casos.	40
Tabla 12: Rasgos predictores de la base de casos del SBC.	40
Tabla 13: Rasgos objetivo de la base de casos del SBC.	42
Tabla 14: Tarea de ingeniería Eliminar usuario.....	46
Tabla 15: Tarea de ingeniería Listar plaguicida.....	47
Tabla 16: Tarea de ingeniería Insertar caso.....	47
Tabla 17: Caso de prueba autenticar usuario.....	49
Tabla 18: Caso de prueba insertar plaguicida.	50
Tabla 19: Cuadro Lógico de ladov modificado por el autor.	59
Tabla 20: Encuesta satisfacción de usuarios potenciales.....	70
Tabla 21: Descripción de la HU 1 Gestión de Usuarios.....	71
Tabla 22:Descripción de la HU 4 Consultar Hoja de Seguridad.	71
Tabla 23: Descripción de la HU 5 Exportar a PDF.	72
Tabla 24: Tarjeta CRC clase Gestion_plaguicidaContraller.....	73
Tabla 25: Tarjeta CRC clase Gestion_usuarioContraller.	73

Introducción

Según la Organización Mundial de la Salud (OMS) anualmente se registran entre uno y cinco millones de casos de intoxicación por plaguicidas, con varios miles de muertes. El 99% de estos hechos ocurren en países en desarrollo, entre los cuales los países de América Latina aportan el 75% de los casos y se estima que más de 700 000 personas al año sufren los efectos crónicos. Las intoxicaciones agudas por plaguicidas ocupan un lugar importante, precedidas solo por las reportadas por alimentos y medicamentos (Organización Panamericana de la Salud, 2012).

Según el Código Internacional de Conducta Sobre la Distribución y Uso de Plaguicidas de la Organización mundial para la agricultura y la alimentación (FAO) de las Naciones Unidas, establece que un plaguicida «es la sustancia o mezcla de ellas, destinada a prevenir, destruir o controlar plagas, incluyéndolos vectores de enfermedad humana o animal; las especies no deseadas de plantas o animales que ocasionan un daño duradero u otras que interfieren con la producción, procesamiento, almacenamiento, transporte y comercialización de alimentos; los artículos agrícolas de consumo, la madera y sus productos, el forraje para animales o los productos que pueden administrárseles para el control de insectos, arácnidos u otras plagas corporales» (RAMÍREZ, et al., 2001).

La mayoría de estas sustancias son fabricadas por el hombre, por lo que son llamados plaguicidas sintéticos. Los agricultores y otros usuarios de plaguicidas corren un riesgo elevado de exposición, sobre todo si no adoptan medidas de protección. El riesgo puede afectar también a sus familias y comunidades, cualquier persona puede exponerse a la presencia de plaguicidas en sus alimentos o entorno.

Cada día entran al mercado nuevos plaguicidas y los profesionales de la salud, generalmente, desconocen sus peligros y mecanismos tóxicos. Además, es fundamental tomar en cuenta los episodios graves de efectos adversos sobre la salud que han sido descritos tras la intoxicación por estos agentes.

Actualmente los países desarrollados tienden a evitar el uso de las sustancias persistentes, de elevada toxicidad y sustituirlas por compuesto de menor estabilidad química, menor toxicidad o que imiten a estructuras químicas naturales. En cambio, en los países subdesarrollados están sujetos a una serie de presiones técnicas, económicas y políticas que los inducen a utilizar grandes cantidades de plaguicidas más baratos y efectivos a corto plazo sin tener en cuenta los efectos negativos sobre la salud y el medio ambiente.

Cuba es un país subdesarrollado con una economía agrícola bien establecida, donde el uso y abuso de plaguicidas es amplio, resultando las intoxicaciones accidentales y los daños ocasionados intencionalmente por una inadecuada aplicación de estos potentes compuestos. Esto hace que las intoxicaciones por plaguicidas ocupen un lugar importante, precedida solo por los medicamentos.

El Centro Nacional de Toxicología (Cenatox), creado en 1986 en La Habana, es el coordinador del sistema de tóxico-vigilancia en Cuba. Posee una red integrada por tres centros regionales localizados

al oriente y centro de la isla. El Cenatox es la instancia de supervisión y control donde se gestiona la información de eventos atribuibles a la inmunización, vacunación e intoxicaciones con medicamentos y plaguicidas.

La visión del centro es lograr el fortalecimiento en la toma de decisiones cuando el personal médico, paramédico y o la población se enfrentan a productos químicos de cualquier índole y a su vez retroalimentarnos, desde la base, en relación al comportamiento de los eventos toxicológicos, todo lo anterior con el objetivo de contribuir al mejoramiento de la calidad de vida de la población (CENATOX, 2017).

Actualmente Cenatox no cuenta con un medio informatizado para el manejo de los datos, por lo que se pueden cometer errores en la gestión de estos por parte del personal médico, pues el proceso de diagnosticar a un paciente se realiza de forma manual desde que se recibe la llamada de urgencia hasta que se presenta un diagnóstico.

Al ejecutarse manualmente el proceso de diagnosticar a un paciente el soporte de los datos, lo constituyen los documentos, que al ser de papel; son más vulnerables a eventos como deterioro, humedad e incendios identificando, así como uno de los principales problemas la pérdida de información causado por la acumulación de documentos generados, pues, el centro cuenta con un total de 865 plaguicidas, y genera aproximadamente un total de 500 casos anualmente. Analizar esta cantidad de documentos para llegar a un posible diagnóstico hace que el proceso pueda tardar alrededor de 10 horas en la mayoría de los casos. Esto genera un proceso lento que puede traer como consecuencias la pérdida de vidas humanas, pues existen plaguicidas que causan la muerte en tiempos específicos si no son contrarrestados por un antídoto.

Por lo anteriormente planteado se identifica como **problema a resolver**: ¿Cómo disminuir el tiempo de análisis de los casos de intoxicación por plaguicidas que llegan al Cenatox, para agilizar el diagnóstico al paciente?

El problema está enmarcado en el **objeto de estudio**: proceso de análisis de información epidemiológica en Cuba, centrado en el **campo de acción**: análisis de la información de intoxicaciones por plaguicidas en el Cenatox.

Para solucionar el problema planteado, se define como **objetivo general**: Desarrollar un sistema informático para el análisis de las intoxicaciones por plaguicidas, que permita disminuir el tiempo de respuesta a casos intoxicados.

Tareas de investigación:

1. Elaboración del marco teórico referente a la toxicología derivada al uso de plaguicidas.
2. Modelación del negocio y los requisitos del sistema a partir del análisis de las características, particularidades y procesos de negocio del Centro Nacional de Toxicología de Cuba.

3. Diseño del sistema informático y la base de caso para la gestión y el análisis de la información asociada a las intoxicaciones por plaguicidas en el Cenatox.
4. Validación de los resultados obtenidos a partir de los métodos definidos en la investigación.

Los **métodos científicos** utilizados para desarrollar la investigación fueron:

Teóricos:

- **Analítico-sintético:** se utiliza en la fundamentación del objeto de estudio de la investigación a partir de los conceptos asociados y de la comprensión del negocio.
- **Inductivo-deductivo:** se utiliza en la aplicación de casos de pruebas al sistema, llegando a conclusiones a partir de las respuestas proporcionadas por este.

Empíricos:

- **Observación:** se utiliza para observar cómo se realiza el proceso de recolección y gestión de los datos de los casos de intoxicaciones por plaguicidas en el Cenatox. Permite analizar y comparar los resultados obtenidos de las pruebas realizadas a la aplicación desarrollada.
- **Entrevista:** se utiliza para recopilar información mediante conversaciones con el cliente, para satisfacer sus necesidades y obtener un producto con la calidad requerida.
- **Análisis documental:** se utiliza para identificar los elementos de las hojas de plaguicidas que forman parte de la propuesta y definir los criterios a tener en cuenta para la emisión de diagnósticos.

Para cumplimentar las tareas de investigación el documento se estructura en tres capítulos, siendo estos:

Capítulo 1. Fundamentación teórica referente a la toxicología derivada al uso de plaguicidas: trata los conceptos fundamentales sobre los diferentes elementos en la que se basa la presente investigación, incluye un análisis del estado del arte, a nivel nacional e internacional de sistemas que permitan generar información oportuna y rápida para la prevención y control de eventos y se hace una descripción de las tecnologías y herramientas usadas para el desarrollo del sistema propuesto.

El **Capítulo 2. Características del sistema de gestión y análisis de casos de intoxicación por plaguicida:** describe la propuesta de solución, junto a la modelación del flujo de información del componente ha desarrollado, se describen los patrones arquitectónicos y los de diseños utilizados; se realiza una explicación de la técnica utilizada y del procedimiento seguido para el desarrollo del sistema.

Por último, en el **Capítulo 3. Implementación y validación del sistema**, se valida la aplicación, a partir de un caso de estudio.

Capítulo 1. Fundamentación teórica referente a la toxicología derivada al uso de plaguicidas

Se abordan los elementos de la base teórico-conceptual de la investigación para el desarrollo de la solución propuesta, basándose en el análisis y valoración de forma crítica, de las tendencias, tecnologías actuales y antecedentes asociados al campo de acción. Se selecciona la metodología de desarrollo que guiara el proceso de desarrollo de software y se realiza una caracterización de las distintas soluciones existentes que están en correspondencia con el objeto de estudio de la investigación y se ilustran las etapas que caracterizan el desarrollo de los sistemas basado en casos. Se describen y justifican las características fundamentales de las herramientas y tecnologías de software utilizadas para el desarrollo del sistema.

1.1. La toxicología en Cuba

En Cuba existe una actividad agrícola importante, debido a que es un país que basa su economía fundamentalmente en la agricultura. El empleo de plaguicidas ha ido incrementándose en las últimas décadas, con el afán de mejorar la tecnología agrícola e incrementar la producción.

Cuba cuenta con tres Centros de Información, Asesoramiento y Asistencia Toxicológica, que cuentan con personal especializado para asesorar sobre el tratamiento y la prevención de las intoxicaciones y facilitar información sobre medicamentos, plaguicidas, plantas y animales venenosos, productos de uso doméstico y sustancias químicas. Los especialistas de estos centros están capacitados para recomendar las acciones a ejecutar cuando las personas entran en contacto con algún plaguicida (CENATOX, 2017).

- En la provincia de la Habana el CENATOX con un servicio de 24 horas:
- En la provincia de Villa Clara el UTEX
- En la provincia de Ciego de Ávila
- En la provincia de Santiago de Cuba TOXIMED

Durante muchos años, el combate químico de plagas ha ayudado al hombre a proteger sus cosechas. Sin embargo, paradójicamente, cada año deben ser aplicadas dosis mayores de los plaguicidas para mantener el rendimiento de los cultivos.

Hoy en día, se puede calificar de masivo el uso de plaguicidas por parte de los agricultores, quienes han llegado a depender casi totalmente de estos compuestos. Este uso intensivo de plaguicidas está produciendo efectos negativos a nivel económico, agrícola, ambiental y en la salud. Teniendo en cuenta el uso cada vez más frecuente de plaguicidas, además de que existe por parte del personal facultativo en los niveles de atención primaria y secundario de salud poco dominio del tema de manera general, lo que dificulta este diagnóstico cuando no hay evidencia cierta de la exposición a estos tóxicos y no se sospecha clínicamente la misma, a todo ello se le añade que no existe un protocolo actualizado y estandarizado para

el tratamiento de las intoxicaciones por plaguicidas en los servicios de urgencias de instituciones de salud del Ministerio de Salud Pública (MINSAP).

La mayoría de las personas que manejan los plaguicidas no tienen la preparación adecuada sobre su manipulación y aplicación. Estas deben ser personas adultas y adiestradas en el manejo de plaguicidas, evitando siempre que la manipulación la realicen niños, o mujeres en edad fértil. El almacenamiento muchas veces no es el adecuado. En ocasiones se guardan dentro de las propias viviendas dejándolos al alcance de los familiares.

Cuando una persona entra en contacto con una sustancia tóxica se dice que está expuesta. El efecto de la exposición dependerá de la duración del contacto, del mecanismo por el que el tóxico ingresa en el cuerpo, y también de la cantidad de sustancia tóxica que el organismo puede eliminar durante ese tiempo. La exposición puede ser única o producirse de manera repetida. Por exposición aguda se entiende un simple contacto que dura segundos, minutos u horas, o bien una, sucesión de exposiciones durante un día como máximo. Por exposición crónica se entiende un contacto que dura días, meses o años. Puede ser continua o estar interrumpida por intervalos en los que no se produce ese contacto. La exposición que sólo se produce en el trabajo, por ejemplo, no es continua, aun cuando sea crónica.

La exposición crónica a pequeñas cantidades de una sustancia tóxica puede no dar ningún síntoma o signo de intoxicación al principio. A veces pasan muchos días o meses antes de que el cuerpo albergue suficiente cantidad de sustancia química para que haya intoxicación. Una persona, por ejemplo, puede utilizar a diario un plaguicida, exponiéndose cada día a una pequeña cantidad de éste; ahora bien, la cantidad de plaguicida que se va depositando en el cuerpo aumenta gradualmente hasta que, al cabo de muchos días, se convierte en una dosis tóxica. En ese momento es cuando la persona empieza a sentirse mal (R., 2002).

Clasificación:

Los plaguicidas se pueden clasificar según el grupo químico al que pertenecen, por su acción específica y también de acuerdo a su toxicidad. Algunos ejemplos de estos tipos de clasificaciones son los siguientes (10Mmalagón Cañizares, 2010).

- **Insecticidas**

Los insecticidas son sustancias químicas de origen mineral u orgánico capaces de producir la muerte de los insectos. Los insecticidas polivalentes o de amplio espectro destruyen un grupo numeroso de especies, lo cual constituye una ventaja, pero a su vez un posible inconveniente, ya que pueden provocar la muerte de insectos útiles y un desequilibrio biológico que puede producir el desarrollo de otras plagas que estaban controladas por sus enemigos naturales. Los llamados insecticidas específicos actúan sólo contra una o pocas especies y respetan más a la fauna útil.

- **Herbicidas**

Son productos químicos de origen mineral o de síntesis orgánica que controlan las hierbas no deseadas. El problema de muchos herbicidas es que, al igual que matan a la planta no deseada, pueden hacerlo con la cultivada o provocarle daños. Por ello requieren ser aplicados correctamente y en los momentos adecuados, teniendo en cuenta tanto el estado de desarrollo del cultivo como el de las plantas a controlar.

- **Rodenticidas**

Los rodenticidas son productos que se emplean para la lucha contra ratas, ratones y topillos. Hay rodenticidas inorgánicos, como los fosfuros de magnesio, aluminio y calcio, que al ser ingeridos por los animales reaccionan con los ácidos del estómago produciendo gases tóxicos (la aplicación de estos productos requiere de un carné específico). También existen rodenticidas orgánicos, como los del grupo de las super-warfarinas. Su acción es anticoagulante, de forma que el animal al hacerse una herida, ésta no cicatriza y se desangra.

- **Fungicidas**

Son sustancias químicas de origen mineral u orgánico para el tratamiento de las enfermedades producidas por hongos. Los hay que destruyen el micelio y las esporas del hongo, que detienen su desarrollo e impiden la germinación de las esporas, o que impiden la reproducción del hongo.

Elementos a tener en cuenta para el diagnóstico de la intoxicación aguda por plaguicidas.

Los plaguicidas nunca se comercializan en forma de ingrediente activo sino únicamente como productos ya formulados, o sea ingrediente(s) activo(s) más otros componentes de formulaciones, como por ejemplo solventes, adherentes, surfactantes, etc.

La Dosis letal media oral aguda (DL50) significa la "cantidad de una sustancia que es necesario ingerir de una sola vez para producir la muerte del 50% de los animales de una población. Esta dosis se expresa generalmente en mg/kg de peso del animal empleado como modelo de ensayo. La DL50 de estos últimos es la que resulta de importancia para el médico.

Una de las clasificaciones de los plaguicidas es de acuerdo con su toxicidad, esta se representa con una banda de colores en la parte baja de las etiquetas. Estas bandas se corresponden con los colores que aparecen en la Tabla 1 (López Arias, 2015).

Tabla 1: Clasificación de plaguicidas según riesgo recomendada por la OMS.

Palabra de advertencia en la etiqueta	Clases de Riesgo		DL50 para la rata(mg/kg peso corporal)			
			Vía Oral		Vía Dérmica	
			Sólidos	Líquidos	Sólidos	Líquidos
“Muy Tóxico”	Ia	Extremadamente tóxico	5 o menos	20 o menos	10 o menos	40 o menos
“Tóxico”	Ib	Altamente tóxico	5-50	20-200	10-100	40-400
“Nocivo”	II	Moderadamente tóxico	50-500	200-2000	100-1000	400-4000
“Baja Peligrosidad”	III	Ligeramente tóxico	>500	>2000	>1000	>4000
Precaución	IV	Menos tóxico	>2000	>3000	-----	-----

A su vez existen Hojas de Seguridad (HDS) creadas por empresas comercializadoras de estas sustancias las cuales contiene una gama de información. Una HDS proporciona información básica sobre un material o sustancia química determinada. Esta incluye, entre otros aspectos, las propiedades y riesgos del material, como usarlo de manera segura y que hacer en caso de una emergencia. El objetivo de este documento es el de proporcionar orientación para la comprensión e interpretación de la información presentada (Gob.mx, 2015).

La información de las HDS está organizada en secciones. Los nombres y contenidos específicos de estas pueden variar de un proveedor a otro. La información que se presenta es general y resumida pero esencial para cualquier usuario. El Cenatox depende de las HDS, pues basa sus diagnósticos en la información que contiene acerca de cada sustancia.

1.2. Análisis del estado del arte

Se realiza un análisis sobre las herramientas y software existentes utilizados en centros hospitalarios y en la agricultura teniendo en cuenta tres aspectos fundamentales: si emiten diagnósticos, gestionan casos de intoxicación y si filtran información. Con el objetivo de determinar si pueden ser utilizada en el Cenatox o es necesario la implementación de una nueva herramienta para la gestión de información de los casos de intoxicaciones por plaguicidas en el Cenatox.

DiagnosMD

DiagnosMD (DiagnosMD, 2017) es una potente herramienta de utilidad continua en la consulta, que ayuda al diagnóstico al combinar un conjunto de datos (síntomas, signos, resultados analíticos anormales, datos de Rx, etc) con el país, sexo y edad del paciente, ofreciendo con criterio un listado de enfermedades

posibles, con potentes herramientas para afinar en el diagnóstico. Algunos de los argumentos que hace que esta herramienta sea de vital importancia son:

- En Medicina lo más difícil es el diagnóstico.
- No se diagnostica la enfermedad en la que no se piensa.
- Diagnos ayuda a pensar en enfermedades poco frecuentes o raras (representan el 85% de las descritas en Diagnos).
- La memoria humana es limitada, por lo que DiagnosMD es una gran ayuda.

Además, permite consultar rápidamente enfermedades a través de una búsqueda filtrada de datos, etc. pues por cada campo de un filtro que se complete se van descartando posibles enfermedades.

DiagnosMD contiene una gama de utilidades que permite evitar errores en la medicación tales como contradicciones y más de un diagnóstico. Esto resulta de gran utilidad para dar el diagnóstico a pacientes pues es más preciso en los resultados.

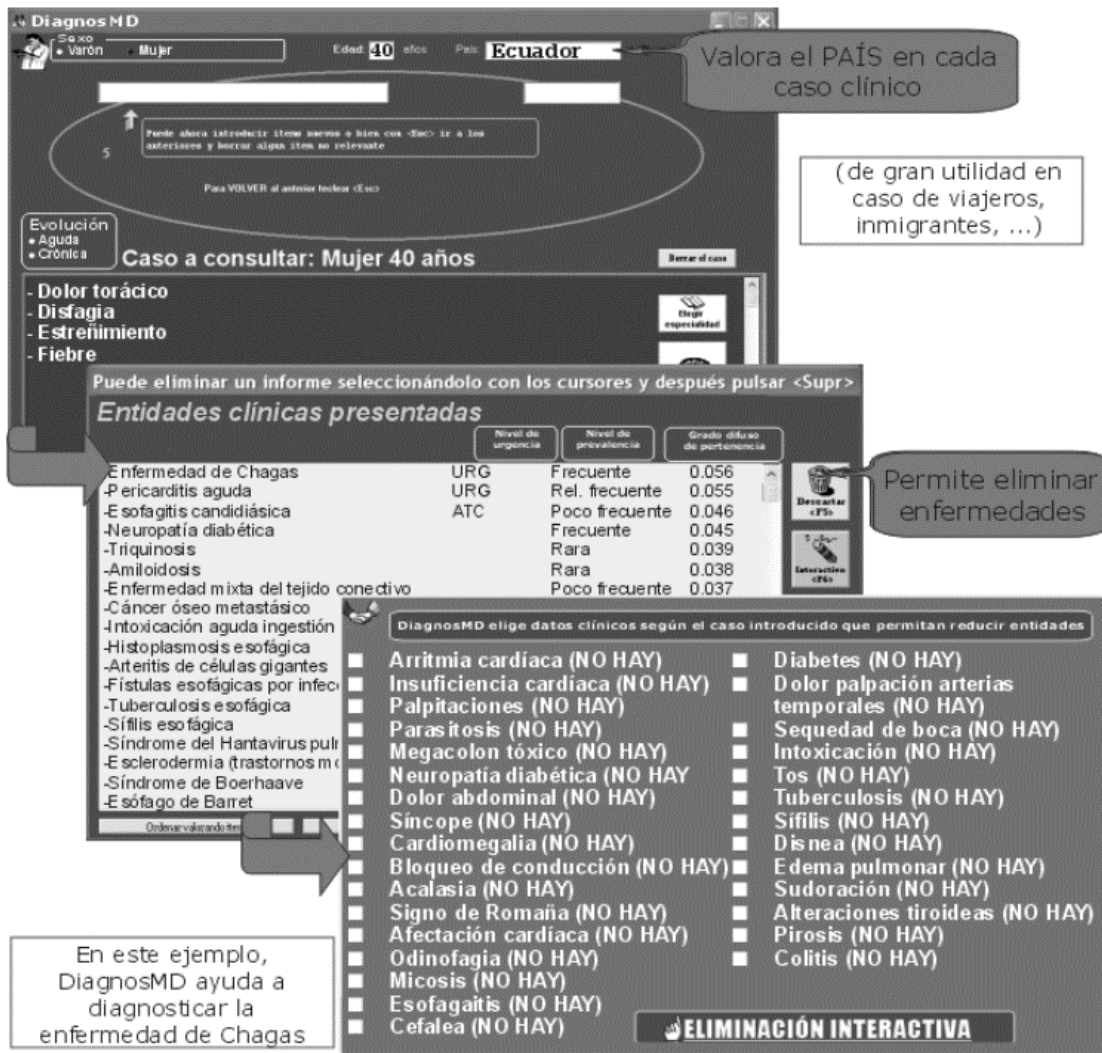


Figura 1: Interfaces de la Herramienta DiagnosMD. Fuente: (DiagnosMD, 2017)

Software de apoyo a la selección de plaguicidas para el cultivo agrícola (Ortiz Gálvez, 2012)

Software utilizado en la agricultura que sugiere a un agricultor el o los plaguicidas que podrían eliminar una determinada plaga de su plantación, dependiendo del cultivo que tenga, la plaga que lo afecta, los días restantes para su próxima cosecha y el país donde pretende comercializar sus productos. Algunas de las características que lo componen son:

- Buscar información de un determinado plaguicida por el nombre comercial que posea.
- Obtener una lista de plaguicidas sugeridos según datos como el cultivo al que se desea aplicar el producto, plaga que se desea combatir, días restantes para la próxima cosecha y el país de destino de la exportación.

El software entrega información sobre la aplicación de un plaguicida en un cultivo a los agricultores asegurándose de que se cumplan las medidas de seguridad en el momento de su aplicación. Para ello cuenta con funcionalidades que permiten:

- Realizar búsquedas de información detallada según el nombre comercial de un determinado plaguicida.
- Desplegar una lista de plaguicidas sugeridas según datos de entrada, estos son: cultivo, nombre común de la plaga, nombre científico de la plaga, días para la próxima cosecha y país de destino de la producción.
- Además, se debe permitir acceder, desde esta lista, a la información detallada de los plaguicidas sugeridos.

Lista de Plaguicidas Posibles

Cultivo: Peras.
Plaga y/o Enfermedad: Acaro del Russet del Peral.
Próxima Cosecha: 18 días.
País de Destino: Chile.

Plaguicida	Carencia en Chile	Carencia del Fabricante	Cumple Condiciones
Abamite	No Exigido	10 días.	<input checked="" type="checkbox"/> Cumple
Pyrinex 48 EC	No Exigido	10 días.	<input checked="" type="checkbox"/> Cumple
Ferbam 76 WG	No Exigido	15 días.	<input checked="" type="checkbox"/> Cumple
Mancozeb 80% ...	No Exigido	15 días.	<input checked="" type="checkbox"/> Cumple
Imidan 70 WP	No Exigido	20 días.	<input type="checkbox"/> No Cumple
Cyren 15 G	No Exigido	25 días.	<input type="checkbox"/> No Cumple
Metomil 90% PS	No Exigido	30 días.	<input type="checkbox"/> No Cumple
Diazol 50 EW	No Exigido	35 días.	<input type="checkbox"/> No Cumple

La presente lista es una sugerencia en base a información obtenida del fabricante de los product...

Figura 2: Interfaces del software que muestra un listado de los posibles plaguicidas dado un caso. Fuente: (Ortiz Gálvez, 2012).

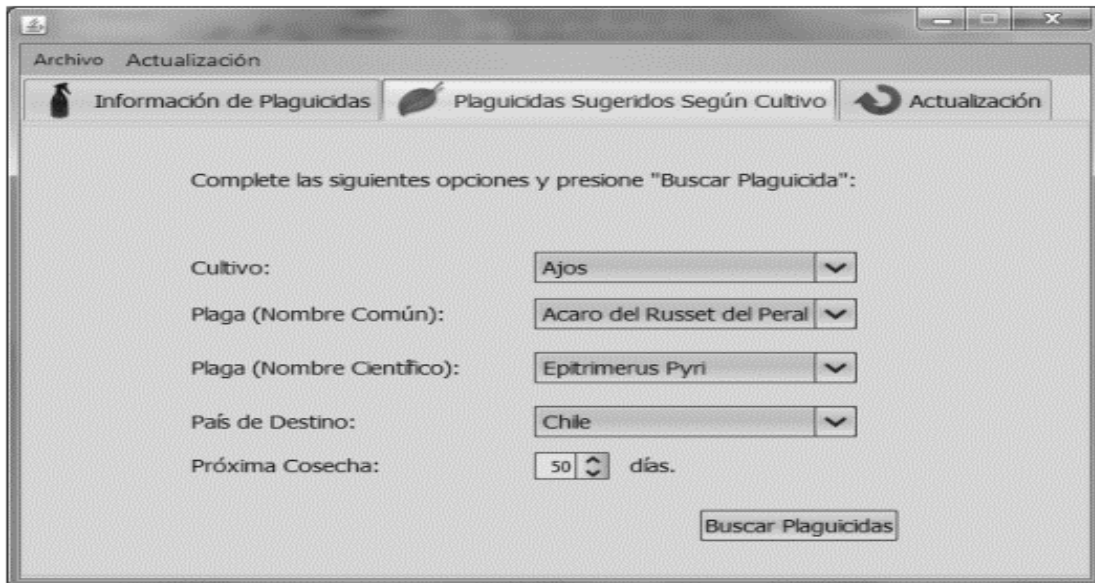


Figura 3: Interfaces del software que muestra los plaguicidas sugeridos según cultivo. Fuente: (Ortiz Gálvez, 2012).

Registro y Estadística de Consultas Toxicológica (Rectox)

Rectox (Rectox, 2017) es un software especialmente diseñado para el Registro de Historias Clínicas (personales y telefónicas) en Centros de Toxicología y para elaborar estadísticas e informes con los datos registrados. Es lo suficientemente versátil como para adecuarlo a las necesidades particulares de cada centro, por ejemplo, se va realizando la propia base de sustancias tóxicas del Centro al ir grabando cada tóxico.

Nace a partir de la necesidad de mantener un registro de historias clínicas y confeccionar una base de datos para la elaboración de estadísticas en el Servicio de Toxicología del Sanatorio de Niños, Rosario (Sertox). La posibilidad de elaborar estadísticas permite conocer las características de las demandas de cada servicio.

Una de las principales ventajas que posee Rectox son sus requerimientos tecnológicos; es un software ligero que funciona correctamente sobre cualquier máquina que cuente solamente con:

- Procesador: Intel 486 o equivalente o posterior
- Disco duro: 200Mb o más
- Software: Microsoft Windows 95

Y para su utilización solo se hace necesario el conocimiento mínimo respecto a Toxicología. No es necesario ser especialista y puede ser operado por personal administrativo pero los mejores resultados se obtienen con personal que conozca el significado de los datos que está registrando.

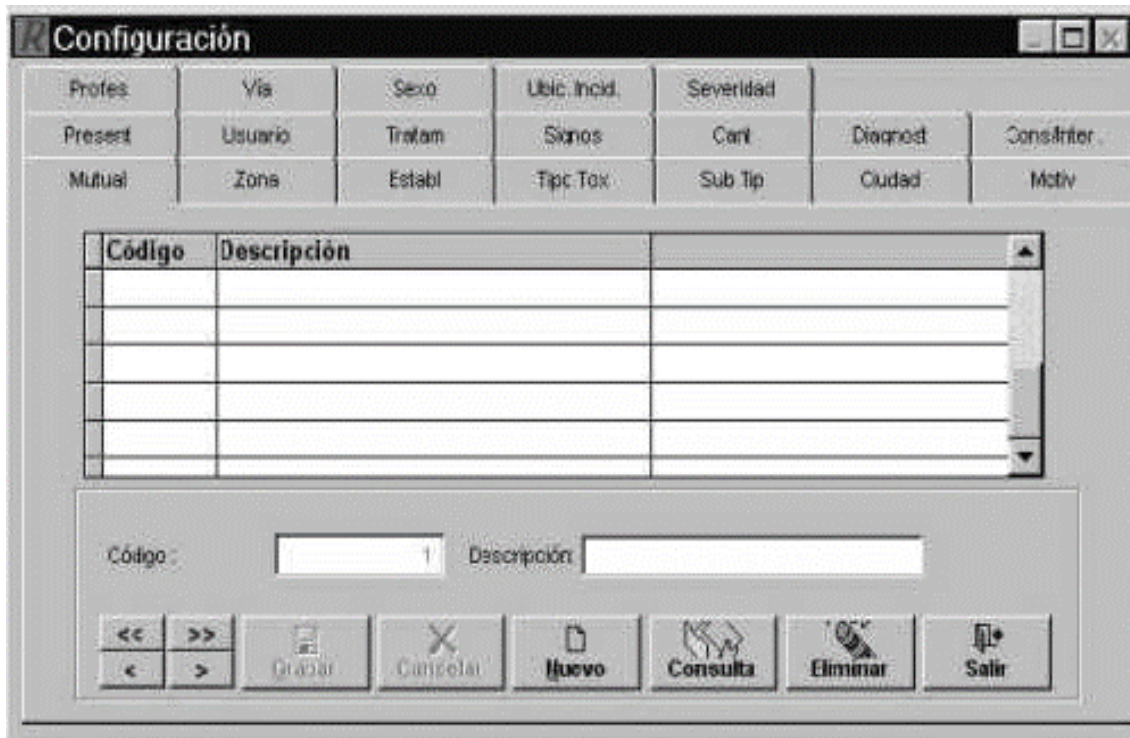


Figura 4: Interfaces Grafica del software Rectox. Fuente: (Rectox, 2017)

En la tabla 2 se muestra un resumen de la comparación realizada a las tres herramientas analizadas, atendiendo a tres aspectos fundamentales para el Cenatox.

Tabla 2: Comparación de las herramientas y software estudiados.

	Emiten diagnóstico	Gestión de casos de intoxicaciones	Filtrado de información
Software de apoyo a la selección de plaguicidas para el cultivo agrícola	Si	No	Si
Rectox	Si	No	Si
DiagnosMD	Si	No	Si

Una vez finalizado el estudio de los sistemas de gestión de información epidemiológica e intoxicaciones se concluye que se tiene que implementar un nuevo software capaz de dar solución a los requisitos exigidos, pues las herramientas y software existentes, en general, no son capaces de contribuir a la recolección y gestión de datos acerca de casos de intoxicaciones por plaguicidas presentando un diagnóstico rápido y conciso. Además, estas no tienen soporte funcional ni propician la sostenibilidad a largo plazo debido a que no se desarrollaron en el país y no existe un contrato con los desarrolladores.

1.3. Sistemas basados en conocimientos

El incremento de los datos en el sector de la salud, ha ocasionado que el análisis de la información cada vez sea más complejo. Con el creciente uso de las tecnologías en el área y los grandes volúmenes de dichos datos que se han generado, se hace necesario introducir nuevas tecnologías para el análisis de estos a partir de las bases de datos que existen sobre estos eventos.

Específicamente en Cuba en el Cenatox se evidencia el atraso de la informatización con la que cuenta el área de toxicología. Si existiera un sistema que gestionara toda la información de toxicología asociada a los pacientes o personas que entran en contacto con algún plaguicida y se ven intoxicados o sufren algún tipo de lesión, sería útil contar con un mecanismo o herramienta que propiciara predecir los casos de intoxicaciones a partir de varios criterios de búsqueda en los datos almacenados en los que se apliquen términos o técnicas de la inteligencia artificial.

En el Cenatox se gestionan los casos asociados a intoxicaciones con diferentes plaguicidas en Cuba. Ante esto se desea predecir el comportamiento ante el contacto con los plaguicidas a partir de determinados y ciertos síntomas que presenta el sujeto a su llegada al área de salud. Por lo que se hace necesario analizar toda la información con la que cuenta para realizar comparaciones, buscar semejanzas y obtener un resultado.

Para procesar información existen diversas técnicas dígase, minería de datos (Orallo, 2004), minería de texto (Montes, 2001), minería de procesos (Pérez Jiménez, 2015), redes neuronales (Freeman, y otros, 1993) y sistemas expertos (Giarratano, y otros, 2001) todos de la familia de la inteligencia artificial (Cruz, y otros, 2011). Entre los que más se destacan y acondicionados al tipo de sistema que se quiere desarrollar se encuentran los sistemas expertos específicamente los Sistemas Basados en Casos (SBC).

En términos generales, un SBC puede ser definido como: un sistema computarizado que usa conocimiento sobre un dominio para arribar a una solución de un problema de ese dominio. Esta solución es esencialmente la misma que la obtenida por una persona experimentada en el dominio del problema cuando se enfrenta al mismo problema (Martínez Sánchez, 2009).

Estos sistemas son muy utilizados en la actualidad pues entre sus propósitos destacan (Ayala, 2006):

- Aprender.
- Evolucionar.
- Adaptar.
- Razonar.
- Analizar problemas.
- Generar alternativas de solución.

- Emular al experto humano.
- Generar conocimiento a partir del que ya se posee.

Y entre los que se destaca la toma de decisiones, implementando así estos sistemas para reducir en gran medida el error humano.

Los SBC imitan la característica que tienen los seres humanos de buscar solución o arribar a conclusiones de un problema a través de experiencias previas en situaciones similares. Los elementos de mayor importancia en este tipo de sistemas son la base de conocimientos (Badaró, 2013) o casos y el motor de inferencia (Badaró, 2013). Existen dos clasificaciones para los SBC, sistemas interpretativos y sistemas que solucionan problemas. Los primeros toman una solución de un problema que no está bien comprendido y tienen, un argumento o crítica de salida sobre esta solución, normalmente se emplean en problemas de clasificación, argumentación o predicción. Los solucionadores son los que mediante la información que obtienen del problema construyen su solución analizando la base de casos.

Para que un SBC encuentre una solución a un nuevo problema debe ejecutar las siguientes etapas (Martí Pérez, 2012):

- **Recuperar:** obtiene los casos similares almacenados en la base de casos.
- **Reutilizar:** utiliza los casos recuperados para confeccionar la solución de una nueva entrada. Esta nueva solución no necesariamente tiene que ser la misma que en algún caso recuperado puesto que pueden adaptarse.
- **Revisar:** verifica si la solución encontrada es correcta o si contiene algún aspecto específico que deba ser transformado. El experto en el dominio valida finalmente si la solución es viable o no.
- **Recordar:** guardar la nueva solución en la base de casos para que pueda ser utilizada en futuras consultas.

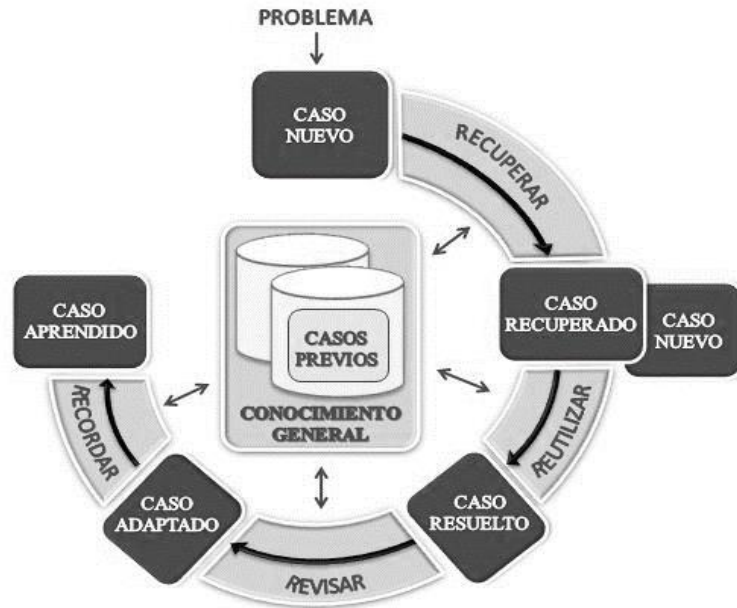


Figura 5: Ciclo del razonamiento basado en caso. Fuente: (Martí Pérez, 2012)

Los casos se representan mediante un conjunto de características que definen al problema en cuestión. Estas características se dividen en: rasgos predictores y la solución según los valores de dichos rasgos. De acuerdo a la situación que se quiera representar en una base de conocimientos se define la estructura de los casos. Para esto se tiene en cuenta cuáles son sus características más relevantes o las de mayor impacto para la solución. Se deben definir, además, los algoritmos que van a operar en el sistema en cada una de las etapas, desde la recuperación hasta la validación de la solución, para determinar si el nuevo caso pasa o no a formar parte de la base de conocimientos. Para poder identificar casos similares al que se trata de resolver es necesario emplear un método de comparación eficaz.

Tipos de búsqueda en los sistemas expertos

Durante la recuperación de los casos, se necesita realizar un proceso de búsqueda, el cual muestre como resultados el o los casos que podrán ser tomados como solución al problema. A continuación, se explica los tipos de búsqueda que se pueden utilizar durante el proceso:

- **Búsqueda por analogía**

La búsqueda por analogía, significa determinar si la descripción del problema es equivalente a la descripción de alguno de los casos almacenados, es decir, probar la equivalencia entre los valores de los rasgos predictores, aunque sean diferentes, comprobando si semánticamente tienen igual significado.

Para este análisis se utiliza una red semántica, en la cual los nodos representan valores y rasgos. Además, los enlaces indican el grado de relación entre ellos. Dos valores son equivalentes si están enlazados con el mismo concepto por el mismo tipo de relación, a partir del mismo punto de vista (Agudo, 2001).

- **Búsqueda por semejanzas**

La búsqueda por semejanzas, tiene como objetivo obtener el caso almacenado que más se parece al nuevo problema. Existen diversas formas de medir el grado de semejanza, estas pueden ser, desde expresiones analíticas de distancia hasta la descripción de algoritmos para obtener el grado de parecido entre dos objetos. Esto depende del tipo de información que se presente (Rodríguez, 2003).

Considerando que un caso está descrito por un número de características, una aproximación sencilla sería contabilizar los atributos comunes entre el caso almacenado y el nuevo problema. Se suele distinguir entre medidas de semejanzas locales y globales, las primeras calculan la semejanza entre valores de un mismo atributo en los dos casos y la segunda combina los resultados de aplicar la similitud local a cada uno de los atributos de los casos que se están comparando. Una estrategia para definir semejanza entre atributos simbólicos, consiste en utilizar una tabla de semejanzas, donde se obtiene explícitamente la similitud entre cada par de valores (Rodríguez, 2003).

Las funciones de semejanza indican cómo se deben comparar los casos, pero es necesario saber qué se debe comparar. De todos los atributos que conforman un caso, es necesario definir cuáles son los más relevantes para efectuar la comparación. Una vez que se decide realizar el proceso de búsqueda utilizando funciones de semejanza, es necesario definir en dependencia del dominio de aplicación, las funciones que pueden emplearse. Existen varias funciones de semejanzas que han sido propuestas para recuperar casos semejantes, muchas de ellas trabajan con atributos numéricos y no son apropiadas para atributos simbólicos y otras permiten buscar semejanza teniendo en cuenta una estructura determinada (Martínez Sánchez Natalia, 2008).

Justificación de la utilización de la búsqueda por semejanza

Para la recuperación de los casos, se propone utilizar el tipo de búsqueda por semejanza. En el momento de ingresar el caso nuevo se buscará si existe un caso semejante que cumpla con la medida de semejanza local, en caso de no existir, se aplica la semejanza global utilizando la función de semejanza.

Funciones de semejanza para los casos donde las características que los representan son numéricas (Martí Pérez, 2012).

- Distancia Euclidiana
- Distancia de Manhattan
- Distancia de Minkowsky
- Coeficiente de Duran y Odell

Funciones de semejanza para los casos donde las características que los representan son simbólicas (Martí Pérez, 2012).

- Distancia de Hamming
- Value Difference Metric (VDM)

Funciones para calcular la semejanza en dominios de problemas donde las características son numéricas y no numéricas incluyendo información incompleta. Se considera como información incompleta, los valores desconocidos en un momento dado para un atributo (Martí Pérez, 2012).

- Función de Gower
- Función de Argelio

Los SBC se desempeñan de diferentes maneras, no obstante, cuando estos están encaminados específicamente a resolver problemas en un entorno de la misma forma que lo haría un especialista, empleando para ello conocimiento de otros especialistas, entonces a este tipo de sistema se le denomina Sistema Experto.

Para el desarrollo del software se implementará un Sistema Basado en Conocimientos específicamente un Sistema Basado en Casos, pues dado sus principios y características se concluye que será una poderosa técnica en la toma de decisiones que en cuestiones prácticas del trabajo en el Cenatox puede salvar vidas en cuestiones de segundos.

1.3.1 Razonamiento basado en casos

El proceso de adquisición del conocimiento es una de las tareas fundamentales en el desarrollo de un SBC, mediante este proceso, el ingeniero de conocimiento confecciona y organiza la información necesaria para el desarrollo del sistema, este proceso transita por varias etapas las cuales se muestran a continuación en la figura 6:



Figura 6: Etapas de la adquisición de conocimiento. Fuente: (Martí Pérez, 2012).

A continuación, se detallan cada una de las etapas anteriores (Martí Pérez, 2012):

- **Identificación:** en esta etapa se identifican los participantes, tanto expertos como los ingenieros del conocimiento que intervendrán en la creación de la base de datos, se especifican además los

aspectos fundamentales que se trabajarán para ubicar a los especialistas en lo que se desea hacer e identificar los recursos necesarios dígame fuente de conocimiento, tiempo, facilidades de computo.

- **Conceptualización:** se especifican y relacionan los aspectos fundamentales que se trabajarán. Se identifica el flujo de información y los tipos de datos de que se dispone.
- **Formalización:** se define la forma en que se pudiese representarse la información, lo que llevaría también a identificar la herramienta necesaria para construir la base de conocimiento.
- **Implementación:** se codifica la información extraída de los expertos según la estrategia definida y la herramienta seleccionada.
- **Prueba:** se realizan pruebas a la base de conocimiento para identificar sus potencialidades y debilidades y con ello determinar si es necesario realizarles cambios al modelo o la forma de representación empleada.

1.3.2 Técnicas de adquisición del conocimiento

Las técnicas de adquisición del conocimiento son aquellas que permiten obtener el conocimiento necesario para conformar la base de datos con que el sistema va a funcionar. En esta investigación se emplearon las siguientes:

- **Consulta a expertos:** este método consiste en realizar encuestas y entrevistas a especialistas en el tema para extraer el conocimiento de que disponen para incluirlo en el sistema. También para verificar la veracidad de los datos una vez conformada la base de conocimiento. En la conformación y validación de la base de casos del SBC se realizaron consultas y entrevistas a especialistas del Cenatox.
- **Conocimiento documentado:** el conocimiento es extraído de distintas fuentes tales como libros, revistas y otros documentos que contengan información sobre el tema. En la investigación en curso se emplearon datos de historias clínicas de pacientes y las HDS de plaguicidas.

Mediante la consulta con especialistas y los documentos estudiados se identificaron un conjunto de 46 rasgos predictores, 865 rasgos objetivos que reflejan elementos indispensables a tener en cuenta para indicar un tratamiento.

1.4. Metodología de desarrollo

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos de software (Apuntes, 2017). Estas a su vez se dividen en ágiles y tradicionales.

Las metodologías tradicionales se focalizan en documentación, planificación y procesos. Plantillas, técnicas de administración, revisiones, etc. Ejemplos de estas metodologías son: Proceso Racional Unificado (RUP) y Microsoft Solution Frame (MSF).

Las metodologías ágiles ponen de relevancia que la capacidad de respuesta a un cambio es más importante que el seguimiento estricto de un plan. Para muchos clientes esta flexibilidad será una ventaja competitiva y porque estar preparados para el cambio significa reducir su coste. Ejemplos de estas metodologías son Programación Extrema (XP) y Proceso Unificado Ágil (AUP) (Pressman, 2007).

Principios de agilidad

Se define 12 principios de agilidad para aquellos que la quieran alcanzar (Pressman, 2007):

1. La prioridad más alta es satisfacer al cliente a través de la entrega pronta y continua de software valioso.
2. Son bienvenidos los requerimientos cambiantes, aun en una etapa avanzada del desarrollo. Los procesos ágiles dominan el cambio para provecho de la ventaja competitiva del cliente.
3. Entregar con frecuencia software que funcione, de dos semanas a un par de meses, de preferencia lo más pronto que se pueda.
4. Las personas de negocios y los desarrolladores deben trabajar juntos, a diario y durante todo el proyecto.
5. Hay que desarrollar los proyectos con individuos motivados. Debe darse a éstos el ambiente y el apoyo que necesiten, y confiar en que harán el trabajo.
6. El método más eficiente y eficaz para transmitir información a los integrantes de un equipo de desarrollo, y entre éstos, es la conversación cara a cara.
7. La medida principal de avance es el software que funciona.
8. Los procesos ágiles promueven el desarrollo sostenible. Los patrocinadores, desarrolladores y usuarios deben poder mantener un ritmo constante en forma indefinida.
9. La atención continua a la excelencia técnica y el buen diseño mejora la agilidad.
10. Es esencial la simplicidad: el arte de maximizar la cantidad de trabajo no realizado.
11. Las mejores arquitecturas, requerimientos y diseños surgen de los equipos con organización propia.
12. El equipo reflexiona a intervalos regulares sobre cómo ser más eficaz, para después afinar y ajustar su comportamiento en consecuencia.

1.4.1. Metodología Programación Extrema (XP)

XP es una metodología ágil que se basa en las relaciones interpersonales como clave para el desarrollo de software propiciando un buen ambiente de trabajo. Basa su funcionamiento de la retroalimentación continua

entre el cliente y el equipo de desarrollo además de la simplicidad en las soluciones implementadas (Pressman, 2007).

Ventajas (Pressman, 2007):

- Apropiado para entornos volátiles.
- Estar preparados para el cambio, significa reducir su coste.
- Planificación más transparente para nuestros clientes, conocen las fechas de entrega de funcionalidades. Vital para su negocio.
- Permitirá definir en cada iteración cuales son los objetivos de la siguiente.

Valores de la metodología XP (Pressman, 2007)

Para todo trabajo que se realiza con la metodología XP se definen cinco valores, ellos son: comunicación, simplicidad, retroalimentación, valentía y respeto. Cada uno de estos valores se usa como un motor para actividades, acciones y tareas específicas de XP.

Para el desarrollo de la presente investigación se decidió utilizar como metodología de desarrollo XP, por ser esta la que más se adecua a las características del equipo de trabajo, la duración del proyecto y la relación con el cliente. La duración del proyecto es relativamente corta, el equipo de trabajo está formado por dos programadores entre los cuales existe un alto nivel de interacción debido a su buena comunicación y entendimiento.

Desarrollo mediante XP

El desarrollo de aplicaciones informáticas utilizando la metodología XP usa un enfoque orientado a objetos como paradigma preferido de desarrollo, y engloba un conjunto de reglas y prácticas que ocurren en el contexto de cuatro actividades estructurales: planeación, diseño, codificación y pruebas (Pressman, 2007).



Figura 7: Ilustra el proceso XP. Fuente: (Pressman, 2007)

Artefactos que se generan en la metodología XP (Villegas, 2017)

- **Historias de usuario (HU):** representan una breve descripción del comportamiento del sistema y emplea terminología del cliente sin lenguaje técnico. Se realiza una por cada característica principal del sistema y se emplean para hacer estimaciones de tiempo para el plan de entregas. Difieren de los casos de uso porque son escritas empleando terminología del cliente, son más amigables que los casos de uso formales.
- **Pruebas de aceptación:** permite confirmar que la historia ha sido implementada correctamente.
- **Tarjetas CRC:** estas tarjetas se dividen en tres secciones que contienen la información del nombre de la clase, sus responsabilidades y sus colaboradores.

Esta metodología también define un conjunto de prácticas que son esenciales para el éxito de los proyectos y no deben pasarse por alto en el proceso de desarrollo, estas son (Joskowicz, 2008):

- **Recodificación:** consiste en escribir nuevamente parte del código de un programa, sin cambiar su funcionalidad, a los efectos de hacerlo más simple, conciso y/o entendible.
- **Pruebas unitarias:** las pruebas unitarias son una de las piedras angulares de XP. Todos los módulos deben pasar las pruebas unitarias antes de ser liberados o publicados.
- **Pruebas de aceptación:** las pruebas de aceptación son creadas en base a las HU, en cada ciclo de la iteración del desarrollo. El cliente debe especificar uno o diversos escenarios para comprobar que una historia de usuario ha sido correctamente implementada. Las pruebas de aceptación son consideradas como “pruebas de caja negra”. Los clientes son responsables de verificar que los resultados de éstas pruebas sean correctos.

- **Reuniones diarias de seguimiento:** el objetivo de tener reuniones diarias es mantener la comunicación entre el equipo, y compartir problemas y soluciones.

1.5. Tecnologías y herramientas

A continuación, se describen las principales herramientas y tecnologías seleccionadas para ser utilizadas en el proceso de desarrollo del software, las cuales son de software libre.

1.5.1. Lenguajes de programación

Es lenguaje de programación es aquella estructura que, con una cierta base sintáctica y semántica, imparte distintas instrucciones a un programa de computadora (Definición de, 2017).

- **Java**

El lenguaje de programación Java es robusto, multiplataforma. Tiene muchas similitudes con el lenguaje C y C++. La principal característica de Java es la de ser un lenguaje compilado e interpretado. Todo programa en Java ha de compilarse y el código que se genera es interpretado por una máquina virtual (Lenguajes de programación, 2017). Permitirá la creación e implementación de las clases controladoras.

- **JavaFX**

JavaFX es un conjunto de paquetes de gráficos y multimedia que permite a los desarrolladores diseñar, crear, probar, depurar y desplegar aplicaciones de cliente, que operan constantemente a través de diversas plataformas (Oracle, 2017). Facilita notablemente la construcción de las interfaces de aplicaciones que serán construidas con la herramienta Scene Builder.

Scene Builder

Es una herramienta de diseño visual que permite a los usuarios diseñar rápidamente interfaces de usuario de las aplicaciones JavaFX, sin necesidad de programación. Los usuarios pueden arrastrar y soltar los componentes de interfaz de usuario a un área de trabajo y modificar sus propiedades. Se aplican las hojas de estilo (ccs), y el código FXML se genera automáticamente en segundo plano según los elementos que se vayan creando. El resultado es un archivo FXML que a continuación se puede combinar con un proyecto de Java mediante la unión de la interfaz de usuario a la lógica de la aplicación (Oracle, 2017).

IDE: NetBeans 8.0.2

NetBeans (NetBeans, 2017) es un entorno de desarrollo integrado (IDE), modular, de base estándar (normalizado), escrito en el lenguaje de programación Java. El proyecto NetBeans consiste en un IDE de código abierto y una plataforma de aplicación, las cuales pueden ser usadas como una estructura de soporte general (framework) para compilar cualquier tipo de aplicación. Algunas de las características son:

- Integración MySQL más ajustada y una mejor manera de compartir librerías entre proyectos dependientes.
- El aclamado soporte para Ruby/JRuby ha sido mejorado con un nuevo editor de soluciones rápidas (Quick Fix).
- Un administrador para la plataforma Ruby,
- Soporte para depuración rápida en JRuby.
- Registro de servidores MySQL.

JUnit

jUnit es un framework java que permite la realización de la ejecución de clases de manera controlada, para poder comprobar que los métodos realizan su cometido de forma correcta (La paradita, 2017). Contiene un conjunto de librerías que facilitan la realización de dichas pruebas, necesarias para verificar que los métodos implementados se desempeñen de forma correcta.

1.5.2. Herramienta CASE: Visual Paradigm 8.0

Visual Paradigm es una herramienta CASE que propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Es una herramienta con disponibilidad en múltiples plataformas y soporta varios Sistemas Gestores de Bases de Datos como Oracle, Microsoft SQL Server, MySQL PostgreSQL y SQLite. También permite el uso de las distintas metodologías propias de la Ingeniería de Software (Pressman, 2007).

Lenguaje de modelado: UML 2.0

UML son las siglas de “lenguaje unificado de modelado”. Se trata de un estándar que se ha adoptado a nivel internacional por numerosos organismos y empresas para crear esquemas, diagramas y documentación relativa a los desarrollos de software (programas informáticos) (Krall, 2017).

1.5.3. Sistema gestor de base de datos

Un SGBD es un conjunto de programas que permiten la creación de base de datos y proporciona herramientas para añadir, borrar, modificar, y eliminar datos, además de mantener la integridad de estos.

PostgreSQL 9.3

PostgreSQL es un potente SGBD objeto relacional de código abierto, tiene soporte completo para claves foráneas, vistas y disparadores. Incluye la mayoría de los tipos de datos de SQL 2008 incluyendo integer, boolean y varchar entre otros. También soporta el almacenamiento de objetos binarios grandes como imágenes, sonido o video. Algunas de las características de PostgreSQL son (Microbuffer, 2011):

- Arquitectura cliente servidor con un amplio rango de drivers.
- Diseño de alta concurrencia donde lectores y escritores no se bloquean.
- Altamente confiable y extensible para muchos tipos de aplicación.
- Optimizador de consultas sofisticado.
- Incluye herencia entre tablas.

PgAdmin III

PaAdmin III es una herramienta de código abierto para la administración de bases de datos PostgreSQL y derivados (EnterpriseDB, PostgresPlus, Advanced Server y GreenplumDatabase. Responde a las necesidades de la mayoría de los usuarios, desde escribir simples consultas SQL hasta desarrollar bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y hace simple la administración. Está disponible en más de una docena de lenguajes y para varios sistemas operativos, incluyendo Microsoft Windows, Linux, FreeBSD, Mac OSX y Solaris. Soporta versiones de servidores 7.3 y superiores. Versiones anteriores a 7.3 deben usar PgAdmin II (Michaud, 2015).

Conclusiones del capítulo

En este capítulo se realizó un estudio del estado del arte de los sistemas de gestión de información para realizar diagnósticos a pacientes. El estudio arrojó como resultado que los sistemas existentes no satisfacen las necesidades del problema planteado, por lo que se concluye que la solución adecuada es el desarrollo de un software para la gestión de información de los casos de intoxicaciones por plaguicidas, que permita disminuir el tiempo de respuesta en casos de intoxicaciones por plaguicidas para la prevención y control de eventos.

Además, se propuso las herramientas informáticas y tecnologías que se utilizarán en el desarrollo del software para la gestión de información de los casos de intoxicaciones por plaguicidas en el Cenatox, seleccionándose como metodología de desarrollo de software Extreme Programming, la cual guiará todo el proceso de creación del software propuesto, utilizando para el proceso de modelado se eligió UML 2.0 y Visual Paradigm 8.0 como herramienta CASE. El lenguaje de programación a utilizar Java, el conjunto de paquetes JavaFX para el diseño, apoyándose de la herramienta visual Scene Builder. Para facilitar el desarrollo se utilizarán NetBeans 8.0.2 como IDE y para el tratamiento de la base de datos PostgreSQL 9.3 y PgAdmin III.

Capítulo 2. Características del sistema de gestión y análisis de casos de intoxicación por plaguicida

La propuesta de solución que se caracteriza en este capítulo consta de un sistema de gestión que se apoya en una herramienta de análisis con razonamiento basado en casos. Este sistema se limita a tratar la información referente a las intoxicaciones por plaguicidas.

2.1. Modelo de negocio

El modelo de negocio se estructura en procesos de negocio por estar estos bien identificados en el Cenatox. Los procesos son:

- la gestión de plaguicidas
- la gestión de pacientes.

El proceso de gestión de plaguicidas inicia una vez que el especialista obtiene la información de que se añaden nuevas sustancias que se modifican o que son eliminadas del mercado. En los casos de modificación y eliminación retiran la HDS anterior y se reemplaza en caso necesario. Si es una adición, el especialista agrega al registro del Centro, la HDS proporcionada por el proveedor del nuevo plaguicida. Los datos son almacenados en hojas de papel que se guardan posteriormente en los archivos destinados al almacenamiento de las HDS.

El proceso gestión de pacientes comienza cuando un solicitante llama por teléfono al Cenatox para reportar un caso de intoxicación. El especialista del Cenatox registra de manera manual los datos generales del paciente y del solicitante, los datos del agente causal, la vía de exposición, los datos de la conducta y evolución del paciente, la identificación de los peligros las propiedades físicas y químicas de las sustancias. Una vez registrados los datos se hace una búsqueda exhaustiva en las HDS de plaguicidas, identificando las coincidencias entre las sesiones de identificación de los peligros, propiedades físicas y químicas, y otras informaciones que coincidan con los datos registrados producto de la llamada. Una vez que se identifican semejanzas entre los datos recopilados y las HDS de plaguicidas, los especialistas del centro, apoyándose en la sección de primeros auxilios que enuncia la HDS, valoran un posible diagnóstico.

Personas que intervienen en el negocio

Tabla 3: Personas que intervienen en el negocio.

Nombre	Descripción
Especialista del Cenatox.	Encargada de gestionar un plaguicida y diagnosticar a un paciente ante una intoxicación.
Solicitante	Persona que refiere los datos del nuevo caso de intoxicación

2.2 Análisis de la solución

Para dar solución a la problemática planteada se propone el desarrollo de una aplicación de escritorio que contará con tres módulos de gestión y un SBC que ofrece la posibilidad de trabajar mediante experiencias previas de especialistas y es capaz de confeccionar propuestas de solución para casos similares a los que contiene en su base de conocimientos. Su implementación proporciona al Cenatox una herramienta capaz de garantizar la recolección, gestión y control de toda la información de las HDS de plaguicidas y de los casos de intoxicaciones, ofreciendo la posibilidad mediante métodos java de filtrar con los datos recopilados, siendo capaz de confeccionar un posible diagnóstico.

El software será capaz de gestionar las HDS de plaguicidas y de los casos que contendrán información relacionada con la intoxicación del paciente, la información será procesada para dar un diagnóstico.

Descripción del software PlaguiTox

Para resolver la problemática propuesta se analizan las características que debe tener el software para que se cumpla con los objetivos planteado. Para ello se identifican las funcionalidades y requisitos no funcionales del software PlaguiTox.

Funcionalidades del software PlaguiTox

A continuación, se listan las funcionalidades que a partir de los procesos de negocio estudiados y las tareas realizadas en el Cenatox fueron identificadas:

- **Gestión de usuarios:** permite gestionar los usuarios que harán uso del sistema, especificando dos roles que pueden ser asignados (Administrador y Usuario). Posibilita insertar, modificar, listar, buscar, habilitar y deshabilitar los roles con los que trabaja el software.
- **Gestión de plaguicidas:** permite gestionar las sustancias con las que se trabaja en el Cenatox. Posibilita insertar, modificar, listar y buscar las sustancias que gestionan por los especialistas del centro.
- **Gestión de casos:** permite almacenar la información de los casos que presentan intoxicaciones por plaguicidas. posibilita insertar, eliminar, modificar, listar y buscar los datos personales y clínicos de las personas intoxicadas.
- **Consultar HDS:** permite consultar las HDS que contiene la gestión de plaguicidas y las resultantes de la búsqueda de las posibles causas de intoxicación. Posibilita visualizar la HDS del Plaguicida.
- **Exportar a PDF:** permite generar un documento en formato PDF que contiene los datos que se obtienen como resultado de haber realizado un diagnóstico a un paciente.

Requisitos no funcionales del software PlaguiTox

Las propiedades del producto en un software, constituyen las cualidades que debe tener para su correcto funcionamiento. Se tuvo en cuenta las propiedades del producto para la realización de la solución propuesta, atendiendo a los siguientes requerimientos definidos anteriormente con el cliente:

- **Usabilidad**

Los grupos de botones y vínculos deben organizarse por funcionalidad, con el objetivo de facilitar al usuario la interacción con el software.

Los mensajes para interactuar con los usuarios y los de error deben ser lo suficientemente informativos, en idioma español y no deben revelar información interna.

- **Software**

Características del software de los ordenadores donde se desee instalar:

- ✓ Sistema operativo: Windows 7 Ultimate 64 bits.

- **Hardware**

Características de hardware de los ordenadores donde se desee instalar:

- ✓ RAM: 2 GB.
- ✓ Procesador: Intel(R) Dual Core CPU @ 2.50 GHz.
- ✓ Espacio en disco disponible: 80 GB.

- **Interfaz**

El software debe tener una interfaz sencilla con una navegación intuitiva para el usuario.

- **Seguridad**

El sistema requiere la autenticación como primera acción, con un nombre de usuario único y una contraseña, que deben ser de conocimiento exclusivo de la persona que se autentica. Los permisos de acceso al sistema podrán ser cambiados solamente por el administrador de acceso a datos.

La información solo puede ser modificada por aquellos usuarios autorizados. La aplicación debe estar disponible en todo momento para las personas autorizadas que necesiten acceder y manejar la información. Todas estas características mencionadas anteriormente ponen en práctica los principios de la seguridad: confidencialidad, integridad y disponibilidad.

La institución debe garantizar la seguridad de los datos almacenados en la computadora donde se utiliza la aplicación, debido a que la base de datos con la que trabaja la aplicación estará en la propia computadora.

2.3 Planeación usando historias de usuario

El artefacto que genera XP para describir el funcionamiento y características de la aplicación son las HU. Estas son el equivalente a los casos de uso empleados en otras metodologías. Deben poder ser programadas en un tiempo entre una y tres semanas. Si la estimación es superior a tres semanas, debe ser dividida en dos o más historias. Si es menos de una semana, se debe combinar con otra historia (Letelier, 2006).

Las HU correspondientes al software fueron creadas especificando la prioridad de cada una de ellas. Están representadas mediante tablas con los siguientes elementos:

- **Número:** número de la HU, incremental en el tiempo.
- **Nombre:** nombre que identifica la HU.
- **Referencia:** es el conjunto de HU de las cuales depende la actual.
- **Prioridad:** esta característica es dada por el cliente con los valores: alta, media o baja en dependencia de la importancia y orden de la implementación.
 - ✓ Baja: se le otorga a las HU que constituyen funcionalidades que sirven de ayuda al control de elementos asociados al equipo de desarrollo, a la estructura, y no tienen que ver directamente con el sistema en desarrollo.
 - ✓ Media: se le otorga a las HU que resultan para el cliente como funcionalidades a tener en cuenta, sin que estas tengan una afectación sobre el sistema que se esté desarrollando.
 - ✓ Alta: se le otorga a las HU que resultan funcionalidades fundamentales en el desarrollo del sistema, las que el cliente define como principales para el control del sistema.
- **Iteración asignada:** número de la iteración en la cual se desarrollará la HU.
- **Puntos estimados:** tiempo estimado en semanas que se le asignará.
- **Descripción:** breve descripción del proceso que define la historia.
- **Observaciones:** alguna acotación importante a señalar sobre la historia.
- **Prototipo de interfaz:** prototipo de interfaz, si aplica.

Se definieron 5 HU, de ellas 2 se clasificaron con prioridad alta y 3 con prioridad media. A continuación, se representan en las tablas 3 y tabla 4 las HU clasificadas con prioridad alta. (Ver anexo 3)

Tabla 4: Descripción de la HU 2 Gestión de Plaguicidas.

Historia de Usuario	
Número: 2	Nombre de HU: Gestión de Plaguicidas
Referencia: ninguna	Prioridad: alta
Iteración Asignada: 1	Puntos Estimados: 2.5
Descripción: el software debe permitir insertar, modificar, listar y buscar las sustancias empleadas por los especialistas del centro.	
Observaciones: los plaguicidas son almacenados indicando: Nombre; Clasificación, Ruta donde se encuentra ubicada la hoja de seguridad, Propiedades físicas (Estado físico, Color y Olor), y la identificación de peligros.	

Prototipo de interfaz:

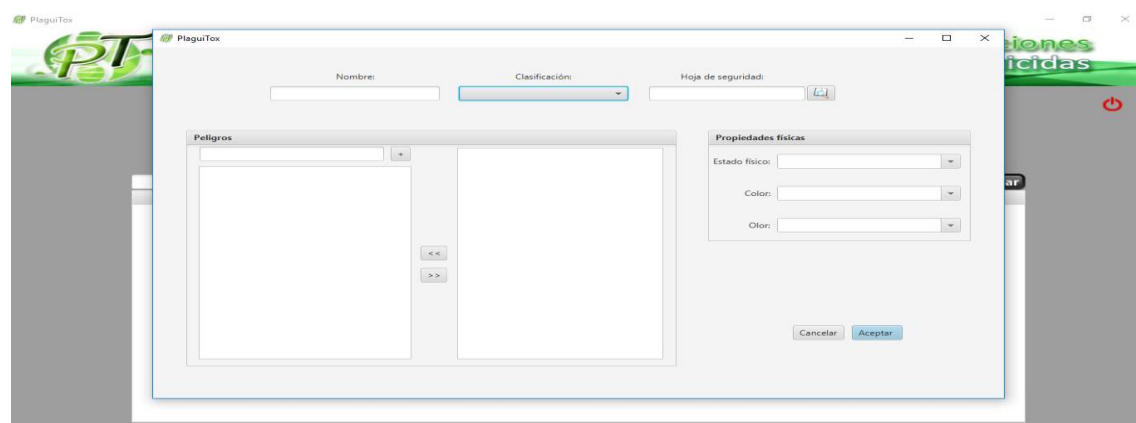
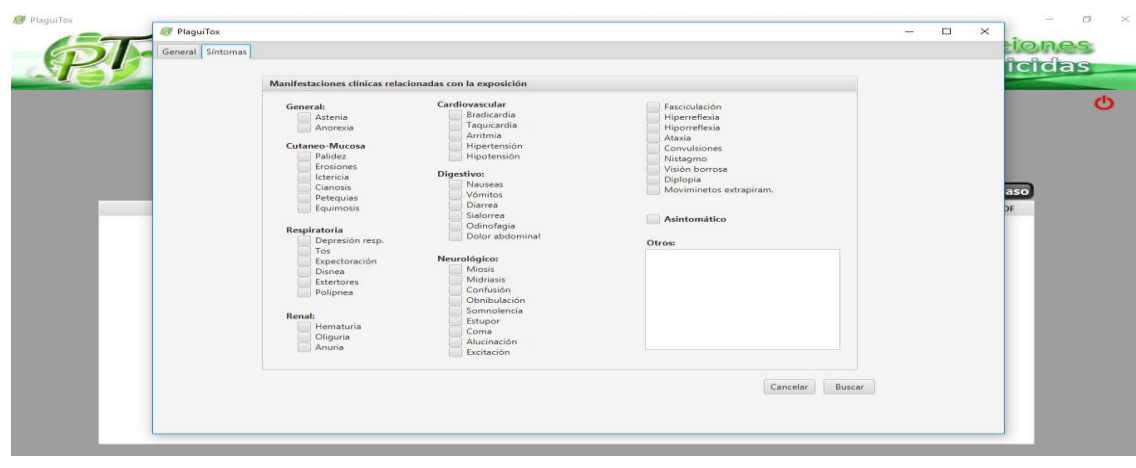


Tabla 5: Descripción de la HU 3 Gestión de Casos.

Historia de Usuario	
Número: 3	Nombre de HU: Gestión de Casos
Referencia: ninguna	Prioridad: alta
Iteración Asignada: 2	Puntos Estimados: 3
Descripción: el software debe permitir insertar, eliminar, modificar, listar y buscar los datos personales y clínicos de las personas intoxicadas.	
Observaciones: se completan los campos correspondientes a la ventana general y síntomas: Nombre; Clasificación, Propiedades físicas (Estado físico, Color y Olor), la identificación de peligros y manifestaciones clínicas correspondientes a la exposición. No es necesario completar todos los campos, solo la información que proporcione el solicitante. Una vez completado los campos se presiona el botón buscar apareciendo los resultados de la búsqueda del plaguicida por dos criterios: basado en casos similares y basado en las características del producto.	

Prototipo de interfaz:



Estimación del esfuerzo por HU

Las estimaciones de esfuerzo asociado a la implementación de las historias la establecen los programadores utilizando como medida el punto (Letelier, 2006), lo que se conoce como estimación de esfuerzo por puntos. Para ello los programadores definen una puntuación para cada HU de acuerdo a la complejidad de implementación de la misma para ellos, teniendo en cuenta una medida de tiempo por cada punto de estimación. A continuación, se muestra la estimación de esfuerzo por cada HU:

Tabla 6: Estimación del esfuerzo por HU.

No.	Nombre de historia de usuario	Puntos de estimación (semanas)
1	Gestión de usuarios	2.5
2	Gestión de plaguicidas	2.5
3	Gestión de casos	3
4	Consultar hoja de seguridad	2
5	Exportar a PDF	2

Al realizar la estimación de esfuerzo por puntos, el tiempo de desarrollo obtenido fue de 12 semanas. Los tiempos estimados de las HU variaron de 2 a 3 semanas.

Plan de iteraciones

Luego de estimar el tiempo de desarrollo de las HU, como parte de la metodología XP, se procede a realizar el plan de iteraciones para así implementar por un orden lógico cada una de las funcionalidades propuestas. A continuación, el plan de iteraciones:

- **Iteración 1:** en esta iteración se confecciona la arquitectura del software, se definen las clases del modelo, se realiza la conexión a la base de datos, y se diseñan las interfaces. Se desarrollan las HU uno y dos correspondientes a la gestión de usuarios que utilizan el software y la gestión de plaguicidas. Como resultado se obtendrá la aplicación en un 40 % de su implementación.

- **Iteración 2:** se desarrollan las HU tres y cuatro que son las encargadas de la gestión de casos y consultar HDS. Como resultado se obtendrá la aplicación en un 85 % de su implementación.
- **Iteración 3:** se le da solución a la HU cinco que es la encargada de exportar a PDF. Al finalizar se desarrollan pruebas de aceptación y unitarias obteniendo un 100 % de la aplicación, lista para realizar la entrega final del software.

Plan de duración de las entregas

El cálculo del tiempo se realizó teniendo en cuenta que una semana de trabajo consta de 5 días laborales que constan de 8 horas dedicadas en cada uno. La siguiente tabla muestra el plan de iteraciones con el tiempo empleado en cada una de ellas y las HU que las componen.

Tabla 7: Plan de Iteraciones.

Iteración	Orden de la HU a implementar	Duración total (semanas)
1	Gestión de usuarios	2.5
2	Gestión de plaguicidas	2.5
3	Gestión de casos	3
4	Consultar hoja de seguridad	2
5	Exportar a PDF	2

Plan de entrega

El plan de entregas es, que una definición de cada una de las entregas de la solución que se le entregará al cliente con su respectiva fecha, acordado previamente con este, en reuniones iniciales del inicio del proyecto. Según el cálculo de tiempo de las iteraciones se confeccionó el siguiente plan de entregas, en el cual se tiene como fecha definitiva el 2 de mayo de 2018.

Tabla 8: Plan de Entrega.

Historias de Usuario	1ra Iteración 14 de marzo del 2018	2da Iteración 18 de abril del 2018	3ra Iteración 2 de mayo del 2018
<ul style="list-style-type: none"> • Gestión de usuarios • Gestión de plaguicidas 	40% del software		
<ul style="list-style-type: none"> • Gestión de casos • Consultar hoja de seguridad 		85% del software	
<ul style="list-style-type: none"> • Exportar a PDF 			100% del software

2.4 Patrones de diseño

“Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software.” En otras palabras, brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares (Microsoft, 2018); ayudan al éxito del proyecto, pues permiten la reutilización de código, garantizan la robustez y extensibilidad del software.

Son útiles pues en el desarrollo del software:

- Ahorran tiempo
- Te ayudan a estar seguro de la validez de tu código
- Establecen un lenguaje común

Patrones (GRASP) utilizados

Lo esencial de un diseño de objetos lo constituye el diseño de las interacciones de objetos y la asignación de responsabilidades. Las decisiones que se tomen pueden influir profundamente en la extensibilidad, claridad y mantenimiento del sistema de software de objetos, además en el grado y calidad de los componentes reutilizables, por esta razón, durante el diseño se deben realizar los casos de usos con objetos basado en los patrones GRASP (Giraldo, et al., 2011).

Los patrones GRASP Codifican buenos principios y sugerencias relacionados frecuentemente con la asignación de responsabilidades. Los patrones utilizados en el desarrollo de la investigación son (Giraldo, et al., 2011) :

- **Patrón controlador:** asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. Esto facilita la centralización de actividades (validaciones, seguridad, etc.). El controlador no realiza estas actividades, las delega en otras clases con las que mantiene un modelo de alta cohesión.

```

@FXML
private void btnAceptar_onClick() throws SQLException {
    Paciente pac = dat.BuscarPaciente(txtNombre_paciente.getText(), txtApellidos_paciente.getText());
    if (pac == null) {
        pac = new Paciente(txtNombre_paciente.getText(), txtApellidos_paciente.getText(), Integer.parseInt(txtEdad.getText()),
            Integer.parseInt(cboOcupacion.getValue().toString()), cboT_edad.getValue().toString(), cboSexo.getValue().toString());
        pac.setId_Paciente(dat.InsertarPaciente(pac));
    }
    Solicitante solic = dat.BuscarSolicitante(txtNombre_solicitante.getText(), txtApellidos_solicitante.getText());
    if (solic == null) {
        solic = new Solicitante(txtNombre_solicitante.getText(), txtApellidos_solicitante.getText(), cboProvincia.getValue().toString(),
            cboProcedencia.getValue().getId_Procedencia(), cboT_solicitante.getValue().getId_Tipo_Solicitante(), txtTelefono.getText());
        solic.setId_Solicitante(dat.InsertarSolicitante(solic));
    }
    Usuario userActual = dat.getUsuario();
    RadioButton radEvol = (RadioButton) togEvolucion.getSelectedToggle();
    int id_evolucion = dat.BuscarEvolucion(radEvol.getText()).getId_Evolucion();
    int idcaso = 0;
    Caso caso = new Caso(fecha.getEditor().getText(), pac.getId_Paciente(), id_evolucion, solic.getId_Solicitante(),
        chkintoxicado.isSelected(), userActual.getId_Usuario());
    dat.InsertarCaso(caso);
    Set<Node> checkBox = getStage().getScene().getRoot().lookupAll(".checkbox");
    for (Node node : checkBox) {
        CheckBox actual = (CheckBox) node;
        String textTitlePaint = ((TitledPane) (node.getParent().getParent().getParent().getParent())).getText();
        if (textTitlePaint.equals(titleViaExposicion.getText())) {
            ViaExposicion via = dat.BuscarViaExposicion(actual.getText());
            dat.InsertarViaExposicion_Caso(idcaso, via.getId_Via());
        } else if (textTitlePaint.equals(titleManifestaciones.getText())) {
            Manifestacion mani = dat.BuscarManifestacion(actual.getText());
            dat.InsertarManifestacion_Caso(idcaso, mani.getId_Manifestacion());
        } else if (textTitlePaint.equals(titleAgenteCausal.getText())) {
            AgenteCausal agente = dat.BuscarAgenteCausal(actual.getText());
            dat.InsertarAgenteCausal_Caso(idcaso, agente.getId_Agente_Causal());
        }
    }
}

```

Figura 8: Ejemplo de patrón controlador en la implementación de la herramienta. Fuente: elaboración propia.

- **Patrón creador:** el patrón creador guía la asignación de responsabilidades relacionadas con la creación de objetos. Se emplea al momento de definir la creación de objetos, se garantiza que si una clase puede crear objetos de otra es porque guardan determinada relación, ya sea que una contiene a la otra, o que una contiene los datos que necesita la segunda.

```

public static IInterfaces cargarInterfaz(String nombreI, Datos dat, Stage Sactual, Object datos, boolean cerrar) throws IOException {

    FXMLLoader loader = new FXMLLoader(Tesis2.class.getResource("/Interfaces/" + nombreI + ".fxml"));
    AnchorPane principal = (AnchorPane) loader.load();
    Stage ventana = new Stage();
    ventana.setTitle("Plaguisoft");

    Scene scene = new Scene(principal);
    ventana.setScene(scene);
    IInterfaces controller = loader.getController();
    controller.setStage(ventana);
    controller.setDat(dat);

    if (datos != null) {
        controller.setDatosM(datos);
    }
    if (!nombreI.equals(NIAutenticar) && Sactual != null) {

        controller.getStage().setMaximized(true);
        ventana.setOnCloseRequest(new EventHandler<WindowEvent>() {
            @Override
            public void handle(WindowEvent we) {
                try {
                    InterfazUtil.cargarInterfaz("Autenticar", dat, Sactual, null, cerrar);
                } catch (IOException ex) {
                    Logger.getLogger(PrincipalController.class.getName()).log(Level.SEVERE, null, ex);
                }
            }
        });
    }
    ventana.show();
    if (cerrar) {
        Sactual.close();
    }
    return controller;
}

```

Figura 9: Ejemplo de patrón creador en la implementación de la herramienta. Fuente: elaboración propia.

- **Patrón alta cohesión:** cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable, lo que facilita la organización del proyecto y que estas no estén sobrecargadas de funcionalidades ajenas.

```

public AgenteCausal BuscarAgenteCausal(String Nombre) throws SQLException {
    try {
        con = DriverManager.getConnection(url, user, this.pass);
    } catch (SQLException ex) {
        Logger.getLogger(Datos.class.getName()).log(Level.SEVERE, null, ex);
    }

    try {
        String consulta = "SELECT * FROM \"Agente_Causal\" WHERE \"Nombre_Agente_Causal\" = ? ";
        st = con.prepareStatement(consulta);
        st.setString(1, Nombre);

        rs = st.executeQuery();
    } catch (SQLException ex) {
        Logger.getLogger(Datos.class.getName()).log(Level.SEVERE, null, ex);
    } finally {
        con.close();
    }

    AgenteCausal agente;
    if (rs.next()) {
        return agente = new AgenteCausal(rs.getInt(1), rs.getString(2));
    }
    return null;
}

```

Figura 10: Ejemplo de patrón alta cohesión en la implementación de la herramienta. Fuente: elaboración propia.

- **Patrón experto:** la responsabilidad de realizar una labor es de la clase que tiene o puede tener

los datos involucrados (atributos). Una clase, contiene toda la información necesaria para realizar la labor que tiene encomendada. El comportamiento se distribuye entre las clases que cuentan con la información requerida, alentando con ello definiciones de clase “sencillas” y más cohesivas que son más fáciles de comprender y de mantener.

```
@FXML
public void Btn_Aceptar_Click() throws SQLException, IOException {
    if (dat.login(usuario.getText(), pass.getText())) {
        InterfazUtil.cargarInterfaz("Principal", dat, stage, null, true);
    } else {
        m_mostrarMensaje_Desvanecer("Usuario o Contraseña incorrectos");
    }
}
```

Figura 11: Ejemplo de patrón experto en la implementación de la herramienta. Fuente: elaboración propia.

2.5 Diseño del software PlaguiTox

El diseño XP sigue rigurosamente el principio mantenlo sencillo (MS). Un diseño sencillo siempre se prefiere sobre una representación más compleja. Además, el diseño guía la implementación de una historia conforme se escribe: nada más y nada menos (Pressman, 2007).

En esta fase se confeccionan las tarjetas CRC para guiar el proceso de implementación de la solución, además se describen los patrones de diseño utilizados. El diseño aporta una representación del software mediante el establecimiento de la arquitectura y el modelo de datos.

Arquitectura del software PlaguiTox

El patrón modelo-vista-controlador (MVC) surge con el objetivo de reducir el esfuerzo de programación, necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos, a partir de estandarizar el diseño de las aplicaciones. El patrón MVC es un paradigma que divide las partes que conforman una aplicación en el Modelo, las Vistas y los Controladores, permitiendo la implementación por separado de cada elemento, garantizando así la actualización y mantenimiento del software de forma sencilla y en un reducido espacio de tiempo. A partir del uso de frameworks basados en el patrón MVC se puede lograr una mejor organización del trabajo y mayor especialización de los desarrolladores y diseñadores (González, et al., 2012).

Partes del MVC (González, et al., 2012):

- **El modelo:** contiene una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia.
- **La vista:** o interfaz de usuario, que compone la información que se envía al cliente y los mecanismos de interacción con éste.

- **El controlador:** actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno.

Este modelo de arquitectura presenta varias ventajas (González, et al., 2012):

- Separación clara entre los componentes de un programa; lo cual permite su implementación por separado.
- Interfaz de Programación de Aplicaciones API (Application Programming Interface) muy bien definida; cualquiera que use el API, podrá reemplazar el modelo, la vista o el controlador, sin aparente dificultad.
- Conexión entre el modelo y sus vistas dinámica; se produce en tiempo de ejecución, no en tiempo de compilación.

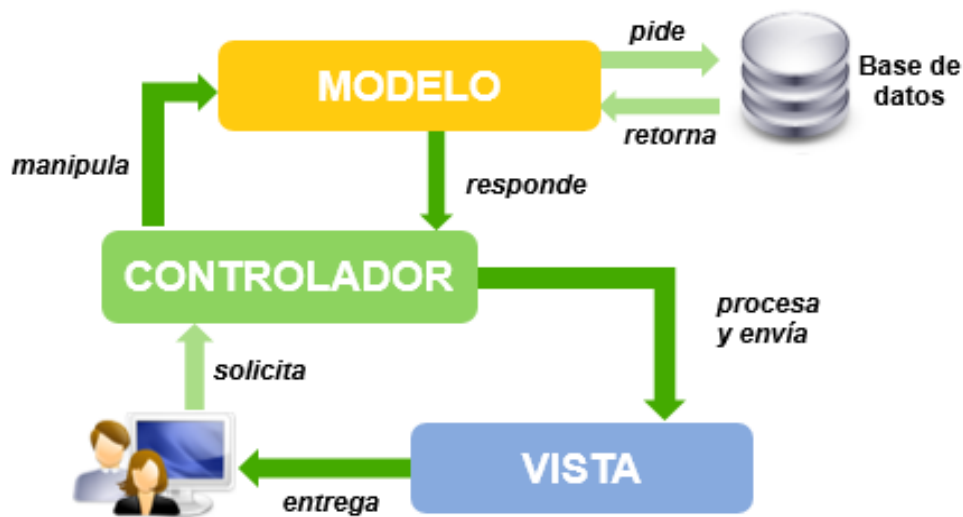


Figura 12: Arquitectura MVC. Fuente: elaboración propia.

Para el desarrollo de la propuesta de solución se utiliza el patrón arquitectónico MVC debido a que este separa la lógica de negocio de la interfaz de usuario en tres capas diferentes, cada una con funcionalidades bien definidas, reduciendo esfuerzo en la implementación de la aplicación y garantizando una mejor organización del trabajo.

En la figura 13 se puede apreciar cómo queda este patrón arquitectónico aplicado a la implementación de la presente investigación.

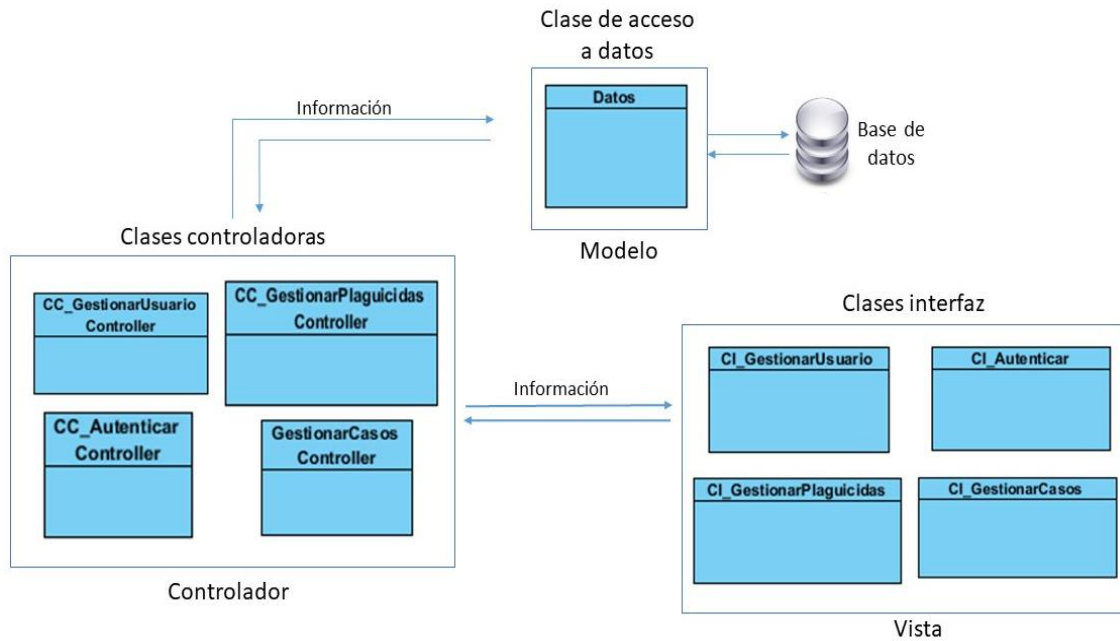


Figura 13: Patrón arquitectónico aplicado a la investigación. Fuente: elaboración propia.

Modelo de datos

Para darle solución a la problemática planteada se obtuvo un modelo relacional de datos, que se desarrolló en la herramienta CASE Visual Paradigm 8.0 para el diseño y análisis lógico de los datos. A continuación, se representa en la figura 15 donde se aprecia la relación de datos:

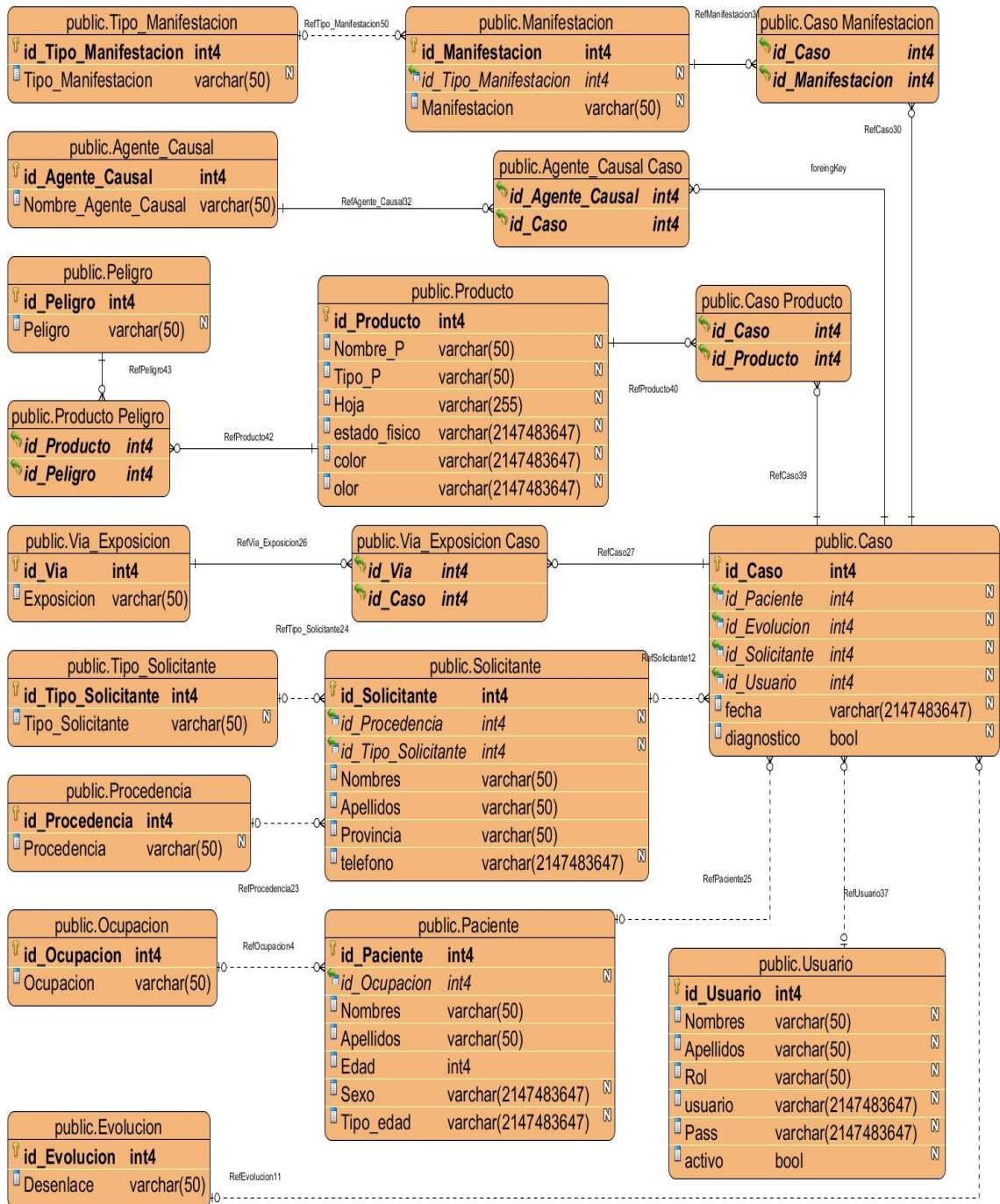


Figura 14: Modelo de datos relacional para el software PlaguTox. Fuente: elaboración propia.

El diagrama representa el modelo de datos de la investigación, donde se expone la información que va a persistir en un total de 19 tablas (peligro, producto, producto peligro, caso, caso producto, solicitante, tipo solicitante, procedencia, evolución, paciente, ocupación, vía exposición, vía exposición caso, usuario, agente causal diagnóstico, agente causal, caso manifestación, manifestación, tipo manifestación) y la relación que existe entre ellas, garantizando una estructura jerárquica; para la representación de los casos de intoxicaciones y mantener la información referente a cada hoja de seguridad.

Tarjetas CRC

Las tarjetas CRC proporcionan una manera sencilla de identificación y organización de las clases que son relevantes para los requerimientos de un sistema o producto. Se confeccionan con el objetivo de desarrollar una representación organizada de las clases (Pressman, 2007).

Las tarjetas se dividen en tres sesiones:

- **Clase:** nombre de la clase que se corresponde con la tarjeta.
- **Responsabilidad:** describe las funcionalidades que contiene clase.
- **Colaboración:** enuncia las clases que guardan relación con la que se describe en la tarjeta.

A continuación, aparecen en las tablas 8, tabla 9 y tabla 10 las tarjetas CRC que corresponden a cada clase del sistema. (Ver anexo 4)

Tabla 9: Tarjeta CRC clase Datos.

Tarjetas CRC	
Clase: Datos	
Responsabilidad	Colaborador
Es la clase encargada de: <ul style="list-style-type: none"> • Realizar la conexión a la base de datos. • Cerrar la conexión con la base de datos. • Hacer las consultas necesarias a la base de datos. 	Todas las clases por las que está conformada la aplicación.

Tabla 10: Tarjeta CRC clase PrincipalController.

Tarjetas CRC	
Clase: PrincipalController	
Responsabilidad	Colaborador
Es la clase encargada de: <ul style="list-style-type: none"> • Insertar casos • Listar casos • Exportar caso a PDF 	Generate_PDFFileiText Datos PrincipalController Caso AgenteCausal Evolucion Manifestacion Ocupacion

	Paciente Peligro Procedencia Producto Solicitante Tipo_Solicitante Usuario ViaExposicion
--	---

Tabla 11: Tarjeta CRC clase Recuperar_Casos.

Tarjetas CRC	
Clase: Recuperar_Casos	
Responsabilidad	Colaborador
Es la clase encargada de: <ul style="list-style-type: none"> Recupera los casos similares en la base de datos. Lista los productos relacionados con casos similares 	Datos Caso Manifestacion PrincipalController Producto

2.6 Diseño de la base de casos

Los casos son una de las formas en que puede ser representado el conocimiento para su posterior utilización. Los SBC aplican el razonamiento basados en casos (RBC) que es: un enfoque que aborda nuevos problemas tomando como referencia problemas similares resueltos en el pasado. De modo que problemas similares tienen soluciones similares, y la similitud juega un rol esencial (A, 2012). Para construir la base de casos es necesario inicialmente identificar los atributos que describirán cada uno de los casos. Estos se dividen en rasgos predictores y rasgos objetivo. Los primeros corresponden a los elementos que el especialista utiliza para caracterizar el estado del paciente y los segundos los plaguicidas que se asemejan a los rasgos predictores utilizados a los cuales se le asigna una hoja de seguridad la cual contiene el tratamiento de desintoxicación. De cada uno se debe especificar el dominio de definición y el tipo de dato correspondiente.

En el sistema que se implementa, cada caso está compuesto por un número fijo de rasgos que se describen a continuación en la tabla 12:

Tabla 12: Rasgos predictores de la base de casos del SBC.

Rasgos predictores	Dominio	Tipo de datos
General		

Astenia	Si, No	Binario
Anorexia	Si, No	Binario
Cutáneo-Mucosa		
Palidez	Si, No	Binario
Erosiones	Si, No	Binario
Ictericia	Si, No	Binario
Cianosis	Si, No	Binario
Petequias	Si, No	Binario
Equimosis	Si, No	Binario
Respiratorias		
Depresión respiratoria	Si, No	Binario
Tos	Si, No	Binario
Expectoración	Si, No	Binario
Disnea	Si, No	Binario
Estertores	Si, No	Binario
Polipnea	Si, No	Binario
Renal		
Hematuria	Si, No	Binario
Oliguria	Si, No	Binario
Anuria	Si, No	Binario
Cardiovascular		
Bradycardia	Si, No	Binario
Taquicardia	Si, No	Binario
Arritmia	Si, No	Binario
Hipertensión	Si, No	Binario
Hipotensión	Si, No	Binario
Digestivo		
Nauseas	Si, No	Binario
Vómitos	Si, No	Binario
Diarrea	Si, No	Binario
Sialorrea	Si, No	Binario
Odinofagia	Si, No	Binario
Dolor abdominal	Si, No	Binario
Neurológico		
Miosis	Si, No	Binario

Midriasis	Si, No	Binario
Confusión	Si, No	Binario
Obnubilación	Si, No	Binario
Somnolencia	Si, No	Binario
Estupor	Si, No	Binario
Coma	Si, No	Binario
Alucinación	Si, No	Binario
Excitación	Si, No	Binario
Fasciculación	Si, No	Binario
Hiperreflexia	Si, No	Binario
Hiporreflexia	Si, No	Binario
Ataxia	Si, No	Binario
Convulsiones	Si, No	Binario
Nistagmo	Si, No	Binario
Visión borrosa	Si, No	Binario
Diplopía	Si, No	Binario
Movimiento extrapiramidal	Si, No	Binario

En correspondencia con los rasgos predictores identificados los especialistas del Cenatox seleccionan el plaguicida que más se asemeja, para indicar el tratamiento que permite la desintoxicación del paciente. La tabla 13 muestra un conjunto de plaguicidas de un total de 865.

Tabla 13: Rasgos objetivo de la base de casos del SBC.

Rasgos objetivos	Posibles valores
Abamectina	Si, No
Temofos-silice	Si, No
Acetoclor	Si, No
Imazapic	Si, No
Dazomet	Si, No

Conclusiones del capítulo

La metodología XP permitió realizar el análisis y diseño de la aplicación, generando los elementos fundamentales para el desarrollo de la herramienta PlaguiTox.

Se definieron un total de 5 HU que describen los aspectos principales a tener en cuenta para el desarrollo de la solución. Se precisó la prioridad de cada HU puntualizando el orden de su implementación y las iteraciones en que fueron implementadas.

Se construyó el plan de entregas y el plan de iteraciones para una mejor organización en el trabajo de los desarrolladores. Además, se describió la arquitectura que va a tener el software, así como las tarjetas CRC.

Los patrones de diseño utilizados propiciaron una organización del código fuente que permite un entendimiento, claridad y reutilización de los diferentes algoritmos desarrollados.

Capítulo 3. Implementación y validación del sistema

A partir del diseño realizado en el capítulo 2, se llega a la codificación del sistema. Para realizar la validación del software con el objetivo de comprobar que funciona según lo diseñado y que las funcionalidades se han implementado de forma adecuada.

3.1 Proceso de extracción del conocimiento

El proceso de extracción del conocimiento en los SBC está compuesto por cuatro etapas que deben ser ejecutadas para obtener una solución factible al problema en cuestión, estas etapas son: recuperación, reutilización, adaptación y almacenamiento o aprendizaje. A continuación, se explican los algoritmos y estrategias empleadas en cada una de estas etapas (Martí Pérez, 2012):

Etapa de recuperación

La primera etapa en el proceso de extracción del conocimiento generalmente es antecedida por labores de pre-procesamiento de datos que garantizan completitud y organización. Esto se garantiza en el proceso de creación de la base de conocimientos pues las variables con que se trabaja son de tipo booleanas, su dominio está definido y se valida que no se almacenen valores incorrectos o nulos en la aplicación.

Esta etapa es la encargada de recuperar instancias o casos con características similares al caso analizado, siguiendo una estrategia de umbral para comparar parámetros de los rasgos predictores. Luego de un análisis con los médicos del Cenatox se determinó que el valor mínimo (umbral) para determinar que 2 casos son semejantes es de 0.75. Para ello se emplean funciones de semejanza entre los atributos del caso nuevo con los casos almacenados en la base de datos.

Para cada uno de los atributos del caso nuevo se verifica que este exista en los casos anteriores, basado en la fórmula de la Distancia Hamming debido a que todos los atributos se tratan como si fueran booleanos, para ellos se emplea la siguiente fórmula:

$$(x, y) = \begin{cases} 0 & \text{si } x \neq y \\ 1 & \text{si } x = y \end{cases}$$

Donde (x,y) representan los valores del atributo analizado en los casos que estén siendo evaluados. Para calcular la similitud entre 2 casos se utiliza la Distancia Euclidiana donde se efectúa una suma de los resultados devueltos después de comparar los atributos, el valor de similitud calculado en este caso es el equivalente a esa distancia.

$$d(C1, C2) = \frac{\sum_{k=0}^n (s(x, y))}{n}$$

Luego de calcular la similitud existente entre cada uno de los casos se procede de la siguiente forma. Si la suma obtenida como resultado de la comparación entre atributos es cero, se devuelve ese valor, de

lo contrario se divide dicha suma entre la cantidad de atributos evaluados del caso de más atributos, esto último para garantizar que el resultado quede normalizado (valores entre 0 y 1), siendo 2 casos similares cuando la distancia o similitud sea mayor o igual que el umbral. La cantidad de atributos evaluados puede variar debido a que todos los casos no tienen la misma cantidad de rasgos.

Etapa de reutilización

Una vez que se han recuperado de la base de conocimientos los casos más semejantes al nuevo caso analizado, se procede a reutilizar ese conocimiento para obtener un posible agente causal (plaguicida) y de ahí obtener el tratamiento adecuado el paciente.

El proceso de reutilización se lleva a cabo aplicando la técnica de la moda para devolver la solución que más se repite de todas las que han sido encontradas. Además, por petición del cliente el usuario que interactúa con la aplicación (especialista del Cenatox) es el encargado de decidir cuál es la solución final, para así dar paso a la fase de Revisión.

Etapa de revisión o adaptación

La revisión consiste de dos tareas:

- Avalar cuidadosamente la solución generada por la reutilización. Si fuera considerada como correcta, aprende y continúa con la retención del nuevo caso en la base de casos.
- En caso contrario, reparar la solución utilizando conocimiento específico sobre el dominio de aplicación o información suministrada por el usuario.

Solo una minoría de los sistemas relacionados con el RBC implementa el proceso de revisión. Si la solución generada en las fases anteriores no es correcta, aun así, puede “aprender de los errores” y repararla. Es decir, la reparación de una solución es una adaptación llevada a cabo cuando se sabe que la solución no es válida. Hay que tener capacidad para evaluar la corrección de las soluciones, ya sea porque la solución se pueda probar de alguna forma, por ejemplo, aplicándola en el sistema real o en algún tipo de modelo, o porque un agente externo al sistema (el usuario) la señale como correcta o errónea.

Luego de analizar diferentes técnicas se decidió aplicar la adaptación nula.

Adaptación nula: es la técnica más simple, no es ni estructural ni derivacional. Consiste en no hacer nada y simplemente aplicar cualquiera que sea la solución recuperada, a la nueva situación. Esta es útil en tareas donde el razonamiento necesario para una aplicación puede ser muy complejo, y la solución en sí misma es muy simple.

Almacenamiento o aprendizaje

Para garantizar que en un futuro el sistema pueda solucionar problemas más complejos este debe ir aprendiendo. Esto se traduce en incorporar nuevos casos a su base de conocimientos para poder

reutilizarlos más adelante. Para ello el sistema guarda automáticamente cada cosa que es registrado en la aplicación.

3.2 Fase de codificación

Un concepto clave durante la actividad de codificación y uno de los aspectos del que más se habla en la XP es la programación por parejas. XP recomienda que dos personas trabajen juntas en una estación de trabajo con el objeto de crear código para una historia. Esto da un mecanismo para la solución de problemas en tiempo real (es frecuente que dos cabezas piensen más que una) y para el aseguramiento de la calidad también en tiempo real (el código se revisa conforme se crea) (Pressman, 2007).

Una vez que los algoritmos de una aplicación han sido diseñados, ya se puede iniciar la fase de codificación. En esta etapa se tienen que traducir dichos algoritmos a un lenguaje de programación específico; es decir, las acciones definidas en los algoritmos hay que convertirlas a instrucciones.

Tareas de Ingeniería

La metodología de software XP plantea que la implementación de un software se hace iterativamente. Durante cada iteración se desarrollan un conjunto de HU definidas por el cliente y descritas por el equipo de desarrollo. En esta fase de implementación las HU se dividen en tareas de ingeniería, las cuales son asignadas a los programadores para ser implementadas durante la iteración correspondiente (Pressman, 2007).

Las tareas de ingeniería serán representadas mediante tablas divididas por las siguientes secciones (Gutiérrez, 2006):

- **Número de tarea:** los números de cada tarea deben ser consecutivos.
- **Nombre de la tarea:** nombre que identifica a la tarea.
- **Número de historia de usuario:** número de la historia de usuario a la que pertenece la tarea.
- **Tipo de tarea:** las tareas pueden ser de Desarrollo, Corrección, Mejora.
- **Puntos estimados:** tiempo estimado en días que se le asignará a cada tarea.
- **Fecha inicio:** fecha en que comienza la tarea.
- **Fecha fin:** fecha en que se concluye la tarea.
- **Programadores responsables:** nombre y apellidos de los programadores que van a desarrollar la tarea.
- **Descripción:** descripción de la tarea.

A continuación, en las tablas 14, tabla 15 y tabla 16 se muestran algunas de las tareas correspondientes a las HU uno, dos y tres:

Tabla 14: Tarea de ingeniería Eliminar usuario.

Tareas de Ingeniería

Número de tarea: 5	Número de HU: 1
Nombre de tarea: Adicionar usuario	
Tipo de tarea: desarrollo.	Puntos estimados: 2/5
Fecha de Inicio: 1/marzo/2018	Fecha de fin: 2/marzo/2018
Programador responsable: Víctor Alexis Mesas Cisneros – Manuel Alejandro Miranda Hernández.	
Descripción: el software adiciona un usuario el cual empieza a interactuar con la herramienta.	

Tabla 15: Tarea de ingeniería Listar plaguicida.

Tareas de Ingeniería	
Número de tarea: 10	Número de HU: 2
Nombre de tarea: Listar Plaguicida	
Tipo de tarea: desarrollo.	Puntos estimados: 2/5
Fecha de Inicio: 13/marzo/2018	Fecha de fin: 14/marzo/2018
Programador responsable: Víctor Alexis Mesas Cisneros – Manuel Alejandro Miranda Hernández.	
Descripción: el software lista cada uno de los plaguicidas con los que cuenta el Cenatox.	

Tabla 16: Tarea de ingeniería Insertar caso.

Tareas de Ingeniería	
Número de tarea: 11	Número de HU: 3
Nombre de tarea: Insertar caso	
Tipo de tarea: desarrollo.	Puntos estimados: 3/5
Fecha de Inicio: 15/marzo/2018	Fecha de fin: 19/marzo/2018
Programador responsable: Víctor Alexis Mesas Cisneros – Manuel Alejandro Miranda Hernández.	

Descripción: el software permite insertar un nuevo caso completando los campos que aparecen dentro de las pestañas datos del paciente, manifestaciones clínica y evolución y diagnóstico.

3.3 Pruebas de software

Las pruebas intentan demostrar que un programa hace lo que se intenta que haga, así como descubrir defectos en el programa antes de usarlo. Al probar el software, se ejecuta un programa con datos artificiales. Hay que verificar los resultados de la prueba que se opera para buscar errores, anomalías o información de atributos no funcionales del programa. El proceso de prueba tiene dos metas distintas: Demostrar al desarrollador y al cliente que el software cumple con los requerimientos y encontrar situaciones donde el comportamiento del software sea incorrecto, indeseable o no esté de acuerdo con su especificación (Sommerville, 2005).

Existen diferentes estrategias de pruebas, pero el estudio de este trabajo se ha centrado en las pruebas de la metodología de desarrollo de software empleada en el presente trabajo de diploma.

3.3.1 Niveles de prueba

El proceso de realización de pruebas está compuesto por una serie de niveles entre los que se pueden encontrar: el nivel de pruebas unitarias, el nivel de pruebas de integración, el nivel de pruebas de sistema y nivel de pruebas de aceptación. Concluida la implementación el sistema fue sometido a los niveles de prueba que se detallan a continuación, los cuales propiciaron la detección de los errores existentes (Pressman, 2007).

Uno de los pilares de la XP es el proceso de pruebas. XP anima a probar constantemente tanto como sea posible. Esto permite aumentar la calidad de los sistemas reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. XP divide las pruebas del sistema en dos grupos: pruebas unitarias y pruebas de aceptación o pruebas funcionales (Gutiérrez, 2006).

Pruebas de aceptación

Las pruebas de aceptación XP, también llamadas pruebas del cliente, son especificadas por el cliente y se centran en las características y funcionalidad generales del sistema que son visibles y revisables por parte del cliente. Las pruebas de aceptación se derivan de las historias de los usuarios que se han implementado como parte de la liberación del software (Pressman, 2007).

Las pruebas de aceptación son creadas en base a las HU, en cada ciclo de la iteración del desarrollo y son la representación de la satisfacción del cliente con el producto que se le entrega y es este el principal encargado de verificar su cumplimiento. A continuación, en la tabla 17 se muestra un caso de prueba

correspondiente a la autenticación de un usuario y en la tabla 18 un caso de prueba correspondiente a la funcionalidad insertar plaguicida.

Tabla 17: Caso de prueba autenticar usuario.

Escenario	Descripción	Variable usuario	Variable Contraseña	Respuesta del sistema	Flujo central
Autenticar usuario.	Autenticar los usuarios de sistema.	V Manuel	V Manu0103	Accede a la interfaz principal del sistema.	Acceder a la aplicación. Llenar los campos de texto usuario y contraseña. Seleccionar el botón aceptar.
		V Víctor	I Victor3	Muestra la notificación, usuario o contraseña incorrectos.	
		I	V Manu0103	Muestra la notificación, complete los campos.	
		V Víctor	I	Muestra la notificación, complete los campos.	
		I	I	Muestra la notificación, complete los campos.	

Tabla 18: Caso de prueba insertar plaguicida.

Escenario	Descripción	Variable nombre	Variable clasificación	Variable hoja de seguridad	Variable peligros	Variable estado físico	Variable olor	Variable color	Respuesta del sistema	Flujo central
Insertar plaguicida.	Insertar un plaguicida en la base de datos del sistema, completando los campos definidos.	V Atrazina	V Herbicida	V C:\Users\ Manuel Alejandro\ Desktop\T esis\app PlaguiTox\ App PlaguiTox\ Hojas de seguridad\ Atrazina.d ocx	V Tóxico por ingestión	V Líquido	V Fuerte	V Rojo	Se agrega el nuevo producto a la base de datos.	Una vez autenticado el usuario accede al botón insertar que está en la interfaz gestión de plaguicid

		V	V	V	V	V	V	I	Verifique que los datos estén correctos.	as. Llenar los campos de texto nombre, peligros, estado físico y los restantes campos que aparecen en la interfaz.
		Atrazina	Herbicida	C:\Users\Manuel Alejandro\Desktop\Tesis\app PlaguiTox\App PlaguiTox\Hojas de seguridad\Atrazina.docx	Tóxico por ingestión	Líquido	Fuerte			
		V	V	V	V	V	I	V	Verifique que los datos estén correctos.	Seleccio na el botón aceptar.
		Atrazina	Herbicida	C:\Users\Manuel Alejandro\Desktop\Tesis\app PlaguiTox\App	Tóxico por ingestión	Líquido		Rojo		

				PlaguiTox\ Hojas de seguridad\ Atrazina.d ocx					
		V Atrazina	V Herbicida	V C:\Users\ Manuel Alejandro\ Desktop\T esis\app PlaguiTox\ App PlaguiTox\ Hojas de seguridad\ Atrazina.d ocx	V Tóxico por ingestión	I	V Fuerte	V Rojo	Verifique que los datos estén correctos.
		V Atrazina	V	V C:\Users\ Manuel	I	V Líquido	V Fuerte	V Rojo	Verifique que los

				Alejandro\Desktop\Tesis\app PlaguiTox\App PlaguiTox\Hojas de seguridad\Atrazina.docx	Tóxico por ingestión				datos estén correctos.
		V Atrazina	V Herbicida	I C:\Users\Manuel Alejandro\Desktop\Tesis\app PlaguiTox\App PlaguiTox\Hojas de seguridad\	V Tóxico por ingestión	V Líquido	V Fuerte	V Rojo	Verifique que los datos estén correctos.

				Atrazina.d ocx					
		V	I	I C:\Users\ Manuel Alejandro\ Desktop\T esis\app PlaguiTox\ App PlaguiTox\ Hojas de seguridad\ Atrazina.d ocx	V Tóxico por ingestión	V Líquido	V Fuerte	V Rojo	Verifique que los datos estén correctos.
		I	V	I C:\Users\ Manuel Alejandro\ Desktop\T esis\app	V Tóxico por ingestión	V Líquido	V Fuerte	V Rojo	Verifique que los datos estén correctos.

				PlaguiTox\ App PlaguiTox\ Hojas de seguridad\ Atrazina.d ocx						
										Verifique que los datos estén correctos.

Pruebas unitarias

Las pruebas unitarias son una de las piedras angulares de XP, se realizan para controlar el funcionamiento de pequeñas porciones de código. Generalmente son realizadas por el mismo programador, debido a que al conocer con mayor detalle el código, se le simplifica la tarea de elaborar conjuntos de datos de prueba para testearlo (Malforá, 2006).

Se recomienda que estas pruebas sean automatizadas, pues esto facilita al programador identificar funciones que no ofrecen una salida acorde con la lógica que se deseaba implementar. La herramienta JUnit permite correr un conjunto de pruebas de forma automática e informa de aquellas que han fallado. A continuación, se representa una de las funcionalidades del sistema a la cual se le realiza una prueba unitaria:

```

45     @Test
46     public void testGetId_Evolucion() {
47         System.out.println("getId_Evolucion");
48         Caso instance = new Caso(0, "fecha", 0, 4, 0, true, 0);
49         int expectedResult = 4;
50         int result = instance.getId_Evolucion();
51         assertEquals(expectedResult, result);
52         // TODO review the generated test code and remove the default call to fail.
53         //fail("The test case is a prototype.");
54     }

```

Figura 15: Caso de prueba del método GetId_Evolucion de la clase Caso. Fuente: elaboración propia.

```

43     @Test
44     public void testGetId_Paciente() {
45         System.out.println("getId_Paciente");
46         Paciente instance = new Paciente(2, "nombres", "apellidos", 0, 0, "id_Tipo_Edad", "id_Sexo");
47         int expectedResult = 2;
48         int result = instance.getId_Paciente();
49         assertEquals(expectedResult, result);
50         // TODO review the generated test code and remove the default call to fail.
51         //fail("The test case is a prototype.");
52     }

```

Figura 16: Caso de prueba del método GetId_Paciente de la clase Paciente. Fuente: elaboración propia.

```

128     @Test
129     public void testGetId_Usuario() {
130         System.out.println("getId_Usuario");
131         Caso instance = new Caso(1, "", 0, 0, 0, true, 3);
132         int expectedResult = 3;
133         int result = instance.getId_Usuario();
134         assertEquals(expectedResult, result);
135         // TODO review the generated test code and remove the default call to fail.
136         //fail("The test case is a prototype.");
137     }

```

Figura 17: Caso de prueba del método GetId_Usuario de la clase Usuario. Fuente: elaboración propia.

Métodos de pruebas

La prueba de caja negra se refiere a las pruebas que se llevan a cabo en la interfaz del software. Una prueba de caja negra examina algunos aspectos fundamentales de un sistema con poca preocupación por la estructura lógica interna del software. La prueba de caja blanca del software se basa en el examen cercano de los detalles de procedimiento. Las rutas lógicas a través del software y las colaboraciones entre componentes se ponen a prueba al revisar conjuntos específicos de condiciones y/o bucles (Pressman, 2007).

Pruebas de caja negra

Las pruebas de caja negra, también llamadas pruebas de comportamiento, se enfocan en los requerimientos funcionales del software; es decir, las técnicas de prueba de caja negra le permiten derivar conjuntos de condiciones de entrada que revisarán por completo todos los requerimientos funcionales para un programa (Pressman, 2007).

Las pruebas de caja negra intentan encontrar errores en las categorías siguientes (Pérez Jiménez, 2015):

- funciones incorrectas o faltantes.
- errores de interfaz.
- errores en las estructuras de datos o en el acceso a bases de datos externas.
- errores de comportamiento o rendimiento.
- errores de inicialización y terminación.

Pruebas de Caja Blanca

La prueba de caja blanca, en ocasiones llamada prueba de caja de vidrio, es una filosofía de diseño de casos de prueba que usa la estructura de control descrita como parte del diseño a nivel de componentes para derivar casos de prueba (Pressman, 2007).

Al usar los métodos de prueba de caja blanca, puede derivar casos de prueba que (Pressman, 2007):

- Garanticen que todas las rutas independientes dentro de un módulo se revisaron al menos una vez.
- Revisen todas las decisiones lógicas en sus lados verdadero y falso.
- Ejecuten todos los bucles en sus fronteras y dentro de sus fronteras operativas.
- Revisen estructuras de datos internas para garantizar su validez.

Resultados de las pruebas de Caja Negra

A través del método de caja negra y con el diseño de los casos de prueba se realizaron tres iteraciones de pruebas internas pertenecientes al nivel de aceptación. Dichas pruebas fueron realizadas con el objetivo de detectar y corregir errores que impidieran el correcto funcionamiento de la solución.

Para evaluar la solución se realizaron dos iteraciones donde se probó el software íntegramente, finalmente se realizó una tercera iteración donde se comprobó la resolución de todas las no conformidades detectadas. Encontrando en dichas pruebas en la primera iteración un total de siete no conformidades, de las cuales se resolvieron las siete; en la segunda iteración se detectaron tres no conformidades, de las cuales las tres fueron resueltas y en una tercera iteración no se encontraron no conformidades, quedando evidenciado el cumplimiento del objetivo general propuesto como solución a los problemas existentes. A continuación, se presentan los resultados arrojados durante las iteraciones de las pruebas aplicadas a través de un gráfico de barras:

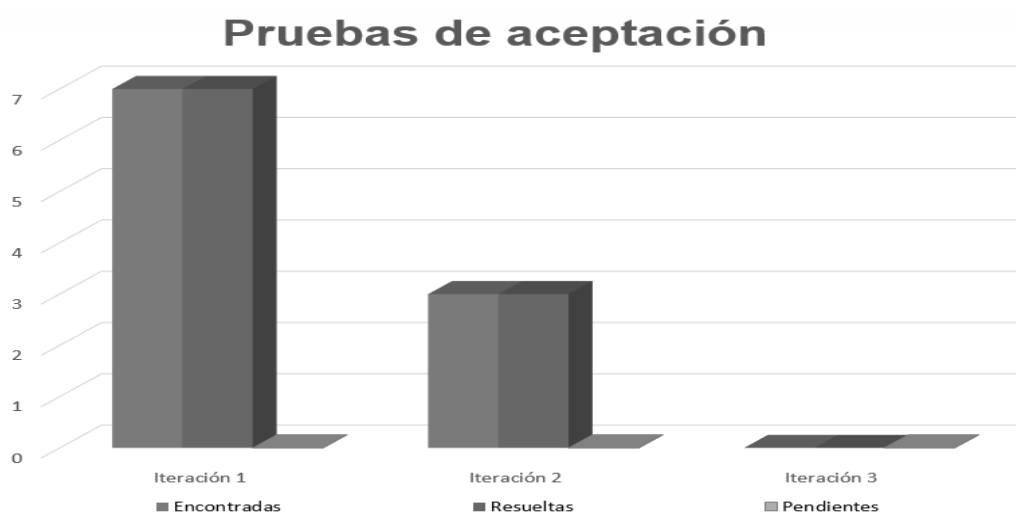


Figura 18: Resultado de las pruebas de aceptación. Fuente: elaboración propia.

Resultados de las pruebas de caja blanca

En total se implementaron 10 casos de pruebas unitarias, de ellas 3 aportaron no conformidades y se corrigieron antes de terminar la iteración. Los resultados aparecen en el siguiente gráfico.

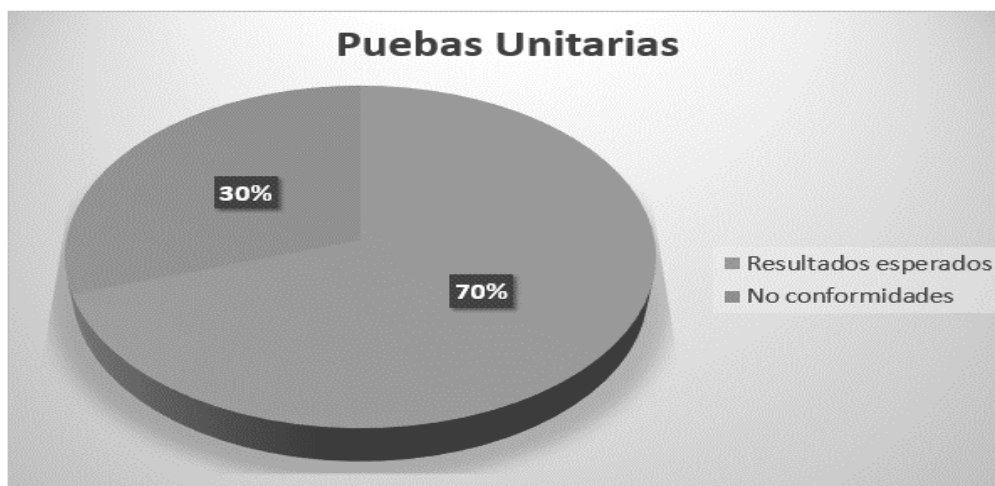


Figura 19: Resultado de las pruebas unitarias. Fuente: elaboración propia.

Una vez realizada la corrección de los errores detectados, los métodos de cada clase se ejecutaron de forma correcta, obteniendo de ellos los resultados esperados. Se tomaron precauciones para evitar errores similares y se definieron nuevos casos de prueba para verificar el correcto desempeño de las funcionalidades.

Técnica ladov

La técnica ladov constituye una vía para el estudio del grado de satisfacción. Fue creada para establecer el nivel de satisfacción por la profesión de carreras pedagógicas. Luego, algunos autores la han modificado y aplicado para valorar la satisfacción en múltiples campos y como parte de diagnósticos y validaciones en diferentes investigaciones. Se basa en la aplicación de un cuestionario que tiene una estructura interna determinada, que sigue una relación entre tres preguntas cerradas y un análisis posterior de otro conjunto de preguntas abiertas. La relación entre las preguntas cerradas se establece a través del denominado Cuadro Lógico de ladov (Ver Tabla 19), el cual posibilita determinar posteriormente el nivel de satisfacción del usuario y del grupo.

Tabla 19: Cuadro Lógico de ladov modificado por el autor.

	¿Considera usted qué no es necesario disminuir el tiempo de respuesta a casos de intoxicaciones por plaguicidas en el Cenatox?								
	No			No se			Si		
	¿Cree usted que el uso de la aplicación desarrollada puede disminuir el tiempo de respuesta a casos de intoxicaciones por plaguicidas en el Cenatox?								
¿Le satisface la propuesta de solución desarrollada para el Cenatox?	Si	No se	No	Si	No se	No	Si	No se	No
Me satisface mucho	1	2	6	2	2	6	6	6	6
No me satisface tanto	2	2	3	2	3	3	6	3	6
Me da lo mismo	3	3	3	3	3	3	3	3	3

Me insatisface más de lo que me satisface	6	3	6	3	4	4	3	4	4
No me satisface nada	6	6	6	6	4	4	6	4	5
No sé qué decir	2	3	6	3	3	3	6	3	4

Se utilizó la técnica para medir la satisfacción de usuarios del Cenatox con relación a la herramienta desarrollada.

Para obtener los resultados de la aplicación de la técnica es necesario conocer la escala de satisfacción, así como la fórmula para determinar el Índice de Satisfacción Grupal (ISG). La escala de satisfacción responde a la siguiente estructura, en función de la puntuación obtenida luego de aplicado el cuestionario referido:

1. Clara satisfacción
2. Más satisfecho que insatisfecho
3. No definida
4. Más insatisfecho que satisfecho
5. Clara insatisfacción
6. Contradictoria

Luego de aplicado el cuestionario y haber triangulado las preguntas cerradas en el Cuadro Lógico de ladov, el número resultante de la interrelación de las tres preguntas cerradas indica la posición de cada cual en dicha escala de satisfacción.

Para poder ponderar el ISG se establece una escala numérica entre +1 y -1 como se muestra a continuación:

- +1 Máximo de satisfacción
- +0.5 Más satisfecho que insatisfecho
- 0 No definido y contradictorio
- 0.5 Más insatisfecho que satisfecho
- 1 Máxima insatisfacción

Luego es posible calcular el ISG a partir de la siguiente fórmula:

$$ISG = \frac{A (+1) + B (+0.5) + C (0) + D (-0.5) + E (-1)}{n}$$

El ISG, como se especificó en la escala numérica anterior, fluctúa entre + 1 y - 1. Es por ello que, una vez calculado, los valores que se encuentren comprendidos entre - 1 y - 0,5 indican insatisfacción; los comprendidos entre - 0,49 y + 0,49 evidencian contradicción y los que se ubiquen entre 0,5 y 1 indican que existe satisfacción.

En la presente investigación fue aplicada una encuesta integrada por 8 preguntas (Ver Anexo 2) a una muestra representativa de 15 personas, 10 de ellos personal médico, 3 estadísticos, y 2 administrativos pertenecientes al Cenatox. La población está constituida por los estadísticos y administrativos del nivel

secundario de la salud en la provincia La Habana. Para la selección de la muestra se utilizó el muestreo probabilístico aleatorio simple. El valor ISG obtenido al aplicar la técnica fue 0,867, el cual se encuentra en el intervalo de satisfacción, por lo que se puede concluir que existe un alto grado de satisfacción la aplicación desarrollada.

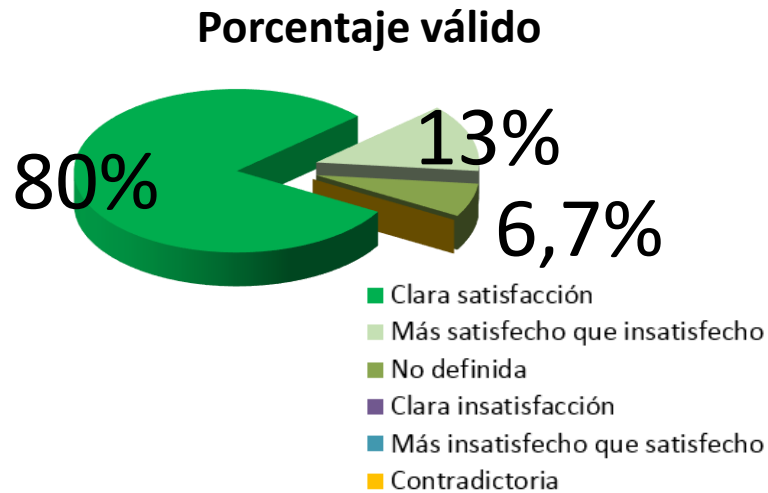


Figura 20: Resultado de la técnica ladov.

Conclusiones del capítulo

En este capítulo se describieron las pruebas correspondientes a la metodología XP. Se implementaron las tareas de ingeniería correspondientes a las HU definidas en cada iteración. Por último, se realizaron las pruebas unitarias y de aceptación, las cuales arrojaron ciertas no conformidades que fueron solucionadas.

Conclusiones

Con la realización del presente trabajo de diploma se ha cumplido con el objetivo general propuesto, así como con las tareas de la investigación, arribándose a las siguientes conclusiones:

- El análisis de las herramientas y software identificados propicia asegurar que estos no cubren todas las necesidades descritas en la problemática relacionadas con el análisis de datos acerca de casos de intoxicaciones por plaguicidas presentando un diagnóstico en menor tiempo que con las vías tradicionales.
- El empleo de las herramientas, metodologías y tecnologías seleccionadas posibilitaron que la implementación del sistema se realizara exitosamente, cumpliendo de esta forma con el objetivo general.
- Con el diseño e implementación de las funcionalidades propuestas se obtuvo una aplicación informática que responde a las necesidades del cliente definidas durante la etapa de análisis.
- El sistema basado en casos desarrollado propició la identificación oportuna de casos por intoxicación basándose en la experiencia adquirida de registros previos.
- La validación del software desarrollado mostró que los resultados obtenidos tras su ejecución son correctos y que puede ser utilizado por especialistas del Cenatox.

Recomendaciones

Para futuras investigaciones se recomienda lo siguiente:

- Informatizar la hoja de seguridad de plaguicidas para propiciar una mejor gestión de la información que estas poseen.
- Incorporar funcionalidades que respondan a la informatización de los demás procesos que se ejecutan en el Cenatox.

Referencias Bibliográficas

- Organización Panamericana de la Salud. Situación Epidemiológica de las Intoxicaciones Agudas por Plaguicidas en el Istmo Centroamericano, Boletín Epidemiológico, Vol. 23 No. 3, septiembre 2002.
- RAMÍREZ, J. A.; LACASAÑA, Matías. Plaguicidas: clasificación, uso, toxicología y medición de la exposición. Arch Prev Riesgos Labor, 2001, vol. 4, no 2, p. 67-75.
- CENATOX. [en línea], [sin fecha]. [Consulta: 15 noviembre 2017]. Disponible en: <http://www.cenatox.sld.cu/>.
- Ortiz R. (2002). Los plaguicidas en México. Monografías.com. <http://www.monografias.com/trabajos14/losplaguicidas/losplaguicidas.ml> [Acceso 21 Julio 2015]
- Malagón Cañizares, J., Cervera López, e. Y Lliso Laguarda, J.J., 2010. Productos fitosanitarios: materias activas y preparados.
- López Arias, D.Y., 2015. Guía para el diagnóstico y tratamiento de las intoxicaciones por plaguicidas. 2015. S.l.: s.n.
- (COATEA), C.D.O.P.L.A.D.E.A., [sin fecha]. Entendiendo una hoja de seguridad [en línea]. S.l.: s.n. [Consulta: 25 noviembre 2017]. Disponible en: <https://www.gob.mx/cms/uploads/attachment/file/278111/HDS.pdf>.
- DiagnosMD-Características principales. [en línea], [sin fecha]. [Consulta: 21 noviembre 2017]. Disponible en: http://www.diagnosmd.com/que_es.php.
- Ortiz Gálvez, C.D.-- criortiz@alumnos ubiobio cl y QUIJADA FUENTES, C.F.-- caquijad@alumnos ubiobio cl, 2012. Software de apoyo a la selección de plaguicidas para cultivos agrícolas. [en línea], Disponible en: <http://repobib.ubiobio.cl/jspui/handle/123456789/684>.
- Rectox - Institucional - Software ::: SerTox ::: Sertox [en línea], [sin fecha]. Disponible en: <http://www.sertox.com.ar/modules.php?name=Content&pa=showpage&pid=39>.
- Orallo, Hernández, et al. Introducción a la Minería de Datos. Pearson Prentice Hall,, 2004.
- Montes, Manuel; DE LENGUAJE NATURAL, Gómez Laboratorio. Minería de texto: Un nuevo reto computacional. Obtenido de [https://ccc. inaoep. mx/~ mmontesg/publicaciones/2001/MineriaTexto-md01. pdf](https://ccc.inaoep.mx/~mmontesg/publicaciones/2001/MineriaTexto-md01.pdf), 2001.
- Pérez Jiménez, Sebastián, et al. Minería de procesos. 2015.
- Freeman, James A.; Skapura, David M. Redes neuronales. Algoritmos, aplicaciones y, 1993.
- Giarratano, Joseph; Riley, Gary. Sistemas expertos: principios y programación. International Thomson, 2001.

- Cruz, Pedro Ponce; Herrera, Alejandro. Inteligencia artificial con aplicaciones a la ingeniería. Marcombo, 2011.
- Martínez Sánchez, N., García Lorenzo, M.M. y García Valdivia, Z.Z., 2009. Modelo para diseñar sistemas de enseñanza-aprendizaje inteligentes utilizando el razonamiento basado en casos. Revista Avances en Sistemas e Informática [en línea], vol. 6, no. 3. ISSN 1657-7663. Disponible en: <http://www.redalyc.org/resumen.oa?id=133112611006>.
- Ayala, Alejandro Peña. Sistemas basados en Conocimiento: Una Base para su Concepción y Desarrollo. México: s.n., 2006.
- Badaró, Sebastián. Sistemas Expertos: Fundamentos, Metodologías y Aplicaciones. 2013.
- Martí Pérez, I., 2012. Sistema para la ayuda a la toma de decisiones en el diagnóstico y tratamiento de dislipoproteinemias basado en la ingeniería del conocimiento. La Habana: s.n.
- Agudo, B. 2001. Introducción al razonamiento basado en casos. Departamento de sistemas informáticos y programación. España: Universidad complutense de Madrid, 2001.
- Rodriguez, Yalila Arean. 2003. Una Herramienta de búsqueda inteligente en bases de datos utilizando técnicas de RBC. Habana: s.n., 2003.
- Martínez Sánchez Natalia, Gheisa Ferreira Lorenzo, María M. García Lorenzo, Zenaida García Valdivia. 2008. El Razonamiento Basado en Casos en el ámbito de la Enseñanza/Aprendizaje. s.l. : Revista de Informática Educativa y Medios Audiovisuales, 2008. 1667-8338.
- Apuntes. Ingeniería del software. Sistemas Informáticos. Nivel de madurez software. Informática Aplicada a la Gestión Pública. 2005/06-2. Universidad de Murcia. Rafael Barzanallana. [en línea], [sin fecha]. [Consulta: 9 noviembre 2017]. Disponible en: <http://www.um.es/docencia/barzana/IAGP/lagp2.html>.
- Pressman, Roger. Ingeniería de Software. Un enfoque práctico. 2007.
- Villegas, Adrian Anaya. Monografías.com. [En línea] 2009. <http://www.monografias.com/trabajos51/programacion-extrema/programacion-extrema2.shtml#top>.
- Joskowicz, José. Reglas y prácticas en eXtreme Programming. Universidad de Vigo, 2008, vol. 22.
- Definición de lenguaje de programación — Definición de. Definición. de [en línea], [sin fecha]. Disponible en: <https://definicion.de/lenguaje-de-programacion/>.
- Lenguajes de programación, programación Java. [en línea], [sin fecha]. [Consulta: 9 noviembre 2017]. Disponible en: <http://www.lenguajes-de-programacion.com/programacion-java.shtml>.
- Oracle. Oracle Help Center. [En línea] ene de 2018. <http://docs.oracle.com/javase/8/javafx/get-started-tutorial/jfx-overview.htm#JFXST784>

- Oracle Help Center. [En línea] ene de 2018.
<http://www.oracle.com/technetwork/java/javase/downloads/javafxscenebuilder-info-2157684.html>
- NetBeans IDE 6.1 Information. [en línea], [sin fecha]. [Consulta: 15 noviembre 2017]. Disponible en:
<https://netbeans.org/community/releases/61/>.
- ¿Qué es junit? | La paradita. [en línea], [sin fecha]. [Consulta: 9 mayo 2018]. Disponible en:
<https://elrincondeaj.wordpress.com/2012/07/09/junit/>.
- ¿Qué es y para qué sirve UML? Versiones de UML (Lenguaje Unificado de Modelado). Tipos de diagramas UML. [en línea], [sin fecha]. [Consulta: 13 mayo 2018]. Disponible en:
https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=688:que-es-y-para-que-sirve-uml-versiones-de-uml-lenguaje-unificado-de-modelado-tipos-de-diagramas-uml&catid=46&Itemid=163.
- ¿Qué es PostgreSQL? | Microbuffer. [en línea], [sin fecha]. [Consulta: 14 noviembre 2017]. Disponible en: <https://microbuffer.wordpress.com/2011/05/04/que-es-postgresql/>.
- Michaud, Frederic. 2015. Identifying Key Attack Surface Resources with Dynamic Analysis. 2015.
- Chávez-Herández J. A., et al. Utilización de la Inteligencia Artificial en el diagnóstico patológico de edificaciones de valor patrimonial. Informes de la Construcción, 2012, vol. 64, no 527, p. 297-305.
- Letelier, P. y Penadéz, M.C., 2006. Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP). www.cyta.com.ar/ta0502/v5n2a1.htm [en línea]. Disponible en: <http://www.cyta.com.ar/ta0502/v5n2a1.htm>.
- González, Yanette Díaz; Romero, Yenisleidy Fernández. Patrón Modelo-Vista-Controlador. Revista Telemática, 2012, vol. 11, no 1, p. 47-57.
- ¿Qué es un Patrón de Diseño? [en línea], [sin fecha]. [Consulta: 6 marzo 2018]. Disponible en: <https://msdn.microsoft.com/es-es/library/bb972240.aspx>.
- Giraldo, Gloria L.; Acevedo, Juan F.; Moreno, David A. Una ontología para la representación de conceptos de diseño de software. Revista Avances en Sistemas e Informática, 2011, vol. 8, no 3.
- Sommerville, Ian. Ingeniería del software. Pearson Educación, 2005.
- Gutiérrez, J. J., et al. Pruebas del Sistema en Programación Extrema. Departamento de Lenguajes y Sistemas Informáticos. Universidad de Sevilla, 2006.
- Malforá, Dayvis, y otros. Testing en eXtreme Programming. 2006.

Glosario de términos

Toxicología: es la ciencia que se encarga de analizar los efectos y daños reversibles e irreversibles causados en el organismo por sustancias tóxicas, que se basan en una estructura química de la naturaleza que no existen o son inhabituales, ya que se trata de compuestos que fueron sintetizados por los hombres en laboratorios. Existen varios tipos de toxicologías, entre ellas encontramos: toxicología ambiental, industrial, de los alimentos y de los medicamentos.

Intoxicación: conjunto de síntomas y signos resultante de la acción de un tóxico sobre el organismo.

Tóxico: cualquier sustancia que por sus propiedades físico químicas es capaz de provocar alteraciones funcionales y/o orgánica en los seres vivos.

Antídotos: aquellas sustancias que se oponen a la acción del tóxico no actuando sobre los receptores biológicos, sino sobre el propio tóxico, por inactivación o impidiendo su conexión con los receptores, pero siempre sobre la sustancia tóxica.

Bibliografía

- Alxbl, 2009. /*Diagonal Asterisco*/: Integración de JUnit y Netbeans. /*Diagonal Asterisco*/ [en línea]. [Consulta: 9 mayo 2018]. Disponible en: <http://diagonalasterisco.blogspot.com/2009/07/integracion-de-junit-y-netbeans.html>.
- Amatrian, I.H. y Tondato, L.G., [sin fecha]. *Sistemas Expertos y Sistemas Basados en Conocimientos* [en línea]. S.I.: s.n. Disponible en: <http://sistemas.unla.edu.ar/sistemas/sls/ls-4-optativa-SBC/pdf/ISE-GdE-1-Introduccion-a-los-SSEE-y-SSBBCC-Material.pdf>.
- Casos de prueba: JUnit. [en línea], [sin fecha]. [Consulta: 9 mayo 2018]. Disponible en: <http://www.jtech.ua.es/j2ee/publico/lja-2012-13/sesion04-apuntes.html>.
- Díaz Labrador, M. y Collazo Garcia, A., 2013. *La programación extrema*. S.I.: s.n.
- Distancia de Hamming - Algoritmos Similaridad & Distancia. [en línea], [sin fecha]. Disponible en: <https://sites.google.com/site/algoritmossimilaridaddistancia/distancia-de-hamming>.
- Garzías, J., 2014. Los patrones GRASP. *Javier Garzías* [en línea]. Disponible en: <http://www.javiergarzas.com/2014/08/los-patrones-grasp.html>.
- Londoño, J.H.A., 2005. Ingeniería de Software: TIPOS DE PRUEBAS DE SOFTWARE. *Ingeniería de Software* [en línea]. [Consulta: 28 marzo 2018]. Disponible en: <http://ing-sw.blogspot.com/2005/04/tipos-de-pruebas-de-software.html>.
- Martinez, A.S., Luciano, [sin fecha]. Pruebas Unitarias, Parte 1: Introducción y utilización de objetos simulados (Mock). [en línea]. Disponible en: <http://www.microgestion.com/index.php/mg-developers/articulos/74-unit-test-part1-mock>.
- Miluska Azabache Gonzales, 14:39:07 UTC. Pruebas de software. [en línea]. Software. S.I. [Consulta: 29 marzo 2018]. Disponible en: <https://es.slideshare.net/miluskaazabachegonzales/pruebas-de-software-36522714>.
- Ortiz Gálvez, C.D.-- criortiz@alumnos ubiobio cl y QUIJADA FUENTES, C.F.-- caquijad@alumnos ubiobio cl, 2012. Software de apoyo a la selección de plaguicidas para cultivos agrícolas. [en línea], Disponible en: <http://repobib.ubiobio.cl/jspui/handle/123456789/684>.
- Sánchez, N.F., 2013. Alcasoft: Java: pruebas unitarias con JUnit y NetBeans. *Alcasoft* [en línea]. [Consulta: 9 mayo 2018]. Disponible en: <http://alcasoft.blogspot.com/2013/05/java-pruebas-unitarias-con-junit-y.html>.
- Sánchez, P.J.Z., 2013. Niveles de Prueba del Software. *PRUEBAS DE SOFTWARE* [en línea]. Disponible en: <https://pruebasdelsoftware.wordpress.com/2013/01/21/niveles-de-prueba-del-software/>.
- Un enfoque actual sobre la calidad del software. [en línea], [sin fecha]. [Consulta: 7 marzo 2018]. Disponible en: http://www.bvs.sld.cu/revistas/aci/vol3_3_95/aci05395.htm.
- Dante, Gloria Ponjuán. Gestión documental, de información y del conocimiento... puntos de contacto y diferencias. *Ciencias de la Información*, 2003, vol. 34, no 3.

Macías, Arturo Barraza. La consulta a expertos como estrategia para la recolección de evidencias de validez basadas en el contenido. Investigación Educativa Duranguense, 2007, no 7, p. 5-14.

Anexos**Anexo 1:** Entrevista realizada al Dr. Raúl González Pérez.

Para la entrevista fue seleccionado la Dr. Raúl González Pérez, con varios años de experiencia en el tema de investigación.

Preguntas:

1. ¿Con que cantidad de plaguicidas cuenta el Cenatox en la lista oficial de plaguicidas?
2. ¿Cuál es el promedio de casos que residen anualmente?
3. ¿Aproximadamente cuánto tiempo tardan en diagnosticar un caso?

Anexo 2: Encuesta aplicada a especialistas del Cenatox.

Estimado (a): la presente encuesta forma parte de una investigación que está dirigida a comprobar la factibilidad del uso de la herramienta PlaguiTox para uso del Cenatox. Por cuanto, sus valoraciones acerca de los asuntos que se someten a su consideración servirán de ayuda.

Tabla 20: Encuesta satisfacción de usuarios potenciales.

Evaluación del software PlaguiTox para el Cenatox.		
No.	Preguntas	Respuestas
1	¿Conoce los procesos que se llevan a cabo en el Cenatox?	Si__ No__
2	¿Sabe usted el tiempo que demora un caso de intoxicación en tener solución?	Si__ No__
3	¿Sabe usted lo que representa que exista demora en diagnosticar un caso?	Si__ No__ No se__
4	¿Considera usted que no es necesario disminuir el tiempo de respuesta a casos de intoxicaciones por plaguicidas en el Cenatox?	Si__ No__ No se__
5	Si tuviera que realizar un proceso del Cenatox con una herramienta informática ¿Cuál sería su motivación? Seleccione una sola opción	Ahorro de tiempo__ Resultados confiables__ Basado en datos reales__
6	¿Cree usted que el uso de la aplicación desarrollada puede disminuir el tiempo de respuesta a casos de intoxicaciones por plaguicidas en el Cenatox?	Si__ No__ No se__
7	¿Qué elemento del software desarrollado le satisface más?	Usabilidad__ Comprensión__ Ninguno__ No sé__
8	¿Le satisface la propuesta de solución desarrollada para el Cenatox?	Me gusta mucho__ No me gusta tanto__ Me da lo mismo__ Me disgusta más de lo que me gusta__ No me gusta nada__ No sé qué decir__

Anexo 3: Historias de usuario.

Tabla 21: Descripción de la HU 1 Gestión de Usuarios.

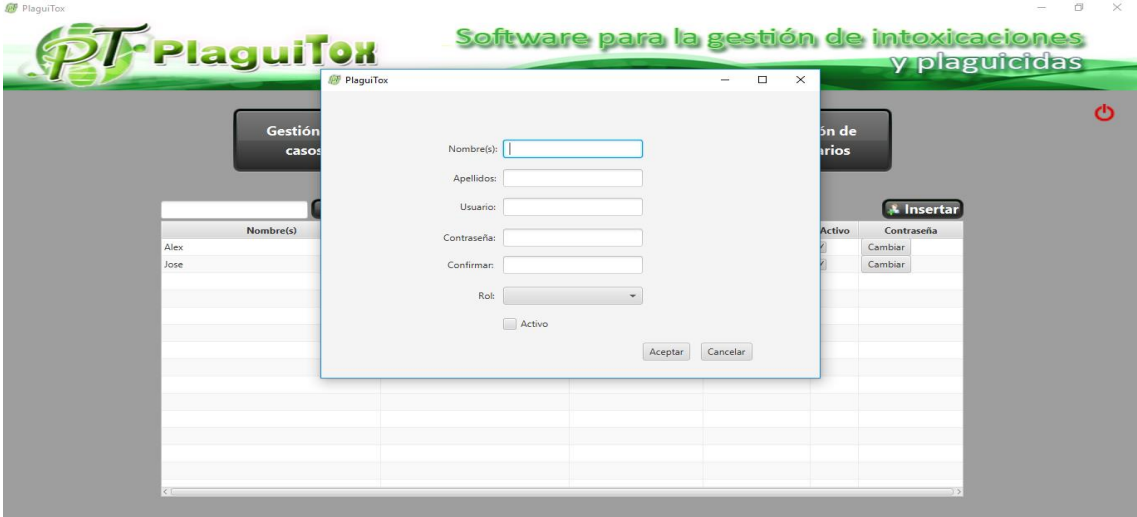
Historia de Usuario	
Número: 1	Nombre de HU: Gestión de Usuarios
Referencia: Ninguna	Prioridad: media
Iteración Asignada: 1	Puntos Estimados: 2.5
Descripción: El software debe permitir insertar, modificar, listar, buscar, habilitar y deshabilitar los roles con los que trabaja el software.	
Observaciones: Los Usuarios son almacenados indicando: Nombre; Apellidos; Rol, si es Usuario o Administrador del sistema; Usuario y Contraseña.	
Prototipo de interfaz:	
	

Tabla 22: Descripción de la HU 4 Consultar Hoja de Seguridad.

Historia de Usuario	
Número: 4	Nombre de HU: Consultar Hoja de Seguridad
Referencia: 2,3	Prioridad: media
Iteración Asignada: 2	Puntos Estimados: 2
Descripción: El software debe permitir visualizar la hoja de seguridad del Plaguicida.	
Observaciones: Para consultar la HDS de plaguicidas se debe especificar la ruta donde se encuentra, indicando en el campo hoja de seguridad, dentro de la funcionalidad insertar, en la gestión de plaguicidas, la dirección.	

Prototipo de interfaz:



Tabla 23: Descripción de la HU 5 Exportar a PDF.

Historia de Usuario

Número: 5	Nombre de HU: Exportar a PDF
Referencia: 3	Prioridad: media
Iteración Asignada: 3	Puntos Estimados: 2

Descripción: El software debe permitir generar un documento en formato PDF que contiene los datos que se obtienen como resultado de haber realizado un diagnóstico a un paciente.

Observaciones: Para exportar a PDF se debe de tener al menos un caso en la base de datos.

Prototipo de interfaz:



Anexo 4: Tarjetas CRC.


Tabla 24: Tarjeta CRC clase Gestion_plaguicidaContraller.

Tarjetas CRC	
Clase: Gestion_plaguicidaContraller	
Responsabilidad	Colaborador
Es la clase encargada de: <ul style="list-style-type: none"> • Insertar plaguicidas. • Modificar plaguicidas. • Listar plaguicidas. 	Datos Insertar_plaguicidasController Buscar_plaguicidasController Producto Peligro

Tabla 25: Tarjeta CRC clase Gestion_usuarioContraller.

Tarjetas CRC	
Clase: Gestion_usuarioController	
Responsabilidad	Colaborador
Es la clase encargada de: <ul style="list-style-type: none"> • Insertar usuarios. • Modificar usuarios. • Listar usuarios. • Habilitar usuarios. • Deshabilitar usuarios. 	Insertar_usuarioController AutenticarController Cambiar_contrasennaController Datos Usuario

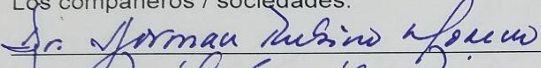
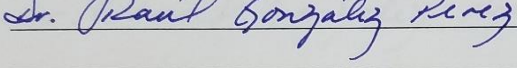
Anexo 5: Aval del resultado científico técnico.


CENTRO DE INFORMÁTICA MÉDICA
AVAL DE RESULTADO CIENTÍFICO TÉCNICO

La Habana, mayo 16 de 2018

Aval sobre el trabajo titulado "*PlaguiTox: Software para la gestión de hojas de seguridad de plaguicidas del Centro Nacional de Toxicología*" a propósito de su presentación como Trabajo de Diploma para Optar por el título de Ingeniero en Ciencias Informáticas.

Los compañeros / sociedades:

Luego del intercambio realizado con los autores de la propuesta y en correspondencia con el valor científico técnico y social de la misma deciden aprobar y avalar el resultado obtenido.

Como resultado del trabajo de investigación, desarrollo e innovación tecnológica fueron obtenidos 4 componentes de software:

- Módulo para la gestión de casos de intoxicaciones.
- Módulo para la gestión de plaguicidas.
- Módulo para la gestión de usuarios.
- Sistema Basado en Casos para el apoyo a la toma de decisiones

Se considera que los resultados obtenidos son de gran valor práctico y social para el desarrollo de la salud y la informatización del sector en Cuba.

Teniendo en cuenta los elementos mencionados antes, se considera que la investigación presentada reúne los requisitos para ser presentado como resultado práctico de la investigación de los autores.

Firman la presente:

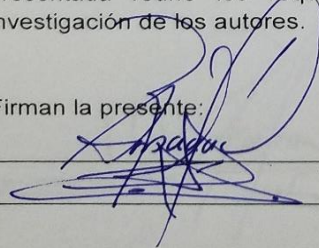
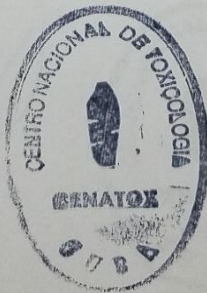



Figura 18: Aval del resultado científico técnico. Fuente: Centro de informática médica.

Anexo 6: Solicitud de cooperación.

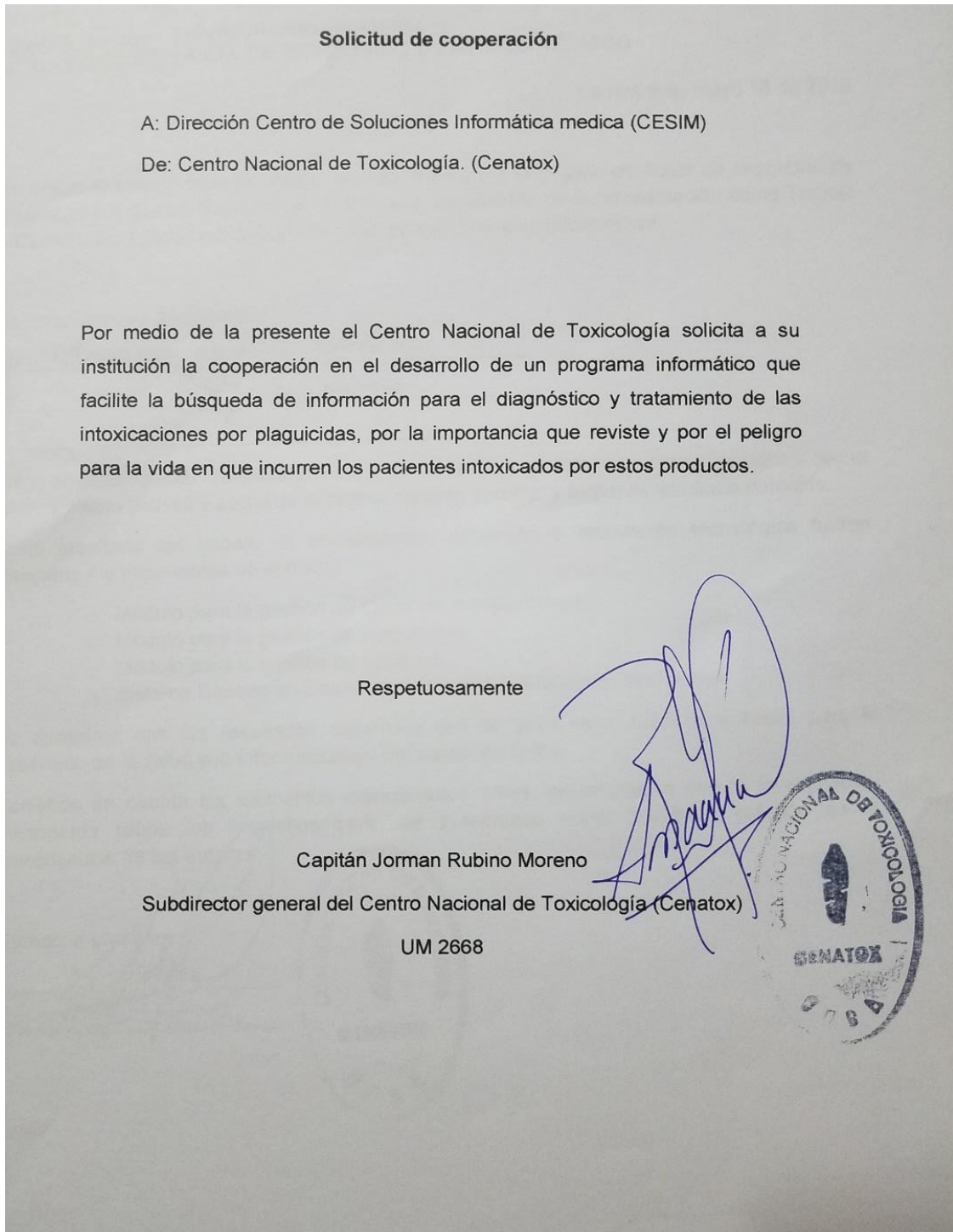


Figura 19: Solicitud de cooperación. Fuente: Cenatox.