



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

**SISTEMA INTEGRADO PARA EL
ANÁLISIS DE FALLA DE ELEMENTOS
MECÁNICOS**

TESIS

Que, para obtener el título de
INGENIERO EN COMPUTACIÓN

P R E S E N T A

Julián Valle Galindo

DIRECTOR DE TESIS

Dr. Víctor Hugo Jacobo Armendáriz



Ciudad Universitaria, Cd. Mx., 2017



Universidad Nacional
Autónoma de México



UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

DERECHOS RESERVADOS © PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis está protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

Agradecimientos

A mi alma mater, la Universidad Nacional Autónoma de México, por haberme dado la oportunidad de tener una formación académica y permitirme ser parte de la máxima casa de estudios.

A mi segundo hogar, la Honorable Facultad de Ingeniería que a través de sus profesores e instalaciones me brindaron conocimientos y experiencias para mi formación.

A mis padres, por brindarme su paciencia, comprensión y apoyo incondicional, en especial a tí madre por alentarme y darme el coraje necesario para conseguir un objetivo más en mi vida.

A tí hermano, por la paciencia y el apoyo incondicional brindado durante mis estudios.

A mi director de tesis, por su paciencia, confianza, orientación y apoyo para concluir este trabajo.

A toda mi familia y amigos que directa e indirectamente influyeron para conseguir una de mis metas.

Dedicatorias

En memoria de mi abuelita Gloria, por su fuerza y coraje, me enseño que a pesar de los obstáculos y limitaciones siempre hay que buscar la manera de seguir adelante.

Índice

Prólogo	viii
Capítulo 1. Definición del proyecto	1
1.1 Objetivo general	3
1.2 Objetivos específicos	3
1.3 Alcances	4
Capítulo 2. Marco Teórico	5
2.1 Aplicación Web	6
2.2 Modelo Vista Controlador	7
2.3 Sistema Experto	8
2.4 Modelos del proceso del software	10
2.4.1 Modelo en cascada	11
2.4.2 Modelo de prototipos	11
2.4.3 Modelo evolutivo	12
2.4.4 Modelo en espiral	12
2.5 Reingeniería del software	13
Capítulo 3. Análisis	17
3.1 Análisis del proyecto	18
3.1.1 Razonamiento Basado en Casos (RBC)	18
3.1.2 Sistema experto en ejes	20
3.1.3 Sistema experto en engranes	21
3.1.4 Sistema experto en tornillos	23
3.1.5 Sistemas web	24
3.2 Requerimientos del SIAFEM	25
3.2.1 Requerimientos funcionales	25
3.2.2 Requerimientos no funcionales	26
3.2.3 Restricciones	26
3.3 Diseño del SIAFEM	27
Capítulo 4. Desarrollo	28
4.1 Herramientas de desarrollo	29
4.1.1 NetBeans IDE	29
4.1.2 Maven	29
4.1.3 PrimeFaces	30
4.1.4 Java EE	30
4.1.5 JESS	32
4.1.6 Tomcat	33
4.2 Estructuración y estrategia para el desarrollo del SIAFEM	33
4.3 Creación de la estructura de la aplicación web	35
4.3.1 Creación de las vistas de usuario	37

4.3.2 Generación de la lógica de negocio	38
Capítulo 5. Verificación y Validación	40
5.1 Pruebas	41
5.1.1 Pruebas de caja blanca	42
5.1.2 Pruebas de caja negra	43
5.1.3 Pruebas de integración	47
5.1.4 Pruebas de compatibilidad	47
5.1.5 Pruebas de usabilidad	48
Conclusiones	49
Recomendaciones	53
Referencias	55
Anexos	
Anexo A: Marco teórico de las aplicaciones cliente	
Anexo B: Capturas de pantalla de los sistemas cliente	
Anexo C: Operación del SIAFEM	

Prólogo

El presente trabajo tuvo como objetivo diseñar e implementar una herramienta tecnológica que a través de internet pueda ser consultada por personal especializado o interesado en el análisis de falla de elementos mecánicos, el usuario podrá ver soluciones que fueron implementadas y con base en la información contenida en esta herramienta tecnológica será capaz de establecer una solución particular a su problema. El sistema desarrollado consistió en integrar aplicaciones web y se realizó la migración de aplicaciones cliente desarrolladas con anterioridad a un entorno web.

En el primer capítulo se realiza la definición del proyecto, se describen las características que debe de tener el trabajo final y lo que otras personas han realizado para determinar las causas de falla en elementos mecánicos.

En el segundo capítulo se tiene de manera general conceptos teóricos empleados en el proceso de desarrollo del sistema web.

En el tercer capítulo se realizó el análisis de los sistemas que formarán parte del trabajo final, permitiendo diseñar e identificar los requerimientos funcionales y no funcionales.

En el cuarto capítulo se presenta el proceso de construcción del sistema web, la arquitectura empleada, la generación de los componentes de la interfaz de usuario y la implementación de la lógica.

En el quinto capítulo se hace la validación y verificación del sistema web a través de pruebas de software para la garantía de calidad de éste.

Capítulo 1

Definición del proyecto

Definición del proyecto

El análisis de falla de elementos mecánicos es un asunto complejo, pues implica conocimientos en áreas como mecánica, física, metalurgia, química, electroquímica, procesos de manufactura, análisis de esfuerzos, diseño y mecánica de la fractura. Siendo complicado contar con una persona experta en todas las áreas mencionadas (Jacobo, 2005).

A nivel nacional, los expertos especializados en el análisis de falla son escasos (Jacobo, Ortiz y Schouwenaars, 2007). Para dar cuenta a esta problemática Díaz (2014), Hernández (2003), Jacobo (2005) y Martínez (2014), desarrollaron diversos sistemas expertos que permiten identificar y diagnosticar las causas de falla de los elementos mecánicos (ejes, engranes, tornillos, entre otros). La función principal de los sistemas expertos es capacitar personal a través de su uso o disponer de ellos como una herramienta para determinar las causas de falla. Para Jacobo (2005), “(...) podrá ser empleado en la solución de problemas convencionales, funcionando como un colega” (p. 3).

Los sistemas expertos fueron desarrollados para ejecutarse en una computadora personal (PC), presentado inconvenientes como portabilidad, movilidad, accesibilidad y concurrencia de usuarios. Para resolver éstas limitaciones es necesario contar con una herramienta tecnológica como un ambiente integrado de información que permita a través de *Internet* la interacción con los sistemas expertos y la consulta de documentos sobre el *análisis de falla*¹.

¹ Análisis de falla, este término se refiere al análisis de problemas en elementos mecánicos.

La finalidad de este proyecto es diseñar y desarrollar una aplicación web con el objetivo de proporcionar una fuente de información relacionada con la falla de piezas mecánicas. Permitiendo al usuario consultar información sobre fallas registradas, donde verá las soluciones que fueron implementadas y será capaz de establecer una solución particular a su necesidad. Cabe mencionar que antes de comenzar con el presente trabajo se estaban desarrollando dos sistemas que formarán parte de la aplicación web final.

1.1 Objetivo general

Diseñar y desarrollar una aplicación web como un ambiente integrado de información de consulta para el análisis de falla de elementos mecánicos.

1.2 Objetivos específicos

- Crear una herramienta tecnológica de apoyo para determinar las causas de falla de elementos mecánicos.
- Construir la aplicación web para un ambiente de alojamiento Linux, también puede ser alojado en la plataforma Windows.
- Diseñar e implementar la aplicación web a manera que permita la incorporación de módulos desarrollados en un futuro.
- Integrar y/o desarrollar la migración de documentos, aplicaciones de escritorio y web que formarán parte del sistema de consulta.

1.3 Alcances

En este apartado se especifican los alcances que tendrá el Sistema Integrado para Análisis de Falla de Elementos Mecánicos (SIAFEM)².

El SIAFEM incluye:

- Dos sistemas expertos para determinar las causas de falla presentes en tornillos y engranes.
- Un buscador de información específica dentro de archivos PDF a través de palabras clave.
- Un buscador de videos.
- Apuntes sobre *análisis de falla*.

Exclusiones o limitaciones del SIAFEM:

- No genera reportes en formato PDF.
- No incorpora el sistema experto en ejes y el sistema de razonamiento basado en casos.

² En adelante se usará SIAFEM.

Capítulo 2

Marco Teórico

Marco Teórico

En este capítulo se exponen conceptos teóricos relacionados con el desarrollo del SIAFEM, los temas descritos son: Aplicación web, patrón de arquitectura de software Modelo-Vista-Controlador (MVC), sistema experto, modelos del proceso del software, reingeniería de software.

2.1 Aplicación Web

Una aplicación web es un software que el usuario puede utilizar a través de un navegador como Chrome, Firefox, Internet Explorer, Safari, Opera, entre otros.

Las ventajas que presenta una aplicación web son las siguientes:

- Información centralizada.
- El usuario no debe preocuparse por las actualizaciones del sistema.
- El usuario no necesita instalar ni comprar herramientas adicionales.
- Multiplataforma.
- Bajo costo en mantenimiento.
- Movilidad.
- Seguridad

Y sus desventajas son:

- Es necesario contar con una conexión de red.
- El tiempo de respuesta puede llegar a ser lento, dependiendo de las características del servidor y/o de la conexión de red.

2.2 Modelo Vista Controlador

El patrón de arquitectura de software MVC (*Model-View-Controller*) fue concebido en 1978-1979 por *Trygve Reenskaug* durante el desarrollo de *SmallTalk*. El patrón MVC es una arquitectura de software usado para separar la lógica de negocio de la interfaz de usuario. Permite el desarrollo de una aplicación en tres capas: el Modelo, la Vista y el Controlador. Las capas se definen como:

- El Modelo representa los datos de la aplicación y la lógica de negocio. Accede a la información y actualiza su estado.
- La Vista corresponde a la interfaz del usuario y es la encargada de mostrar la información que ha sido solicitada.
- El Controlador dirige el flujo de datos de la aplicación entre la vista y el modelo. Responde a los eventos realizados por el usuario e invoca los métodos definidos en la lógica de negocio.

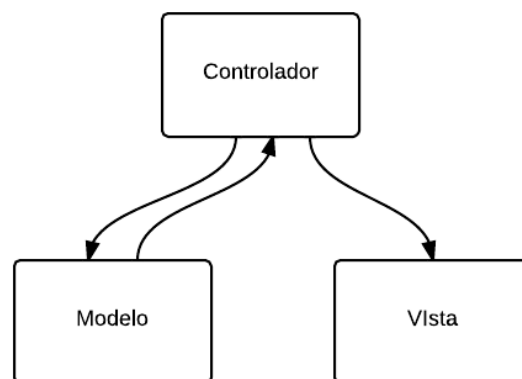


Figura 2.1 Patrón de diseño MVC. Obtenido de <http://goo.gl/wgCOKW>.

2.3 Sistema Experto

Un sistema experto es un sistema informático que emula el conocimiento y la habilidad de tomar decisiones como una persona experta lo haría para resolver problemas sobre un dominio específico. Los componentes de un sistema experto son la base de conocimientos, la base de hechos, el motor de inferencia y la interfaz de usuario. A continuación se describen:

- La base de conocimiento contiene todo el conocimiento relevante sobre el dominio del problema en donde se tienen hechos y relaciones bien definidos. El conocimiento es representado por expresiones lógicas, redes semánticas, distribuciones de probabilidad, etc.
- La base de hechos o datos, es la memoria de trabajo y contiene los datos relativos al problema que se intenta resolver.
- El motor de inferencia realiza el proceso de razonamiento, se encarga de gestionar los datos existentes en la base de conocimientos y los datos de la base de hechos, comparándolos para seleccionar las reglas que permitan obtener resultados.
- La interfaz de usuario produce el diálogo entre el sistema experto con el usuario, gestiona las entradas y salidas.

En la figura 2.2 se muestra el esquema de un sistema experto. En esta figura se puede observar la integración y la relación entre los distintos componentes mencionados anteriormente.

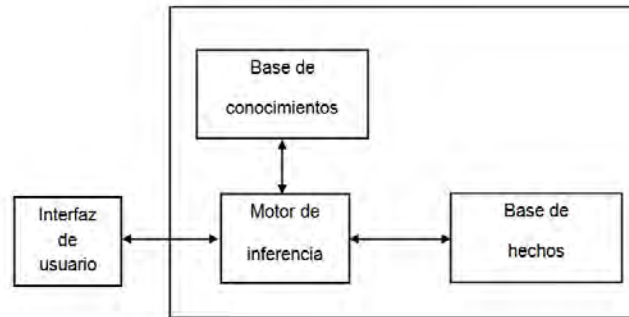


Figura 2.2 Componentes básicos de un Sistema Experto

Giarratano *et al.* (1998) señalan las siguientes características y ventajas que poseen los sistemas expertos generalmente:

- Alto desempeño: el sistema debe tener la capacidad para dar solución a un problema específico como una persona con amplia experiencia lo haría.
- Tiempo de respuesta adecuado: el sistema debe trabajar en un intervalo de tiempo para llegar a una decisión.
- Confiabilidad: el sistema debe proporcionar respuestas confiables y no ser propenso a errores.
- Comprensible: el sistema debe de tener la facultad de explicar cada una de las decisiones que le permitieron llegar a una conclusión.
- Flexibilidad: el sistema experto puede tener una gran cantidad de conocimiento, por lo que es fundamental tener un mecanismo eficiente para añadir, modificar y eliminar conocimiento.

Las ventajas que presentan los sistemas expertos para su uso son las siguientes:

- Preservación del conocimiento (permanencia): el conocimiento del sistema es permanente, a diferencia de las personas con amplia experiencia que pueden retirarse, renunciar o morir.

- Bajo costo: el precio de poner a disposición del usuario el conocimiento de un tema específico disminuye.
- Disponibilidad: el sistema podrá estar disponible las 24 horas del día.
- Experiencia múltiple (diversidad de criterios): la experiencia combinada de un grupo de especialistas o la interacción con otros sistemas expertos da como resultado un sistema más confiable.
- Rapidez: el sistema puede dar respuesta de una manera rápida y fiable en comparación con un experto humano, el sistema es más valioso cuando el tiempo de respuesta es crítico.
- Difusión del conocimiento: los usuarios podrán adquirir el conocimiento del sistema a través de su uso. Lo que permite la formación de nuevos expertos humanos o un usuario con poca experiencia pueda dar solución a un problema que requiere el conocimiento de una persona especializada.

2.4 Modelos del proceso del software

Un modelo es una abstracción de un objeto, sistema o idea. El modelo capta los elementos que tengan más relevancia y omite algunos detalles. Se define como ciclo de vida del software a las etapas o fases por las que pasa el sistema de software, es decir, desde la definición de los requisitos hasta la finalización de su uso. Las etapas por las que pasa un proyecto de software son: análisis, diseño, implementación, pruebas, despliegue, mantenimiento y retiro. Se han definido varios modelos de ciclo de vida para el desarrollo del software, los más comunes se presentan a continuación.

2.4.1 Modelo en cascada

Es un modelo secuencial para el desarrollo de software, consiste en que la siguiente fase no debe comenzar hasta que la fase previa haya finalizado.

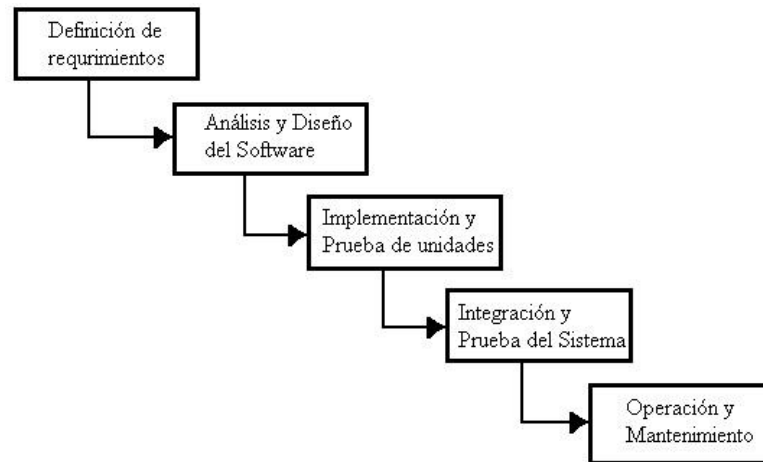


Figura 2.3 Modelo en cascada. (Sommerville, 2005).

2.4.2 Modelo de prototipos

Un sistema de software provisional o prototipo es desarrollado con rapidez y presenta una amplia facilidad de modificación priorizando ésta con respecto a la eficiencia de funcionamiento, satisfaciendo requisitos preestablecidos por el usuario final. Una vez que se han definido los requisitos, el prototipo puede ser desechado o puede ser evolucionado hasta obtener el sistema final.

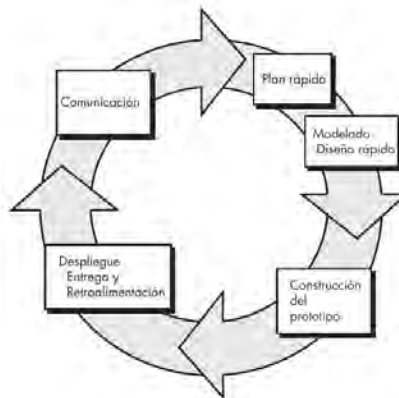


Figura 2.4 Modelo de prototipos. (Pressman, 2010).

2.4.3 Modelo evolutivo

Es un modelo iterativo, en donde los requisitos del software son flexibles y cambiantes. Se desarrollan versiones del software, cada versión del software es más completa e incluye nuevas funcionalidades.



Figura 2.5 Modelo evolutivo. (Sommerville, 2005).

2.4.4 Modelo en espiral

El proceso de desarrollo de software es representado por una espiral y es un modelo orientado a riesgos. Este modelo permite refinar de manera iterativa los requisitos, utilizando una serie de prototipos que permiten evaluar los objetivos y evaluar los riesgos. Se divide en cuatro fases:

- Planificación: se definen los objetivos, las alternativas y las restricciones. Se analizan e identifican los riesgos.
- Análisis de riesgos: se evalúa y se toma la decisión de continuar con un ciclo posterior de la espiral.
- Ingeniería: se desarrolla y se valida el sistema de software.

- Evaluación del cliente: el cliente evalúa el trabajo y sugiere modificaciones, con base en los comentarios se produce la siguiente fase de planificación y análisis de riesgos. La culminación del análisis de riesgo en cada bucle alrededor de la espiral resulta en una decisión de seguir o no seguir.

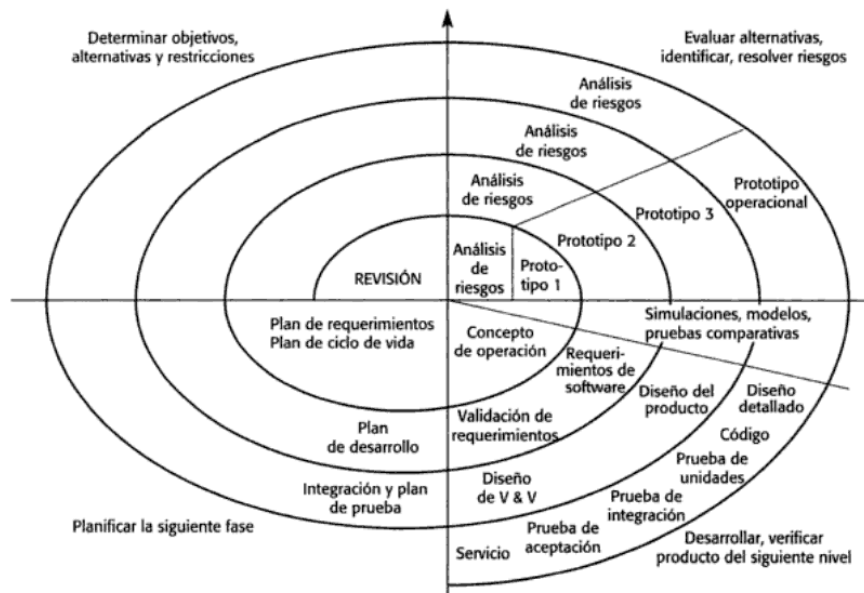


Figura 2.6 Modelo de espiral. (Sommerville, 2005).

2.5 Reingeniería del software

La reingeniería del software se refiere a la reimplementación de los sistemas heredados³, su objetivo principal es mejorar el mantenimiento del sistema existente o actualizar los sistemas existentes con las nuevas tecnologías. La funcionalidad del sistema no cambia y la arquitectura normalmente es la misma (Sommerville, 2005: 468). Además, Sommerville señala que “la reingeniería comienza con un sistema existente y el proceso de desarrollo para su reemplazo

³ *Legacy systems*, son sistemas informáticos (hardware, software y procesos) que siguen siendo útiles, fueron desarrollados hace mucho tiempo con tecnología que hoy es considerada obsoleta.

se basa en comprender y transformar el sistema original” (p. 460). Por otra parte señala los principales factores que afectan a los costes de reingeniería (p. 461), estos son:

- La calidad del software sobre el que se va a hacer reingeniería. Cuanto más baja sea la calidad del software y su documentación asociada (si la hay), más altos serán los costes de reingeniería.
- Las herramientas de soporte disponibles para la reingeniería. Normalmente no es rentable hacer reingeniería sobre un sistema de software a menos que puedan utilizarse herramientas CASE⁴ para automatizar la mayor la mayor parte de los cambios en los programas.
- La amplitud de los datos requerida. Si el sistema sobre el que se va a hacer reingeniería requiere que se conviertan grandes volúmenes de datos, el coste del proceso se incrementa de forma significativa.
- La disponibilidad de personal experto. Si el personal responsable de mantener el sistema no puede implicarse en el proceso de reingeniería, los costes se incrementarán debido a que los ingenieros encargados de la reingeniería tienen que invertir una gran cantidad de tiempo en comprender el sistema.

Los costos de la reingeniería de software dependen de la magnitud de trabajo a realizar. En la figura 2.7, se puede observar el incremento de los costos de la reingeniería de software.

⁴ *Computer Aided Software Engineering*, (Ingeniería de Software asistida por computadora).

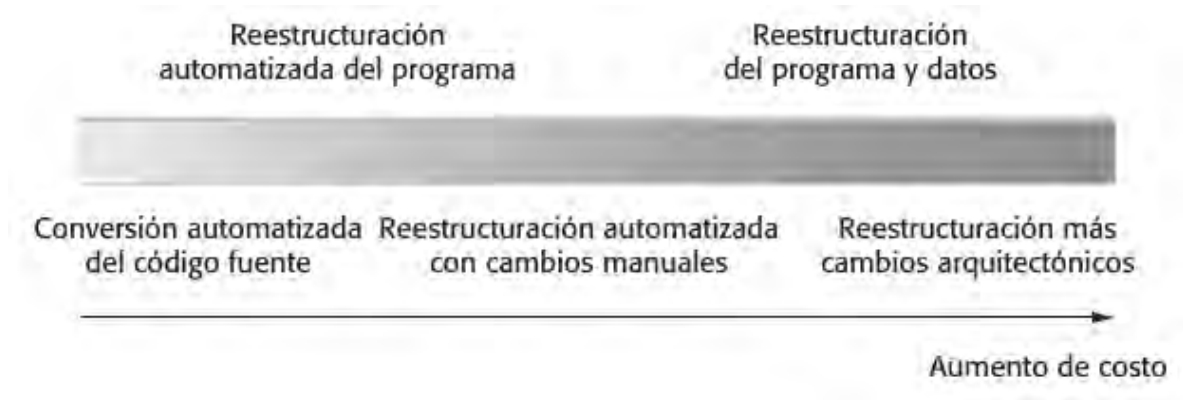


Figura 2.7 Costos de reingeniería de software. (Sommerville, 2005).

Con base en la figura anterior, según Sommerville (2005: 461), al realizar reingeniería de software la traducción del código fuente es la opción más barata mientras la migración de arquitectura del sistema de software es la más costosa.

Las actividades de reingeniería del software son:

- Ingeniería inversa: consiste en analizar y extraer información del sistema. Ayuda a documentar su estructura y funcionalidad. El objetivo principal es comprender el sistema para facilitar el mantenimiento y con base en la información obtenida permite migrar el sistema a una nueva tecnología o desarrollar nuevos sistemas.
- Reestructuración: modifica la representación del software (se estandarizan nombres, definiciones y estructuras lógicas), sin modificar su funcionalidad. Tiene como objetivo mejorar la legibilidad de los programas y simplificar la lógica.
- Ingeniería avanzada: es el proceso de trasladar las abstracciones lógicas de alto nivel a implementación física del sistema. Los niveles de la

ingeniería avanzada son: reimplementar el código, rediseñar la aplicación y revisar las especificaciones.

- Migración: es un proceso de cambio de tecnología total, consiste en convertir el sistema desde el lenguaje de programación empleado para su desarrollo a una versión más moderna o a otro lenguaje de programación, cambio de un entorno operativo a otro, cambio del sistema de gestión de base de datos (DBMS⁵).

⁵ *Database Management System*

Capítulo 3

Análisis

Análisis

3.1 Análisis del proyecto

Para que la aplicación web integre distintos sistemas de software desarrollados con diferentes herramientas tecnológicas en momentos distintos, es necesario realizar un análisis sobre cada sistema, este análisis permitirá establecer una solución sobre el sistema a desarrollar.

Los sistemas a analizar son aplicaciones cliente que permiten realizar el análisis y diagnóstico sobre fallas de elementos mecánicos, a continuación se describe el análisis realizado a cada sistema.

3.1.1 Razonamiento Basado en Casos (RBC)

El sistema de razonamiento basado en casos aplicado al análisis de falla de elementos mecánicos metálicos (Hernández, 2003), es una herramienta que cuenta con una base de conocimientos de 30 casos históricos. Fue desarrollado en *Visual Basic 6.0* (versión *Enterprise Edition*) en el año 2003 y usa un archivo de *Microsoft Access '97* para almacenar los valores lógicos de los atributos de cada uno de los 30 casos. Del sistema RBC se tiene lo siguiente:

- Documentación.
- Manual usuario.
- Código fuente.
- La base de conocimiento (cuatro imágenes para cada caso, archivos de texto en donde se tiene: el título, discusión, descripción, causas probables y

acciones correctivas del caso, la base de datos que almacena el nombre de los atributos y los valores lógicos de los 30 casos).

- Tiene una estructura que permite visualizar y localizar los diferentes recursos que necesita para su ejecución.

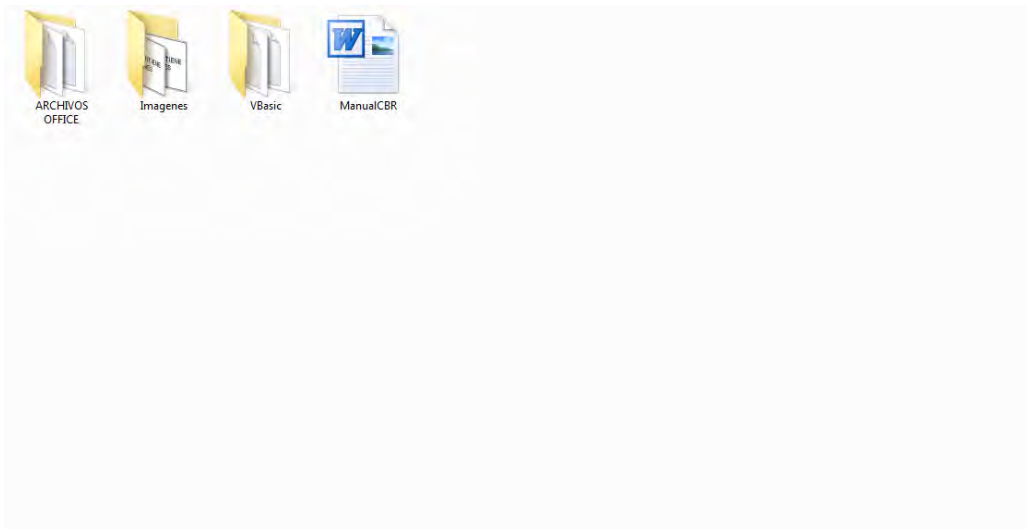


Figura 3.1 Recursos del sistema de razonamiento basado en casos

En la figura 3.1 se puede observar cómo están distribuidos los recursos del sistema anteriormente descritos. Del análisis realizado sobre este sistema se concluye:

- Cuenta con un archivo ejecutable (*.exe), lo que permite ser ejecutado en cualquier ordenador con sistema operativo *Windows*.
- Para poder ejecutar el sistema es necesario colocarlo en el directorio raíz de *Windows*.
- Durante su ejecución no se presentaron problemas de funcionalidad (ver Anexo B).

3.1.2 Sistema experto en ejes

El sistema experto en ejes permite diagnosticar el tipo de falla o fractura presente en un eje (Jacobo, 2005), es una herramienta que tiene las reglas de la base de conocimientos en un archivo de *Visual Rule Studio*. Fue desarrollado en *Visual Basic 6.0* e integra el archivo de la base de conocimientos como un archivo de tipo *Active X* en el tiempo de ejecución del programa. Del sistema experto se cuenta con lo siguiente:

- Código fuente.
- Un archivo que contiene las reglas de la base de conocimientos.
- Imágenes para cada tipo de fractura.
- La organización de los recursos del sistema es buena, lo que facilita visualizar y ubicar los elementos que necesita el sistema para su ejecución, en la figura 3.2 se puede observar lo descrito anteriormente.

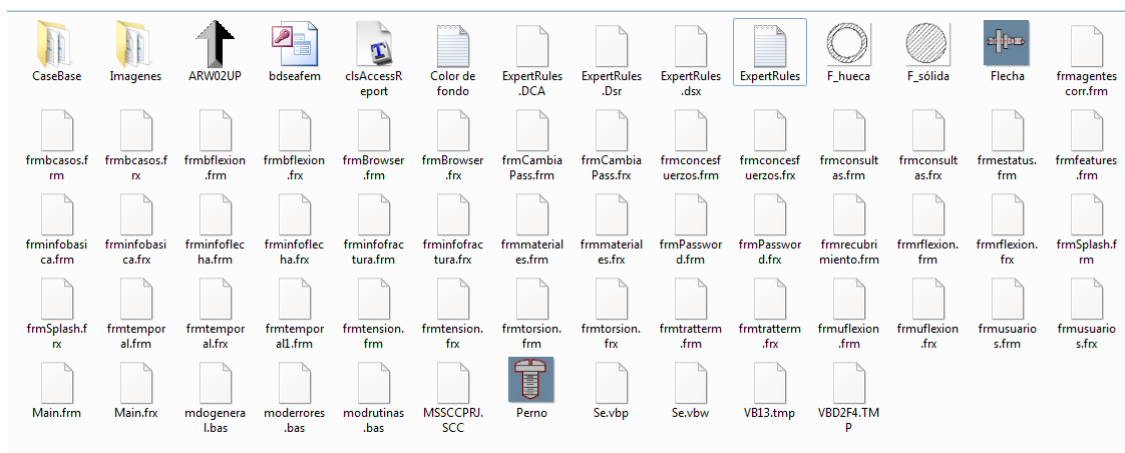


Figura 3.2 Recursos del sistema experto en ejes.

El análisis realizado sobre el sistema se concluye:

- No cuenta con un archivo ejecutable, por lo que es necesario instalar *Visual Basic 6.0* en el ordenador para ejecutar el código fuente.
- Falta la biblioteca de *Visual Rule Studio*, impidiendo la ejecución de la aplicación. Actualmente la biblioteca no está disponible comercialmente (ver Anexo B).
- No cuenta con alguna documentación o manual de usuario.

3.1.3 Sistema experto en engranes

El sistema experto para el análisis de fallas en engranes (Díaz, 2014), es un herramienta que permite diagnosticar 14 tipos de fallas. Fue desarrollado en Java y usa el *plugin* CLIPSJNI⁶ para hacer uso del *Shell* de CLIPS⁷ como motor de inferencia del sistema experto. Del sistema experto se tiene:

- Código fuente.
- Las reglas de la base de conocimiento en archivos de CLIPS.
- Una biblioteca JAR y un archivo DLL⁸ del *plugin* CLIPSJNI que permite hacer el uso del *Shell* de CLIPS dentro de cualquier aplicación java.
- Una biblioteca JAR del *plugin* *iText* para generar reportes en formato PDF.
- La estructura de los recursos del sistema están bien organizados, lo que permite ubicar y visualizar los elementos que necesita para su ejecución, ver figura 3.3.

⁶ CLIPSJNI (CLIPS *Java Native Interface*), ver Anexo A.

⁷ Acrónimo de *C Language Integrated Production*, ver Anexo A.

⁸ DLL (*Dynamic-Link Library*) es un archivo ejecutable que permite a los programas compartir código y otros recursos necesarios para realizar determinadas tareas.

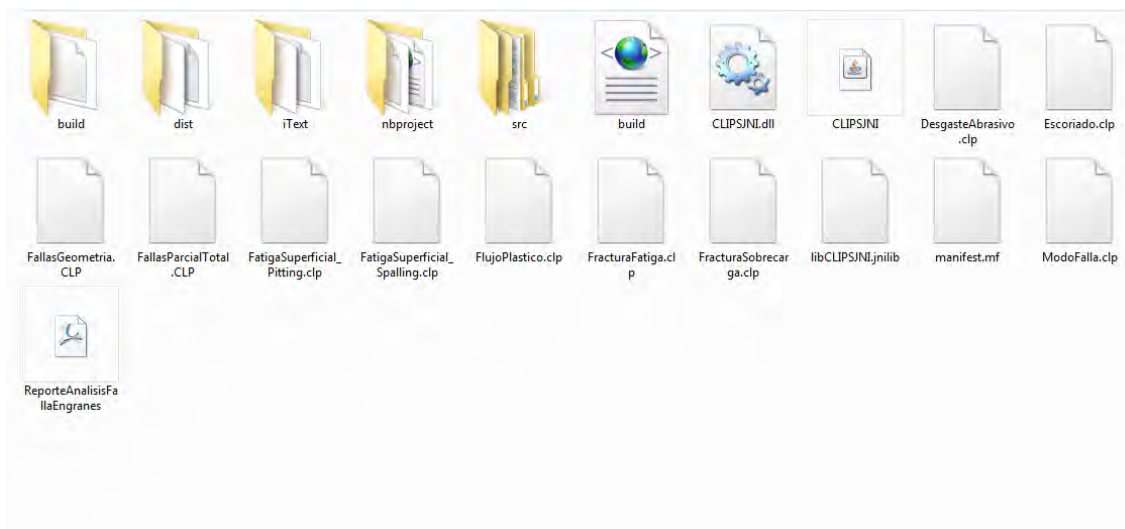


Figura 3.3 Recursos del sistema experto en engranes.

Sobre el análisis realizado a este sistema se concluye lo siguiente:

- Para hacer uso de esta aplicación es necesario contar con un entorno de desarrollo que permita ejecutar el código java.
- Es necesario agregar el archivo DLL a una ruta específica del sistema operativo *Windows* para que la aplicación java pueda interactuar con el motor de inferencias de CLIPS.
- No cuenta con algún documento o manual de usuario.
- Se tiene problemas de navegación en el botón “Siguiente” del formulario “Análisis de falla para perdida de geometría del diente” (ver Anexo B, figura B.25) y en el formulario “Análisis de falla para pérdida total o parcial del diente” (ver Anexo B, figura B.27).

Para conocer la funcionalidad que debería de hacer el botón “Siguiente” de ambos formularios se tuvo que revisar el código fuente. No se hizo corrección alguna para

el correcto funcionamiento del sistema, se tomo en cuenta este problema para cuando se realizará la migración del sistema a un entorno web.

3.1.4 Sistema experto en tornillos

El sistema experto para el análisis de falla en tornillos ferrosos (Martínez, 2015), es una herramienta que permite diagnosticar 16 tipos de fallas. Fue desarrollado en Java y al igual que el sistema experto en engranes usa el *plugin* CLIPSJNI para hacer uso del *Shell* de CLIPS. Este sistema cuenta con:

- El código fuente.
- Las reglas de la base de conocimiento están definidas en archivos de CLIPS.
- Una biblioteca JAR y un archivo DLL del *plugin* CLIPSJNI, permitiendo la interacción del *Shell* de CLIPS con la aplicación Java.
- Una biblioteca del *plugin* iText para generar archivos en formato PDF.
- La estructura de los recursos no está bien organizada, esto hace que sea difícil ubicar los elementos que la aplicación requiere para su ejecución. Se tienen recursos duplicados y hay recursos que la aplicación no usa, esto se puede observar en la figura 3.4 y en el Anexo B (figura B.28).

Del análisis realizado sobre este sistema se concluye:

- No se tiene documentación o manual de usuario.
- Se requiere de un entorno de desarrollo que permita ejecutar el código Java.

- Se necesita especificar una ruta dentro del sistema operativo *Windows* para agregar el archivo DLL, esta acción permitirá a la aplicación Java interactuar con el motor de inferencias de CLIPS.
- La ejecución del sistema en general tuvo un buen funcionamiento no se encontró ningún problema durante su operación, salvo el mal diseño que tiene la interfaz de usuario (ver Anexo B). Pues en algunos casos no permite ver la descripción de las acciones que realiza el usuario.

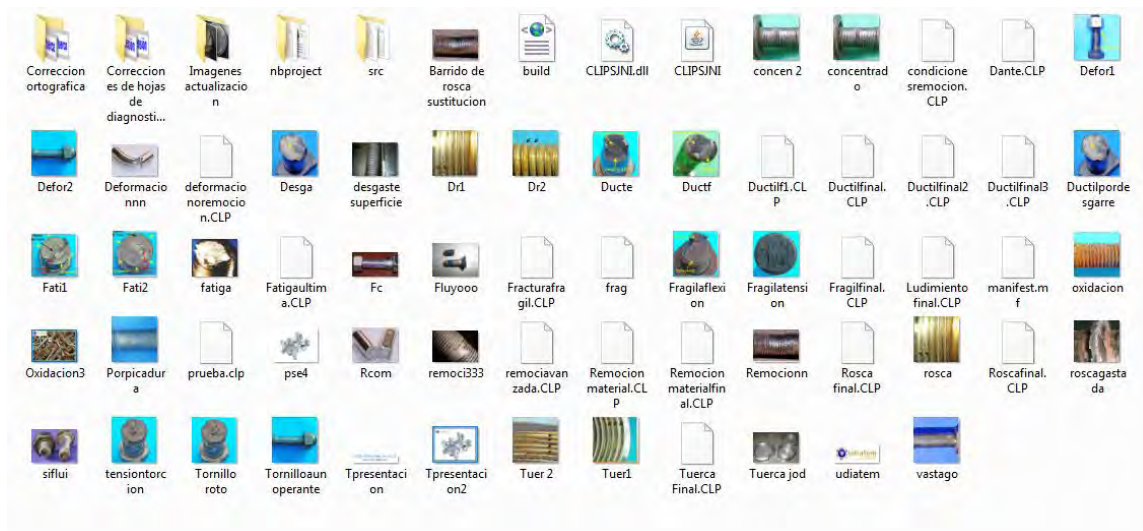


Figura 3.4 Recursos del sistema experto en tornillos.

3.1.5 Sistemas web

Antes de comenzar con el presente trabajo, se encontraban en la etapa de desarrollo dos sistemas web, a continuación se describen.

Casos históricos

La funcionalidad de este sistema es buscar palabras clave dentro de artículos relacionados con el análisis de falla de elementos mecánicos. El resultado que da

este sistema es una lista de los artículos que coinciden con los criterios de búsqueda. De este sistema se conoce:

- Herramientas tecnológicas empleadas en su desarrollo, estas son: Java EE, Spring, Hibernate, PrimeFaces, Maven, PostgreSQL.
- Cuenta con una base de datos de 300 artículos relacionados con el análisis de falla en piezas mecánicas.
- Los artículos están en formato PDF.
- El diseño de la interfaz de usuario.

Videos y presentaciones

Este sistema tiene como funcionalidad buscar videos y/o presentaciones en *Internet* que coincidan con los criterios de búsqueda del usuario, el resultado de la búsqueda se muestra en forma de lista. Las herramientas tecnológicas empleadas para el desarrollo de este sistema son las mismas del sistema mencionado anteriormente.

3.2 Requerimientos del SIAFEM

El análisis realizado anteriormente permitió identificar y definir los procesos que el SIAFEM realizará, las restricciones, los requerimientos funcionales y no funcionales del SIAFEM se describen de forma detalla en esta sección.

3.2.1 Requerimientos funcionales

Los requisitos funcionales definen que debe de realizar un sistema. Para el SIAFEM se tiene únicamente lo siguiente:

- Generar un menú de contenidos: El sistema deberá de desplegar un menú de las categorías que tendrá el SIAFEM. Se determinaron cinco categorías: Sistemas Expertos, Razonamiento Basado en Casos, Casos Históricos, Apuntes, Videos y presentaciones. Esto permitirá al usuario conocer y seleccionar de acuerdo a su interés lo que el SIAFEM provee.

3.2.2 Requerimientos no funcionales

Los requisitos no funcionales definen como debe ser el sistema. El SIAFEM deberá de tener las siguientes cualidades:

- Disponibilidad: deberá de estar en funcionamiento y ser capaz de proporcionar los servicios solicitados, por lo que deberá de evitar interrupciones a causa del corte de energía eléctrica, fallas de hardware o actualizaciones del sistema.
- Apariencia: la presentación deberá de ser homogénea para cada uno de los sistemas o herramientas que lo conforman.
- Concurrencia: Deberá de soportar un gran número de usuarios accediendo a sus recursos.

3.2.3 Restricciones

Las restricciones que tendrá el SIAFEM son:

- No generará reportes en formato PDF.
- Se alojará en una plataforma Linux.
- Deberá de trabajar en los navegadores web (Internet Explorer 8.0 o superior, Chrome, Firefox, Safari, Opera).

3.3 Diseño del SIAFEM

Con base en el análisis realizado anteriormente y en los requerimientos funcionales y no funcionales, se muestra en la figura 3.5 de manera general el diagrama del SIAFEM.

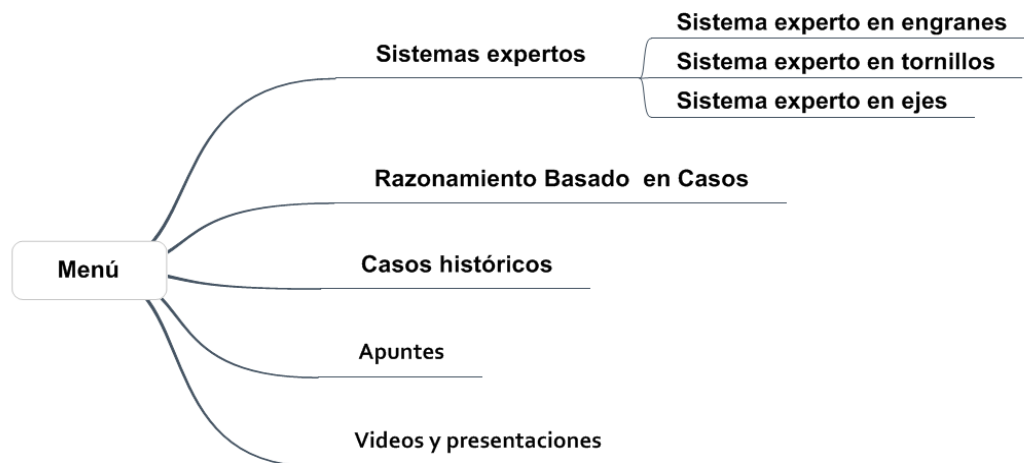


Figura 3.5 Diagrama del SIAFEM

Se desea que el sistema este dividido de la siguiente manera:

- Un menú que permita visualizar el contenido.
- La distribución de los sistemas y recursos en categorías.
- El módulo de apuntes deberá de contar con otro menú para que el usuario pueda acceder a los archivos PDF relacionados con el análisis de falla.

Este diagrama servirá como base para el desarrollo del sistema y será el mapa de navegación que seguirá el usuario para acceder a los diferentes sistemas y recursos del SIAFEM.

Capítulo 4

Desarrollo

Desarrollo

En este capítulo se describen las herramientas que serán empleadas para el desarrollo del SIAFEM y se describen las actividades realizadas para su implementación, estas actividades se sustentan en el análisis realizado en el capítulo anterior.

4.1 Herramientas de desarrollo

Actualmente para el desarrollo de una aplicación web se emplean *frameworks*, bibliotecas y herramientas que disminuyen considerablemente el tiempo de desarrollo y facilitan el trabajo.

4.1.1 NetBeans IDE

NetBeans IDE (*Integrated Development Environment*) es un entorno de desarrollo integrado, libre y de código abierto para el desarrollo de aplicaciones de escritorio, web o móviles basadas en el lenguaje de programación Java. El IDE también ofrece un conjunto de herramientas para el desarrollo de aplicaciones en PHP, C/C++, HTML5 con HTML, JavaScript y CSS.

4.1.2 Maven

Maven es un *framework* de código abierto que proporciona una infraestructura para la construcción y gestión de proyectos basados en Java. Define la configuración del proyecto a través de un archivo central denominado POM (*Project Object Model*), `pom.xml`, donde se describe el proyecto de software a desarrollar, las dependencias, repositorios y *plugins* necesarios para el

despliegue, integración, empaquetado, compilación, documentación, validación y pruebas del proyecto.

4.1.3 PrimeFaces

PrimeFaces es una biblioteca de código abierto para aplicaciones web basadas en JavaServer Faces (JSF). Proporciona una extensa variedad de componentes visuales con sólo un archivo JAR, algunas de las características principales son:

- Es ligero, no requiere configuración ni dependencias.
- Provee un kit para el desarrollo de aplicaciones móviles.
- Soporte para AJAX, basándose en la API AJAX JSF 2.0.
- Compatibilidad con otras bibliotecas de componentes visuales como JBoss y RichFaces.
- Uso de JavaScript no intrusivo.
- Posee una documentación extensa.
- Múltiples temas de apariencia prediseñados.
- Soporte a los diferentes navegadores (Internet Explorer 8.0 o superior, Chrome, Firefox, Safari, Opera).

4.1.4 Java EE

Java EE (*Enterprise Edition*) proporciona una plataforma basada en estándares para el desarrollo de aplicaciones web y empresariales distribuidas en una arquitectura multinivel que ofrece disponibilidad, facilidad de mantenimiento, escalabilidad, portabilidad, seguridad. Se definen cuatro tipos de componentes dentro de la plataforma Java EE, estos componentes son:

- *Applets*: son aplicaciones *GUI*⁹ que se ejecutan en un navegador web.
- Aplicaciones cliente: son programas *GUI* que se ejecutan en el cliente, pueden utilizar servicios Java EE para el acceso a componentes situados en otro nivel.
- Aplicaciones web: *servlets*, páginas JSP (*JavaServer Pages*) y JSF (*JavaServer Faces*), se ejecutan en un contenedor web y responden a las peticiones HTTP del cliente.
- Aplicaciones empresariales: EJBs (*Enterprise Java Beans*), JSM (*Java Service Message*), JTA (*Java Transaction API*), etc. Se ejecutan en un contenedor EJB. *Enterprise Java Beans* son componentes de negocio que se ejecutan en el servidor.

La plataforma Java EE se divide en dominios lógicos llamados contenedores. Cada contenedor tiene una función específica, soporta un conjunto de APIs¹⁰ y ofrece servicios a los componentes (seguridad, acceso a bases de datos, gestión de transacciones, nombres de directorios e inyección de recursos).

La arquitectura de una aplicación Java EE basada en el MVC define la separación de los componentes desarrollados en varias capas, generalmente se consideran hasta cinco capas. Estas capas son:

- La capa cliente procesa la información recibida por medio de un navegador web, permite mostrar la información generada por el sistema a través de lenguaje de marcado (HTML, XML, etc.).

⁹ *Graphical User Interface* (Interfaz Gráfica de Usuario)

¹⁰ *Application Programming Interface* (Interfaz de programación de aplicación)

- La capa de presentación incluye los elementos de la interfaz de usuario de la aplicación, hace posible la interacción de la aplicación con el usuario.
- La capa lógica de negocio procesa las peticiones del usuario procedentes de la capa de presentación. Se implementa la lógica de la aplicación.
- La capa de integración gestiona los recursos empresariales del sistema, permite al sistema interactuar con otros sistemas externos para el intercambio de datos.
- La capa de acceso a datos requiere alguna forma de persistencia para acceder a recursos externos. Incluye una base de datos relacional que se utiliza para leer y almacenar datos.

4.1.5 JESS

JESS (*Java Expert System Shell*) es un motor de reglas desarrollado por *Ernest Friedman-Hill* en *Sandia National Laboratories*, el tipo de licencia es prueba, comercial y gratuita bajo petición para universidades e instituciones académicas. JESS tuvo su origen en CLIPS y fue desarrollado en Java, facilitando la integración de JESS con cualquier aplicación basada en Java como *servlets*, EJBs (*Enterprise JavaBeans*), *applets* y otros sistemas. Para el desarrollo de un sistema experto basado en reglas podemos hacer uso de la línea de comandos de JESS y acceder a la API de Java o importar la biblioteca de JESS a partir de código Java. JESS usa el algoritmo RETE para asociar las reglas con la base de conocimiento

(*pattern matching*), este algoritmo es rápido y eficiente para procesar grandes pilas y hechos en poco tiempo.

4.1.6 Tomcat

Apache Tomcat es un servidor HTTP y contenedor de aplicaciones web basadas en Java como son los *servlets* y *Java Server Pages* (JSP). Tomcat fue desarrollado bajo el proyecto Apache-Jakarta y es un software de código abierto. Un contenedor de *servlets* es un *Shell* de ejecución que maneja e invoca *servlets* por cuenta del usuario y un *servlet* es un objeto java que el servidor web carga para manejar peticiones del cliente, es decir, son pequeños programas escritos en Java que admiten peticiones a través del protocolo HTTP. Los *servlets* reciben peticiones desde un navegador web, las procesan y devuelven una respuesta al navegador web, normalmente en HTML.

4.2 Estructuración y estrategia para el desarrollo del SIAFEM

Las herramientas tecnológicas para el desarrollo de la aplicación web fueron seleccionadas con base en lo siguiente:

- Maven, Java EE, PrimeFaces, Spring, Hibernate y PostgreSQL son herramientas que forman parte del desarrollo de los sistemas de *Casos históricos* y del sistema de *Videos y presentaciones*. El uso de estas herramientas facilitará que estos dos sistemas formen parte de la aplicación web final.
- Se usa JESS como una alternativa de CLIPS, debido a que el *plugin* CLIPSJNI (usado en los sistemas expertos de engranes y tornillos)

necesita que se especifique la ruta del archivo DLL que permite la interacción del *Shell* de CLIPS con Java. Además, CLIPSJNI no proporciona el archivo de biblioteca compartida (*Shared Object*¹¹) que permite usar el *Shell* de CLIPS en la plataforma Linux, para esto hay que generar el archivo y moverlo a un directorio indicado por *java.library.path*.

Con base en los requerimientos y el análisis realizado en el capítulo anterior sobre cada uno de los sistemas que formarán parte del SIAFEM se presenta en la figura 4.1 la arquitectura del sistema.

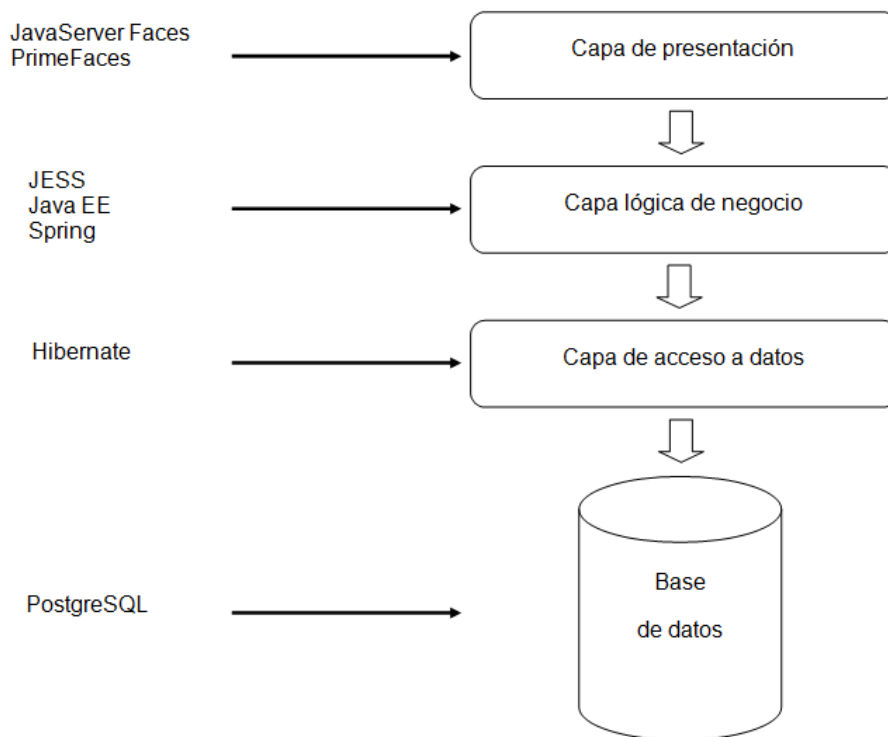


Figura 4.1 Arquitectura del SIAFEM

¹¹ Lleva el sufijo *.so*, son bibliotecas compartidas que usan los programas para invocar funciones o código en tiempo de ejecución. Es el equivalente de *.dll* en la plataforma Windows.

La solución propuesta para el desarrollo del SIAFEM es la siguiente:

- Dividir los recursos que proporcionará el sistema en categorías por medio de un menú, tomando como referencia el diseño propuesto en el capítulo anterior.
- Migrar los sistemas expertos en análisis de falla de engranes y tornillos, pues se requiere que se encuentren en un entorno web, para lograr esta tarea se tiene que reimplementar el código fuente sin cambiar su funcionalidad.
- Integrar a la aplicación final el sistema de Casos históricos y el sistema de Videos y presentaciones.
- Proveer un menú y un visor de archivos PDF para la consulta de los documentos (apuntes sobre *análisis de falla*).

4.3 Creación de la estructura de la aplicación web

Se crea una aplicación web usando el *framework Maven*, la estructura del proyecto generado en NetBeans IDE se muestra en la figura 4.2.

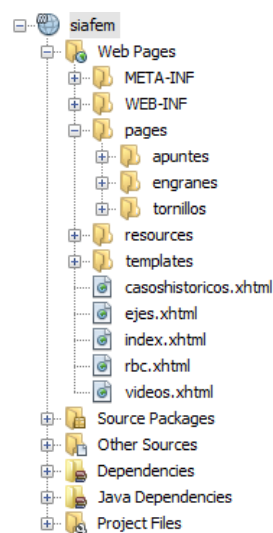


Figura 4.2. Estructura de la aplicación web.

La distribución de los directorios se agrupa de la siguiente manera:

- **META-INF:** se ubican los ficheros de metadatos, por ejemplo el archivo `persistence.xml` que permite a la aplicación acceder a la base de datos.
- **WEB-INF:** contiene recursos relacionados con la aplicación web. En este directorio se tiene el archivo `web.xml`, este archivo establece la configuración de la aplicación web.
- **pages:** contiene las carpetas de apuntes, engranes y tornillos. En estas carpetas se tiene la división de las páginas JSF con base en la función de los recursos que ofrece (Sistema experto en engranes, Sistema experto en tornillos y apuntes sobre el análisis de falla en archivos PDF). Esto permite tener una aplicación web con una estructura clara y organizada.
- **resources:** contiene los recursos de la aplicación web divididas en carpetas, donde se tienen imágenes, archivos PDF, código JavaScript, hojas de estilo (CSS) y la base de conocimiento de los sistemas expertos (archivos `clp`).
- **templates:** contiene las plantillas de la aplicación web, permitiendo implementar interfaces de usuario homogéneas de una manera fácil, rápida y reusable.
- **Source Packages:** contiene la lógica de negocio de la aplicación web, se tienen archivos Java.
- **Other Sources:** se tienen archivos `.properties` que están asociados a las plantillas del sistema.
- **Dependencies y Java Dependecies** contiene las bibliotecas que serán utilizadas por la aplicación web.

- *Project Files*: contiene el archivo pom.xml con la información del proyecto, los *plugins* y dependencias utilizadas en la aplicación.

4.3.1 Creación de las vistas de usuario

Partiendo de la solución propuesta se procede a generar el menú del SIAFEM, esta interfaz de usuario será la página de inicio de la aplicación web. Los *frameworks* empleados para la generación de la interfaz gráfica de usuario son: *JavaServer Faces* y *PrimeFaces*.



Figura 4.3 Página de inicio del SIAFEM

Las herramientas *JavaServer Faces* y *PrimeFaces* facilitaron el desarrollo de la vista del menú del sistema. Siguiendo con la solución propuesta, se desarrollan las demás interfaces de usuario de la aplicación web en el siguiente orden:

- Sistema experto en engranes.
- Sistema experto en tornillos.
- Apuntes.

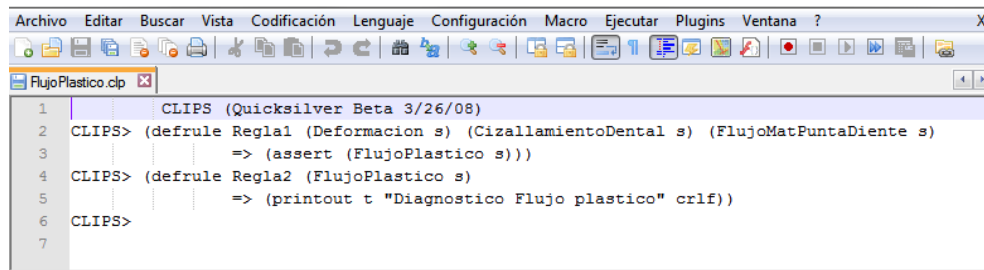
En el Anexo C se podrán visualizar las interfaces de usuario para cada uno de los puntos mencionados.

4.3.2 Generación de la lógica de negocio

La funcionalidad de los sistemas expertos en engranes y tornillos se desarrollo en la capa lógica de negocio de la aplicación web. Para realizar la migración de estos sistemas a un entorno web, fue necesario hacer un análisis del código fuente para comprender su funcionalidad. A continuación se describe el trabajo realizado para ambos sistemas expertos:

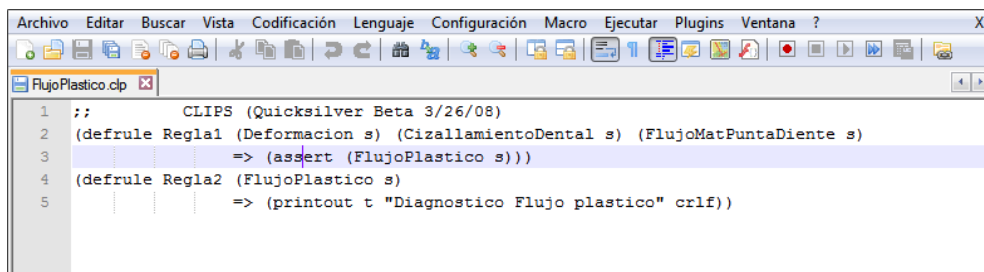
- Creación de las clases *Screw* y *Gear* para la manipulación de los componentes de la interfaz de usuario (*beans*).
- Se emplea la lógica para controlar los eventos realizados en la interfaz de usuario.
- Se procesan los eventos realizados por el usuario, se busca en las reglas de la base de conocimientos y se da un diagnóstico de la posible causa de falla del elemento mecánico.
- Las reglas de la base de conocimientos de los sistemas expertos no sufrió alguna modificación en cada uno de los archivos, lo único que se realizó fue comentar la primera línea y quitar de la sintaxis "CLIPS>", estos cambios permitieron al motor de inferencias de JESS trabajar con cada una de las reglas.

Comparando las figuras 4.4 y 4.5, se observa el cambio realizado en el archivo que contiene las reglas de la base de conocimientos para determinar la falla de flujo plástico en el sistema experto de engranes, esta acción se repitió para cada uno de los archivos de ambos sistemas expertos.



```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Macro  Ejecutar  Plugins  Ventana  ? X
FlujoPlastico.clp
1  CLIPS (Quicksilver Beta 3/26/08)
2  CLIPS> (defrule Regla1 (Deformacion s) (CizallamientoDental s) (FlujoMatPuntaDiente s)
3          => (assert (FlujoPlastico s)))
4  CLIPS> (defrule Regla2 (FlujoPlastico s)
5          => (printout t "Diagnostico Flujo plastico" crlf))
6  CLIPS>
7
```

Figura 4.4 Reglas de la base de conocimientos para CLIPS.



```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Macro  Ejecutar  Plugins  Ventana  ? X
FlujoPlastico.clp
1  ;;          CLIPS (Quicksilver Beta 3/26/08)
2  (defrule Regla1 (Deformacion s) (CizallamientoDental s) (FlujoMatPuntaDiente s)
3          => (assert (FlujoPlastico s)))
4  (defrule Regla2 (FlujoPlastico s)
5          => (printout t "Diagnostico Flujo plastico" crlf))
```

Figura 4.5 Reglas de la base de conocimientos para JESS.

Capítulo 5

Verificación y Validación

Verificación y Validación

La Validación y Verificación dentro de la Ingeniería del Software es el nombre dado a las actividades realizadas en cada etapa del proceso de software, se realizan procesos de análisis y pruebas del producto.

La validación tiene como objetivo determinar si el sistema satisface en su totalidad las especificaciones del cliente, responde a la pregunta: ¿Se está realizando el producto correcto? Mientras la verificación tiene como objetivo determinar si el sistema cumple con los requerimientos establecidos (funcionales y no funcionales), responde a la pregunta: ¿Se está construyendo el sistema correctamente?

El modelo de desarrollo del SIAFEM fue en espiral, los prototipos fueron verificados y validados por el cliente en cada entrega, estas actividades permitieron al sistema seguir evolucionando hasta la versión final.

5.1 Pruebas

Las pruebas de un sistema de software consisten en un conjunto de actividades donde se verifica de una manera objetiva que el producto final cumpla con los requerimientos establecidos por el cliente, ayudan a identificar y modificar posibles errores para el correcto funcionamiento de la aplicación dando como resultado un software de calidad y libre de errores. A continuación se describen las pruebas realizadas al SIAFEM.

5.1.1 Pruebas de caja blanca

Las pruebas de caja blanca o estructurales se basan en conocer el código fuente de la aplicación web, se busca garantizar por medio de diversos valores de entrada que toda lógica de negocio se ejecute al menos una vez.

Al realizar la migración de un entorno cliente a un servidor se examinó el flujo de datos a través del sistema de software, considerando lo siguiente:

- Las dependencias existentes entre los diferentes recursos del sistema, específicamente los datos proporcionados por el usuario (base de hechos) y la base de conocimientos.
- Conocidos los resultados generados por las aplicaciones cliente con base en los valores de entrada, se deben de comparar con los resultados obtenidos de la aplicación web con respecto a los mismos valores utilizados, es decir, los resultados obtenidos de la aplicación web con los resultados de las aplicaciones cliente deben ser iguales.

Las pruebas se realizaron sobre cada una de las rutas de navegación de la aplicación web, permitiendo trabajar con partes definidas de la lógica de negocio. Esta acción facilitó manejar e identificar las rutinas de excepción y de error de cada proceso relacionado con los valores de entrada. En conclusión de las pruebas de caja blanca se tiene:

- En la parte correspondiente al sistema experto en engranes se encontró con errores de ejecución, debido a la relación de la base de conocimientos con respecto a los parámetros seleccionados por el usuario (base de hechos). La información proporcionada por el usuario no era suficiente para

coincidir con la base de conocimientos, provocando que no existiera una respuesta del sistema experto, el problema fue corregido.

- En el sistema experto en tornillos se detectó un problema parecido al anterior, el sistema experto no respondía ante los parámetros proporcionados por el usuario. Para solucionar este error fue necesario consultar la base de conocimientos y se encontró que no había sido considerado anteriormente, el error fue corregido.
- Realizadas las correcciones mencionadas previamente, se asegura que todos los métodos empleados en la lógica de negocio se ejecutan al menos una vez.
- Permitió dividir la lógica de negocio en dos partes, la primera responderá a las acciones realizadas por el usuario en la interfaz gráfica. La segunda evaluará y dará respuesta del tipo de falla presente en la pieza con base en los elementos seleccionados por el usuario. Esto permite tener una estructura clara y definida de la lógica de negocio del proyecto.

5.1.2 Pruebas de caja negra

Las pruebas de caja negra o funcionales se basa en conocer los datos de entrada y sus correspondientes salidas sin importar la lógica de negocio. Las pruebas fueron realizadas a través de un navegador web y para llevar a cabo estas pruebas se tomó en cuenta:

- Validar los datos proporcionados por el usuario, en caso de no ser válidos, se despliegan mensajes de error.

- El diseño y el funcionamiento de los componentes de cada formulario de las aplicaciones cliente.
- Los posibles mensajes que debe mostrar el sistema ante las acciones realizadas por el usuario.
- El flujo de navegación para acceder a las diferentes vistas del sistema y recursos que proporciona la aplicación web.

Al igual que las pruebas de caja blanca, las pruebas de caja negra se realizaron en cada ruta de navegación, trabajando sobre la interfaz gráfica de usuario y sobre cada uno de los componentes de los formularios del SIAFEM; donde se proporcionaron datos de entrada y se estudiaron las salidas, analizando los resultados. Obteniendo las siguientes conclusiones:

- El uso de partición de equivalencias para determinar estados válidos y no válidos, permitió validar los datos proporcionados por el usuario en los campos del primer formulario para los sistemas expertos en engranes y en tornillos, condicionando el flujo de navegación a través del SIAFEM. En caso de existir un campo vacío o de ingresar un dato diferente al esperado, el sistema mostrará los mensajes de error correspondientes a cada campo. Una vez validados los campos, el sistema permite continuar con los siguientes formularios correspondientes a cada sistema experto.
- Para emular la funcionalidad de las aplicaciones cliente en un entorno web en los diferentes formularios, particularmente el sistema experto en tornillos, fue necesario conocer el comportamiento del sistema ante las acciones realizadas en la interfaz gráfica. Donde se tuvo el despliegue de diversas

ventanas (formularios) como respuesta a la selección de dos o más elementos gráficos (*checkbox*). Fue necesario corregir este problema debido a la cantidad de ventanas desplegadas al usuario, siendo necesario limitar al usuario en la selección de parámetros a través del componente *radioButton* en el entorno web, teniendo como resultado un flujo de navegación controlada, clara y definida a los diversos formularios del sistema experto. En los últimos formularios no se realizaron cambios en los componentes gráficos, sólo fue necesario implementar código para controlar las acciones realizadas por el usuario en la selección de cada *checkbox*, igualando el comportamiento del entorno web con el de la aplicación cliente.

- Debido a la gran cantidad de páginas JSF creadas para acceder a los apuntes y a los sistemas expertos fue necesario crear subdirectorios para reubicar las páginas con base en función a los recursos que ofrece. Esta acción permitió tener una estructura clara y organizada de la aplicación web, ver figura 5.1.
- Con base en el punto anterior, se realizaron cambios en las páginas JSF para referenciar nuevamente a las imágenes, las hojas de estilo (CSS) y el flujo de navegación entre las vistas de usuario. Realizado los cambios, se procedió a verificar el funcionamiento de navegación entre las diferentes interfaces de acuerdo al flujo de la lógica de negocio, comprobando que no hubo errores en los mecanismos navegación y en la presentación de la interfaz de usuario.

Como se puede apreciar en la figura 5.1, se tiene una estructura de directorios clara y limpia. La organización de los diversos recursos de la aplicación web permitirá añadir nuevos módulos o funcionalidades al sistema, además de facilitar el mantenimiento de la aplicación web.

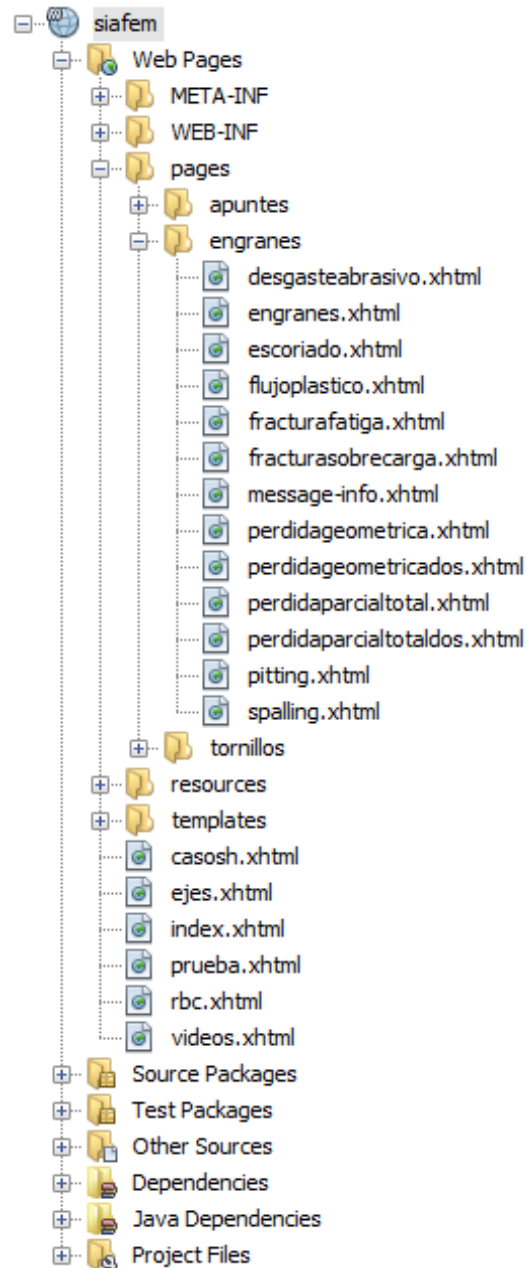


Figura 5.1 Estructura de los directorios de la aplicación web.

5.1.3 Pruebas de integración

Las pruebas de integración se realizaron para comprobar la correcta interacción de las diversas tecnologías empleadas en el desarrollo de los diversos componentes del SIAFEM.

Estas pruebas de integración permitieron corregir y detectar errores de funcionamiento al integrar las diferentes tecnologías en cada uno de los módulos de la aplicación web, particularmente entre Java EE y JESS. De las pruebas de integración se concluye:

- El desarrollo de la aplicación web a través de Maven fue sencilla, permitió integrar y gestionar las dependencias del proyecto de una manera menos compleja.
- La facilidad de integrar a JESS en el entorno web debido a que se obtuvo la ruta real del SIAFEM en el servidor web, permitió interactuar de manera exitosa la base de conocimientos con la aplicación web.

5.1.4 Pruebas de compatibilidad

Para garantizar el correcto funcionamiento del SIAFEM, se realizaron pruebas de compatibilidad de la aplicación web en los navegadores más utilizados como son: *Internet Explorer, Mozilla Firefox, Google Chrome, Opera y Safari*. El resultado de estas pruebas fue satisfactorio, no así en navegadores web que vienen por defecto en algunas distribuciones de Linux como son: OpenSUSE 13.2 y CentOS 7, en el cuál se observaron algunos cambios en las vistas de usuario y se tuvieron retrasos de ejecución en los componentes de los formularios.

5.1.5 Pruebas de usabilidad

Estas pruebas se realizaron para determinar si el diseño del SIAFEM es fácil de usar y de aprender por el usuario final, lo que permitió identificar las áreas de diseño del sistema de difícil uso para el usuario. La importancia de realizar este tipo de pruebas fue para:

- Descubrir la facilidad con la que se puede utilizar eficientemente la aplicación web.
- Facilitar la navegación entre los diferentes recursos que el sistema provee.
- Proveer un ambiente amigable a la vista, evitando la sobrecarga de información.
- Organizar apropiadamente la presentación de la interfaz gráfica por medio de paneles, encabezados y hojas de estilo.

La importancia de realizar estas pruebas fue para optimizar la facilidad de uso y de aprendizaje del usuario a través de la interfaz gráfica del SIAFEM y para proporcionar un entorno agradable que ayude al usuario a entender la información presentada durante la navegación hacia todas las secciones de la aplicación web.

Conclusiones

Conclusiones

Para el desarrollo del sistema web presentado en este trabajo, se puso en práctica las habilidades, aptitudes y conocimientos adquiridos. Obteniendo como resultado una herramienta tecnológica que permite diagnosticar las causas de fallas presentes en elementos mecánicos (ejes, engranes, tornillos, entre otros).

Durante el proceso de análisis para implementar una solución, se observó la importancia que tiene la Ingeniería de Software dentro del proceso de desarrollo del software, pues se presentaron complicaciones con los sistemas disponibles para formar parte de la aplicación web (SIAFEM), dificultando el proceso de análisis. Algunos de estos problemas fueron:

- Falta de documentación.
- Malas prácticas de desarrollo de software, afectando así de forma negativa a la calidad del software.
- La tecnología que fue empleada en el desarrollo de algunos sistemas actualmente es obsoleta.
- Las personas que desarrollaron estos sistemas demostraron de manera fehaciente su falta de conocimientos relativos con relación a las metodologías para la implementación de software.

Fue necesario realizar reingeniería de software para examinar los sistemas existentes y emplear metodologías para conocer la arquitectura, la estructura de

los datos, el lenguaje de programación empleado, las dependencias con otras tecnologías de desarrollo y la lógica de negocio. Las actividades de reingeniería de software empleados fueron:

- Ingeniería inversa.
- Reestructuración.
- Migración y reimplementación del código fuente.

La ingeniería inversa sirvió para identificar los componentes de cada uno de los sistemas, sus respectivas interrelaciones, además de comprender su funcionamiento y tener una visión más amplia del problema que se estaba intentando resolver. Lo que llevó a excluir de la aplicación web al sistema de razonamiento basado en casos y el sistema experto en ejes debido a factores como:

- La curva de aprendizaje en *Visual Basic 6.0* y *Visual Rule Studio*
- Se requiere migrar de la base de conocimientos del sistema de razonamiento basado en casos alojado en *Microsoft Access '97* a otro sistema de gestión de base de datos
- No se dispone de la herramienta de *Visual Rule Studio* para poder ejecutar el sistema de ejes, impidiendo conocer su diseño y funcionamiento.

La reestructuración ayudó a simplificar la lógica de negocio y a tener una mayor legibilidad de los sistemas expertos en engranes y tornillos, para ello fue necesario analizar el código fuente.

Para llevar a cabo la migración de los sistemas expertos en engranes y tornillos a un entorno web fue necesario reimplementar el código fuente Java SE (*Standard Edition*) a Java EE (*Enterprise Edition*). Además de usar el patrón de arquitectura de software MVC (Modelo-Vista-Controlador) para separar la lógica de negocio con respecto a las vistas de usuario creadas con las tecnologías *JavaServer Faces* (JSF) y *PrimeFaces*. El uso de *Maven* para gestionar las dependencias, repositorios y *plugins* utilizados en la creación del proyecto en *NetBeans* y usar a JESS como el motor de inferencias para trabajar con las reglas de la base de conocimientos.

Al final se lograron superar los obstáculos presentados durante la etapa de análisis, se establecieron las posibles soluciones para el desarrollo de la aplicación web. La selección de tecnologías como *Tomcat*, *Maven*, *JavaServer Faces*, *PrimeFaces*, Java EE, *NetBeans* y JESS permitieron desarrollar un sistema web fiable, flexible, escalable, portable y de fácil mantenimiento.

Se cumplió con los objetivos planteados al comienzo de este proyecto y en la práctica se tiene un sistema de software que sirve a través de *internet* como una herramienta de apoyo para un especialista de vasta experiencia en el análisis de falla o se emplea en la formación de nuevos especialistas.

Recomendaciones

En la actualidad la tecnología que fue empleada en el desarrollo del sistema de Razonamiento Basado en Casos y del sistema experto en ejes es obsoleta. Para incorporar estos sistemas en el trabajo final, se requiere llevar a cabo lo siguiente:

- Seleccionar una metodología de Ingeniería de Software que sea adecuada para realizar la migración de estos sistemas a un entorno web.
- Analizar el código fuente de cada uno de los sistemas, con el objetivo de identificar las funcionalidades de los componentes, identificar la estructura de los datos y el diseño de la arquitectura empleada.
- Una vez identificadas las funcionalidades de los componentes, será necesario realizar una breve documentación sobre el proceso lógico.
- Diseñar y generar posibles soluciones que permitan la migración de estos sistemas.
- Seleccionar herramientas tecnológicas que permitan generar un nuevo sistema con las siguientes características: fácil mantenimiento, portabilidad, escalabilidad y confiabilidad.
- Diseñar y ejecutar pruebas que permitan verificar y validar el producto final.

Referencias

Referencias

- Cardador, A. (2014). *Implantación de aplicaciones web en entornos internet, intranet y extranet* (cap. 3). Málaga: IC Editorial. Extraído el 3 de febrero, 2016, de <https://books.google.com.mx/books?id=Lj91CQAAQBAJ>.
- Castillo, E., Gutiérrez, J. y Hadi, Ali. (1998). Introducción. En *Sistemas Expertos y Modelos de Redes Probabilísticas. Monografías de la Academia Española de Ingeniería* (pp. 1-14). Madrid: Academia de Ingeniería. Extraído el 16 de junio, 2016, de <http://personales.unican.es/gutierjm/BookCGH.html>.
- CLIPS. [En línea] Accedido el 15 de febrero, 2017. <http://www.clipsrules.net/>.
- CLIPSJNI. [En línea] Accedido el 4 de noviembre, 2015. <http://www.clipsrules.net/?q=Downloads/CLIPSJNI>.
- Díaz, E. (2014). *Sistema Experto prototipo para análisis de fallas en engranes*. Tesis de licenciatura. Universidad Nacional Autónoma de México.
- Friedman-Hill, E. (2003). *JESS in Action: Rule-Based Systems in Java*. Estados Unidos de América: Manning Publications.
- García, J. M. (2005). Ingeniería del Software. [En línea] 29 de mayo de 2005. Extraído el 27 de junio, 2016, de http://www.falconmarbella.com/esigranada/dmddocuments/Tema_8.pdf.
- Giarratano, J. y Riley, G. (1998). Introduction to Expert Systems. En *Expert Systems: Principles and Programming* (3ª ed, pp.1-4). Boston: Massachusetts.
- Goncalves, Antonio (2010). Java EE 6 at a Glance. En *Beginning Java™ EE 6 Platform with Glassfish™ 3* (2ª ed., pp. 1-9). Estados Unidos de América: Apress.
- Hernández, A. (2003). *Razonamiento Basado en Casos aplicado al análisis de falla de elementos mecánicos metálicos*. Tesis de licenciatura. Universidad Nacional Autónoma de México.
- Jacobo, V. (2005). *Sistema Experto para análisis de falla de elementos mecánicos metálicos*. Tesis de doctorado. Universidad Nacional Autónoma de México.
- Jacobo, V., Ortiz, A. y Schouwenaars, R. (2007). "SEAFEM: Herramienta computacional para la determinación de causas de falla en elementos mecánicos metálicos". *Ingeniería: Investigación y Tecnología*, 4(8), 261-263. Extraído el 7 de febrero, 2016, de <http://www.journals.unam.mx/index.php/ingenieria/article/view/13479>.

-
- JESS. [En línea] Accedido el 24 de enero, 2015. <http://www.jessrules.com/jess/download.shtml>.
- Keyes, J. (1999). Appendix. En *Banking Technology Handbook* (p. A-6). Estados Unidos de América: CRC Press. Extraído el 30 de junio, 2016, de <https://books.google.com.mx/books?id=RnS-tfF1pwQC>.
- Martínez, G. (2014). *Sistema Experto prototipo para el análisis de fallas en tornillos ferrosos*. Tesis de licenciatura. Universidad Nacional Autónoma de México.
- Maven. [En línea] Accedido el 12 de noviembre, 2014. <https://maven.apache.org/>.
- Mukhar, K. Zelenak, C. Weaver, J. L. y Crume, J. (2006). Java EE Essentials. En *Beginning Java EE 5. From Novice to Professional* (pp. 1-28). Estados Unidos de América: Apress.
- NetBeans IDE. [En línea] Accedido el 12 de noviembre, 2014. <https://netbeans.org/>.
- Pressman, R. S. (2010). *Ingeniería del Software. Un enfoque práctico* (7ª ed.). México: McGraw-Hill.
- PrimeFaces. [En línea] Accedido el 12 de noviembre, 2014. <http://www.primefaces.org/>.
- Rumbaugh, J., Jacobson, I. y Booch, G. (2000). La naturaleza y propósito de los modelos. En *El lenguaje unificado de modelado. Manual de referencia* (pp. 11-16). Madrid: Addison-Wesley.
- Siler, B. y Spotts, J. (1999). Programación con Visual Basic. En *Visual Basic 6 Edición Especial* (pp. 3-5). España: Prentice Hall.
- Sommerville, I. (2005). *Ingeniería del software* (7ª ed.). Madrid: Pearson Educación.
- Talledo, J. (2015). *MF0493_3 Implantación de aplicaciones web en entornos internet, intranet y extranet* (cap. 4). España: Paraninfo. Extraído el 3 de febrero, 2016, de <https://books.google.com.mx/books?id=RtESCgAAQBAJ>.
- Tomcat. [En línea] Accedido el 23 de febrero, 2016. <http://tomcat.apache.org/>.
- Varanasi, B. y Belida, S. (2014). Getting Started with Maven. En *Introducing Maven* (pp. 1-3). Estados Unidos de América: Apress. Extraído el 18 de febrero, 2016, de <https://books.google.com.mx/books?id=a2MnCgAAQBAJ>.
- Yener, M. y Theedom, A. (2015). Model View Controller Pattern. En *Professional Java EE Design Patterns* (pp. 183-186). Canadá: John Wiley & Sons.

Anexos

Anexo A: Marco teórico de las aplicaciones cliente

A continuación se exponen algunos conceptos teóricos relacionados con las herramientas que fueron usadas en el desarrollo de los sistemas analizados en el capítulo 3.

CLIPS

CLIPS (C Language Integrated Production System) fue desarrollado por Gary Riley en el centro espacial Johnson de la NASA de 1985 a 1996. Es una herramienta para el desarrollo de sistemas expertos, soporta tres estilos de programación: procedural, orientado a objetos y basado en reglas. Desde 1996, CLIPS está disponible como software de dominio público.

CLIPSJNI

CLIPS Java Native Interface (CLIPSJNI) es una biblioteca escrita en Java que permite interactuar con el motor de inferencias de CLIPS, facilitando la integración de este software con cualquier aplicación Java.

Visual Basic

Visual Basic es descendiente de BASIC (*Beginner's All-purpose Symbolic Instruction Code*) y es un producto de Microsoft. Con el advenimiento de la plataforma Windows, Microsoft desarrollo Visual Basic como un sistema de programación para escribir aplicaciones. Una de las características sobresalientes

fue la de crear aplicaciones solidas en un espacio de tiempo muy corto. Visual Basic 6.0 fue la última versión, liberada en 1998 y tuvo soporte hasta el año 2008.

Visual Rule Studio

Fue un producto de *Rule Machine Corp.* Es una herramienta que permite el desarrollo de sistemas expertos, reglas de negocio y aplicaciones de gestión de conocimientos. *Visual Rule Studio* se instala como una parte integral de *Visual Basic* de Microsoft, apareciendo como un control *Active X*. El motor de inferencia de *Visual Rule Studio* ofrece dos motores primarios para resolver problemas relacionados con los sistemas de producción: el motor de encadenamiento hacia adelante y el motor de encadenamiento hacia atrás.

Spring

Es un *framework* ligero de código abierto creado por Rod Johnson, proporciona un amplio soporte de infraestructura para el desarrollo de aplicaciones empresariales Java, está basado en el concepto de Inversión de Control (IoC) y de Programación Orientada a Aspectos (AOP). Cualquier aplicación Java EE puede beneficiarse de Spring en términos de simplicidad, flexibilidad y prueba.

Hibernate

Es un software libre bajo la licencia de GNU y es un *framework* de Mapeo Objeto/Relacional (ORM) para aplicaciones Java, aunque hay versión para la plataforma .NET conocida como NHibernate. El Mapeo Objeto/Relacional (ORM)

se refiere a la técnica de mapear una representación de datos desde un modelo de objeto a un modelo de datos relacionales bajo un esquema basado en SQL. Existen dos maneras de realizar el mapeo en Hibernate, la primera es por medio de archivos declarativos (XML) y la otra es por medio de anotaciones o metadatos en el código fuente.

PostgreSQL

Es un sistema de gestión de bases de datos relacional orientada a objetos de código abierto y está disponible para los diferentes sistemas operativos como son: Windows, Oracle Solaris, IBM-AIX, MAC OS, Unix, Linux Red Hat y otras plataformas basadas en Linux. PostgreSQL fue desarrollado en el Departamento de Ciencias de Berkeley de la Universidad de California y actualmente es desarrollado por un grupo internacional conocido como *PostgreSQL Global Development Group*.

Razonamiento Basado en Casos (RBC)

Es un conjunto de técnicas relacionadas con la Inteligencia Artificial para proporcionar solución a problemas nuevos mediante la adaptación de soluciones que fueron utilizadas satisfactoriamente para resolver problemas anteriores, es decir, resolver problemas a partir de experiencias previas (casos) y adaptando soluciones pasadas para resolver nuevos problemas. El ciclo de RBC consta de cuatro pasos secuenciales:

- Recuperar: selecciona uno o varios casos similares de la base de casos.
- Reutilizar: las soluciones contenidas en esos casos se adaptan de acuerdo a la consulta.
- Revisar: la solución propuesta se verifica, para construir un nuevo caso.
- Retener: el nuevo caso para una posible resolución de problemas futuros.

Anexo B: Capturas de pantalla de los sistemas cliente

En este anexo se muestran las capturas de pantalla realizadas durante la ejecución de los sistemas analizados en el capítulo 3.

Razonamiento Basado en Casos

A continuación se anexan las capturas realizadas al sistema de razonamiento basado en casos al análisis de falla de elementos mecánicos.



Figura B.1 Bienvenida.

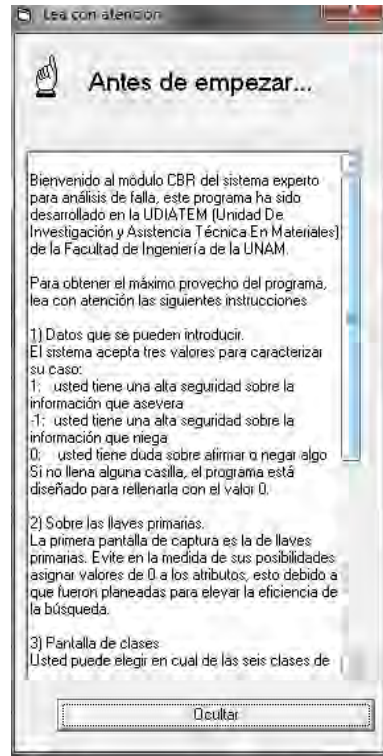


Figura B.2 Instrucciones de uso del sistema de RBC.

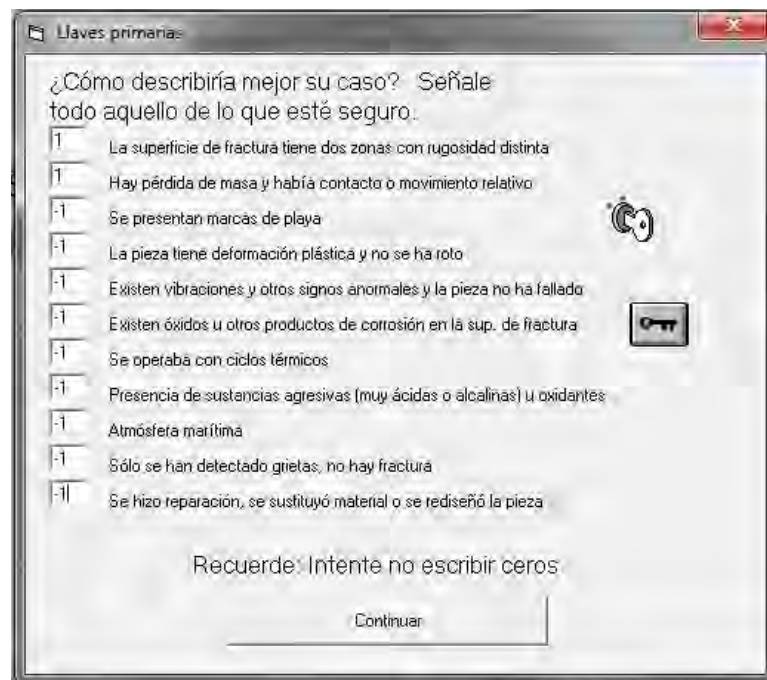


Figura B.3 Llaves primarias.



Figura B.4 Clases.

The 'Información del material 1' window shows a form for material properties. The form has several sections with radio buttons and labels:

- Ferroso / No ferroso:** Radio buttons for '1 Ferroso' and '-1 No ferroso'.
- Fundición / Acero:** Radio buttons for '-1 Fundición' and '1 Acero'.
- Contenido de carbono:** Radio buttons for '1 Menor al 0.25%', '1 Entre 0.25% y 0.55%', '-1 Entre 0.55% y 1%', and '-1 Entre 1% y 5%'.
- Contenido de aleantes:** Radio buttons for '-1 Menor al 0.1%', '1 Entre 0.1% y 1%', '-1 Entre 1% y 5%', '-1 Entre 5% y 11%', and '-1 Mayor al 11% / inoxidable'.

At the bottom, there is a 'Regresar' button and a button with the text 'La información requiere dos páginas. Presione para ir a la página 2'.

Figura B.5 Primera pantalla del formulario de Información del material.

Información del material 2

Aleaciones El material es una aleación de aluminio o magnesio.
 El material es una aleación de zinc.

Tipo de sistema Transmisión de movimiento
 Transmisión de potencia
 Soporte
 Transporte o contención de fluidos.

Tratamiento térmico Recocido
 Normalizado
 Temple y revenido
 Carburizado
 Endurecimiento por trabajo en frío, precipitación, flama

Otros datos Recubrimientos
 Fragilización del material

Ver página 1

Figura B.6 Segunda pantalla del formulario de información del material.

Características de la superficie de fractura / grieta (a simple vista) 1

Apariencia cristalina en una zona
 Patrón de hueso de chevron o espinazo de pescado
 Poca o nula deformación plástica-frágil
 Alta deformación plástica-dúctil
 Cambio de tonalidad en la superficie de la pieza
 Huecos o abolladuras-indentaciones
 Desprendimiento de partículas por abrasión
 Abrasión con ralladuras
 Se encuentra exceso de polvo (productos de corrosión)
 La superficie está deslustrada sin daño aparente
 Presencia de labios
 Apariencia rota o con surcos mostrando la dirección de arranque
 Superficie con deformación en la dirección de rotación
 Superficie de fractura tipo espiral, sin deformación plástica.
 Zonas en forma de creciente brillantes y finas

Regresar

La información requiere dos páginas.
Presione para ir a la página 2.

Figura B.7 Primera pantalla del formulario para la información que se puede obtener a simple vista.

Características de la superficie de fractura / grieta (a simple vista) 2

- Fractura perpendicular al esfuerzo normal máximo
- Fractura a 45° con respecto a la dirección del esfuerzo normal máximo
- Desgaste en las superficies de fractura
- La falla está en un concentrador de esfuerzos mecánico
- Se identifica fácilmente origen, propagación lenta y fractura, hay corrosión
- La fractura comienza en un punto de la periferia
- Grietas adyacentes a la grieta principal
- Porosidad superficial (asociada con abrasión)
- Ampollamiento o delaminación del material
- Origen de falla en el centro de la superficie
- Cuellos en la fractura
- La fractura crece en sentido opuesto a la rotación, pero se desvía 15° o más.
- Tiene regiones a 90° y a 45° con respecto al esfuerzo normal máximo
- Hay variaciones de tonalidad en la superficie de fractura
- La grieta se origina en una zona desgastada
- Marcas de playa simétricas con respecto al origen de la grieta

Ver página 1

Figura B.8 Segunda pantalla del formulario para la información que se puede obtener a simple vista.

Microscopio electrónico 1

- Estrías / marcas de llanta
- Microcavidades equiaxiales
- Ruptura intergranular
- Patrón de río
- Apariencia dúctil
- Ruptura transgranular
- Apariencia frágil
- La microestructura del eje longitudinal muestra fibras dobladas
- Presencia de microcavidades elongadas o parabólicas
- Microcavidades equiaxiales
- Las microcavidades son grandes, asociadas con un esfuerzo de cedencia bajo
- Se inician y se propagan varias grietas simultáneamente
- Los espacios entre las marcas de playa o estrías son difusos y no uniformes
- Líneas de propagación de marcas de fatiga débiles
- Fractura combinada dúctil-frágil (típica de olivaje)

Figura B.9 Primera pantalla para el análisis con microscopio electrónico.



Figura B.10 Segunda pantalla para el análisis con microscopio electrónico.

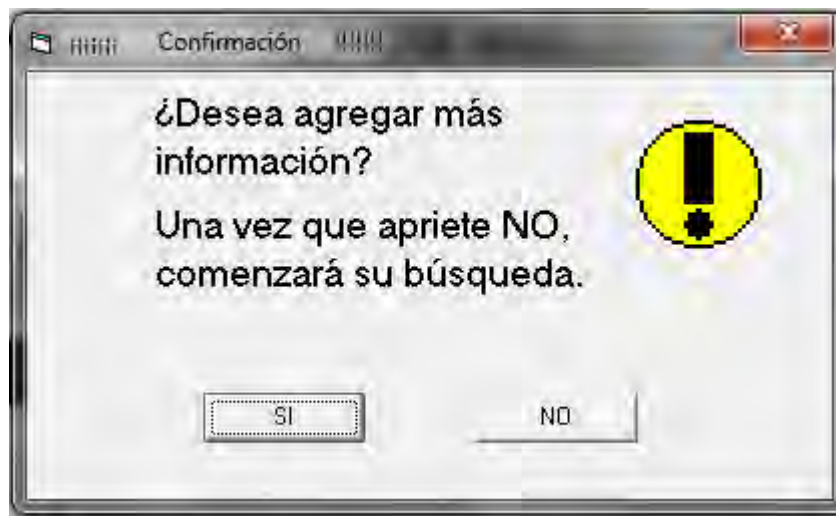


Figura B.11 Pantalla de confirmación.

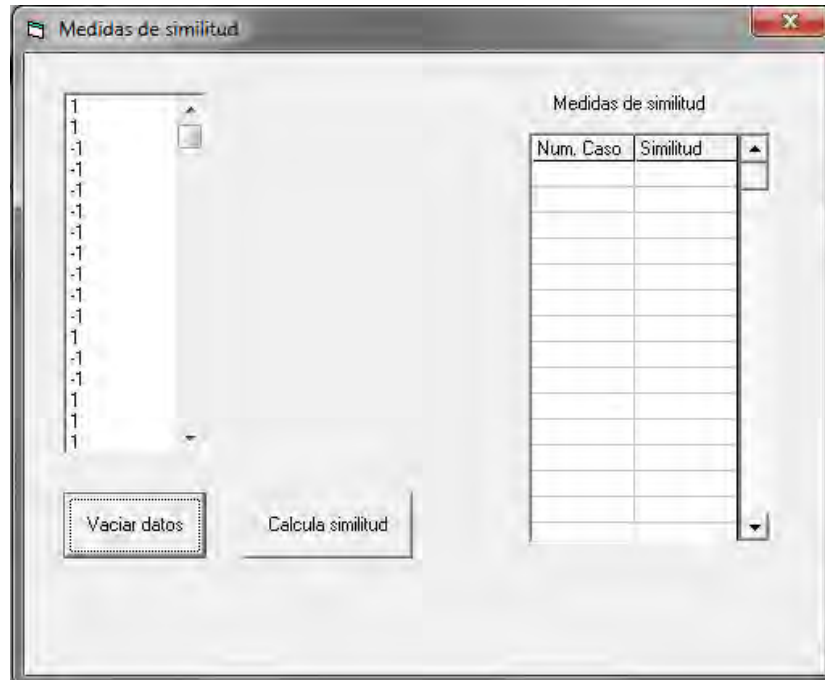


Figura B.12 Pantalla de similitud de datos, una vez que se dio clic en el botón "Vaciar datos".

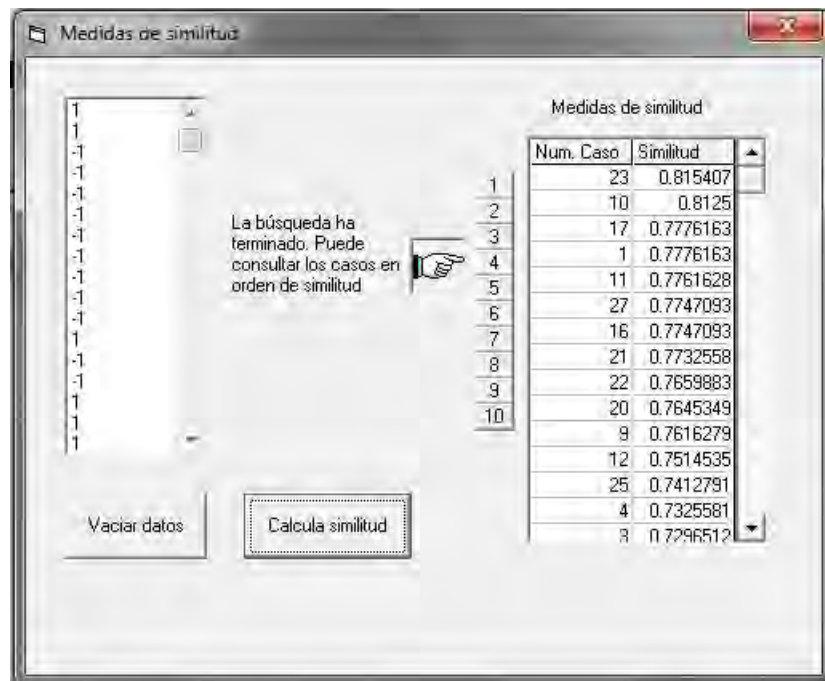


Figura B.13 Pantalla de medidas de similitud después de haber barrido toda la base de casos, se cálculo la similitud y se ordena los casos históricos.

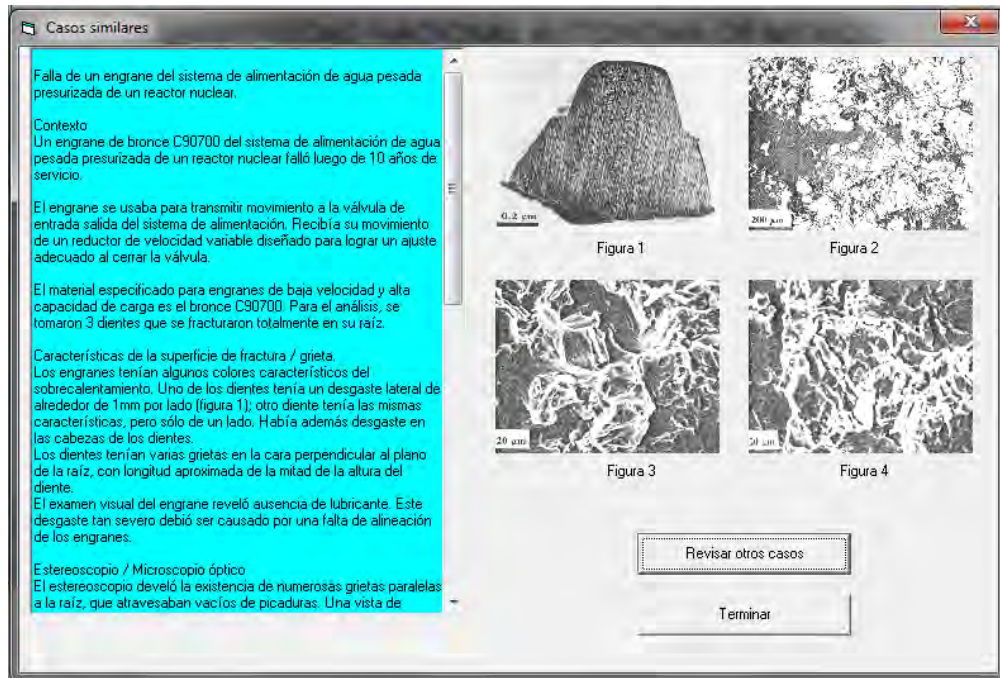


Figura B.14 Pantalla de casos similares

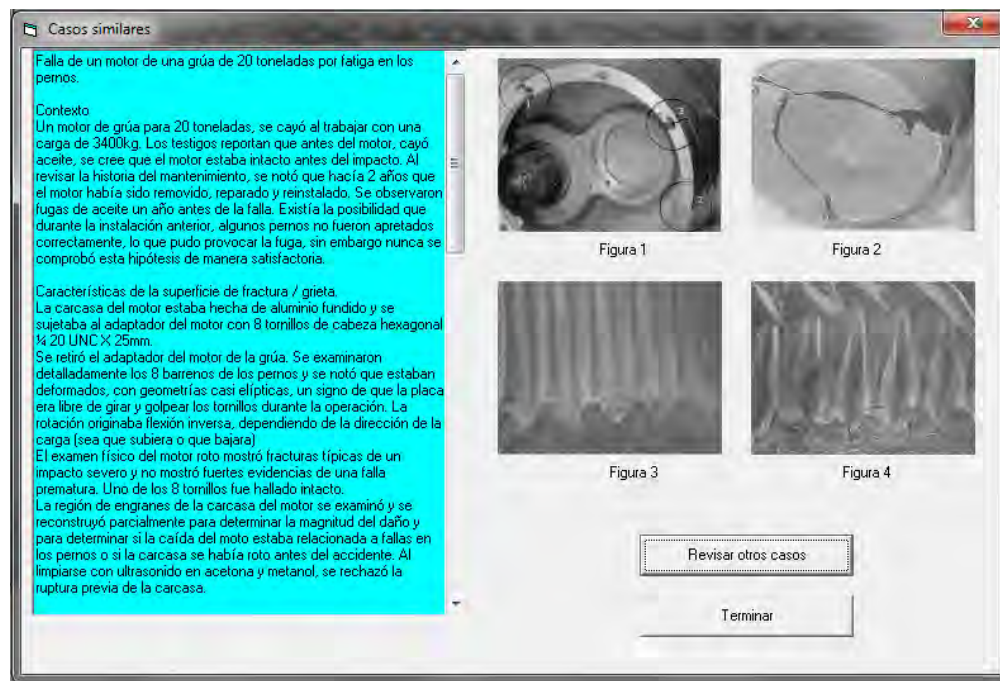
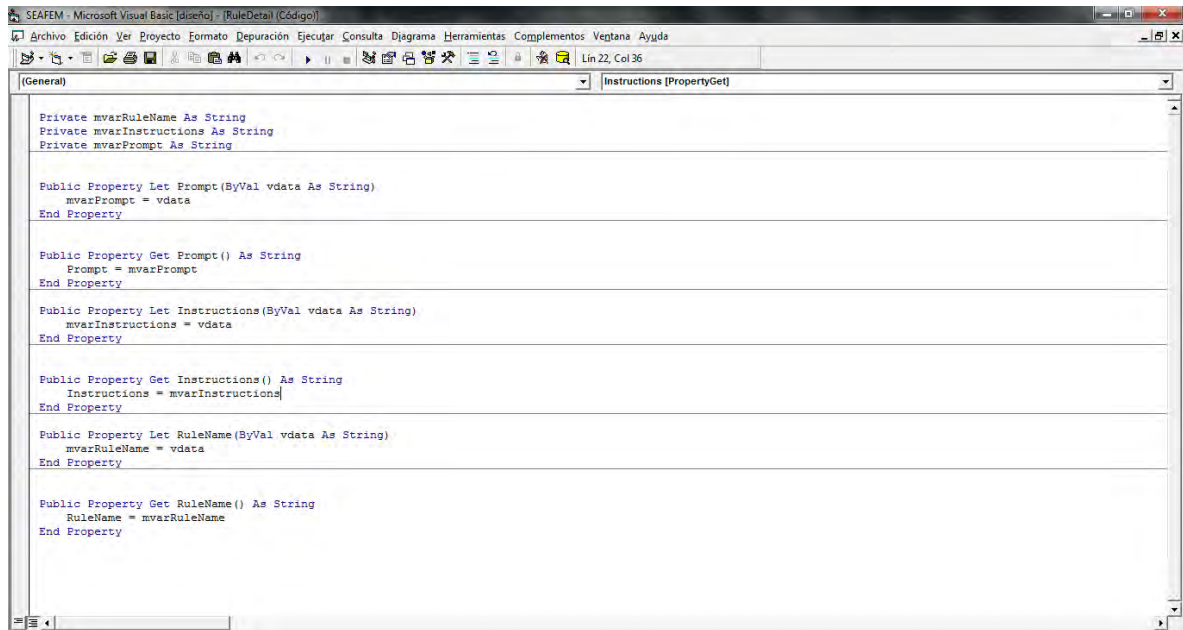


Figura B.15 Pantalla de casos similares

Sistema experto en ejes

Se presentan las capturas de pantalla del sistema experto en ejes.



```
SEAFEM - Microsoft Visual Basic [diseño] - [RuleDetail] (Código)
Archivo Edición Ver Proyecto Formato Depuración Ejecutar Consulta Diagrama Herramientas Complementos Ventana Ayuda
Lin 22, Col 36
[General] Instructions [PropertyGet]
Private mvarRuleName As String
Private mvarInstructions As String
Private mvarPrompt As String

Public Property Let Prompt(ByVal vdata As String)
    mvarPrompt = vdata
End Property

Public Property Get Prompt() As String
    Prompt = mvarPrompt
End Property

Public Property Let Instructions(ByVal vdata As String)
    mvarInstructions = vdata
End Property

Public Property Get Instructions() As String
    Instructions = mvarInstructions
End Property

Public Property Let RuleName(ByVal vdata As String)
    mvarRuleName = vdata
End Property

Public Property Get RuleName() As String
    RuleName = mvarRuleName
End Property
```

Figura B.16 Vista del código fuente dentro del entorno de desarrollo de Visual Basic 6.0.

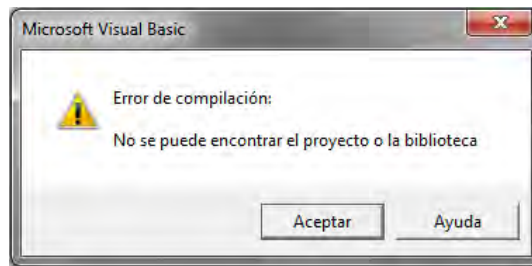


Figura B.17 Mensaje de error después de no poder ejecutar la aplicación por la falta de la biblioteca de *Visual Rule Studio*.

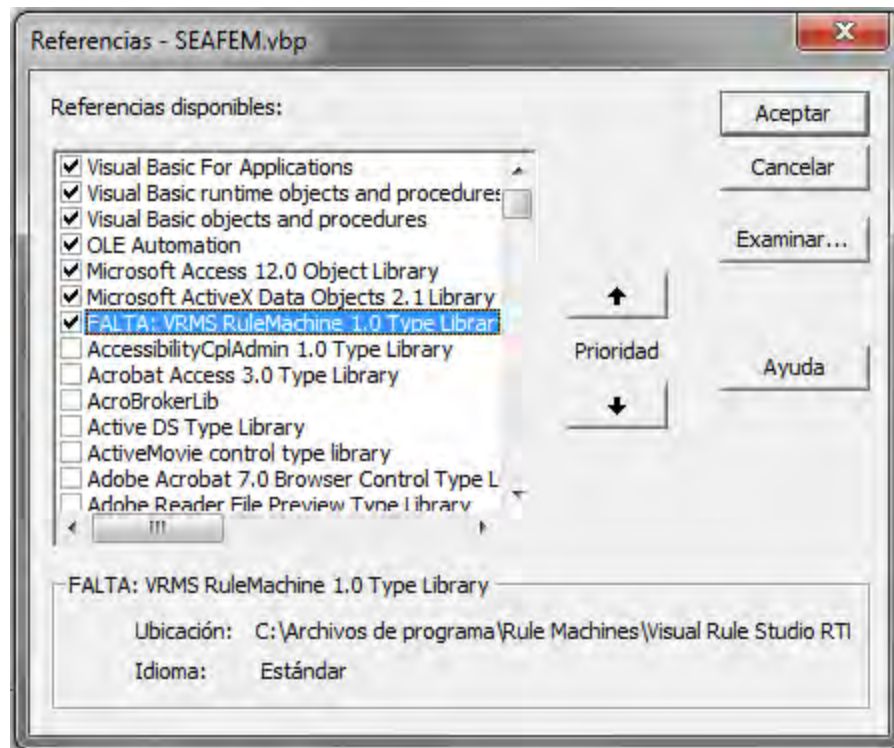


Figura B.18 Ventana para agregar la biblioteca de *Visual Rule Studio*.

Sistema experto en engranes

A continuación se muestran las capturas de pantallas del sistema experto en engranes.



Figura B.19 Pantalla para iniciar sesión en el sistema

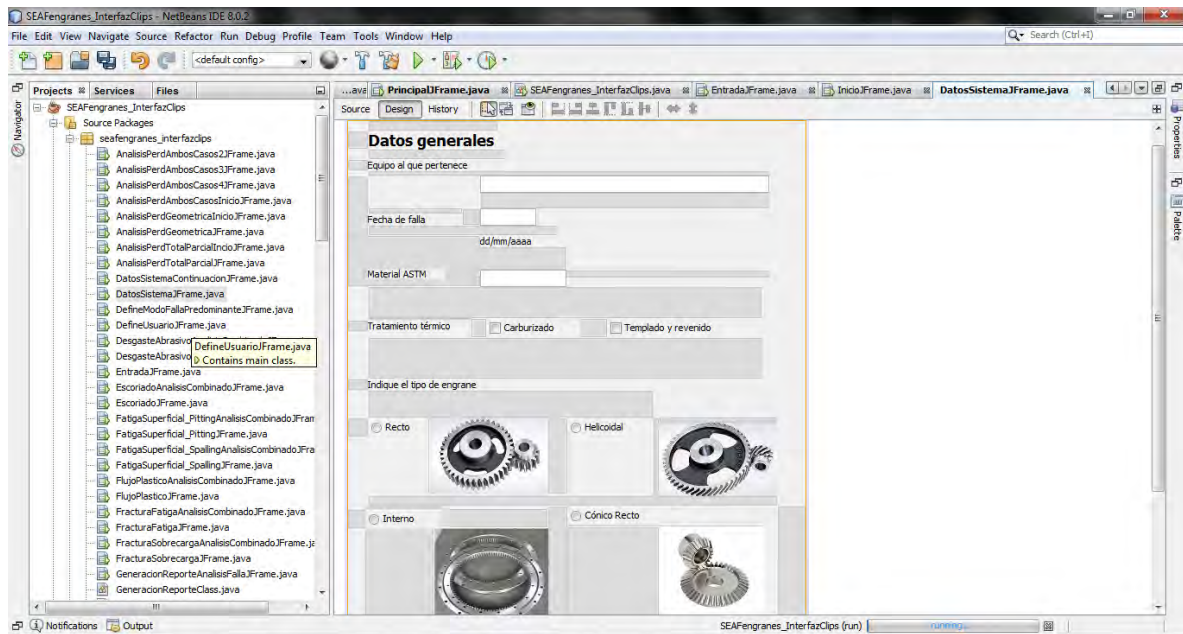


Figura B.20 Vista del proyecto dentro del entorno de desarrollo NetBeans.



Datos generales

Equipo al que pertenece

Fecha de falla
dd/mm/aaaa

Material ASTM

Tratamiento térmico Carburizado Templado y revenido

Indique el tipo de engrane

Recto  Helicoidal 

Interno  Cónico Recto 

Figura B.21 Datos generales, parte uno.

Datos generales_continuacion

Datos de mantenimiento/servicio

horas operativas [hrs]

Tipo de servicio

Condición del lubricante Buena Mala

Tipo de lubricante

- Aceite mineral puro
- Aceite mineral AP
- Aceite compuesto
- Aceite sintético
- Grasas
- Reemplazo periódicos de lubricante



Datos de desensamble

Condición de lubricante a la hora del desensamble

Buena Mala

Partículas ajenas precipitadas o en suspensión en el lubricante

Condición de pernos y sellos

Buena Mala


Fugas de lubricante





Figura B.22 Datos generales, parte dos

Datos de Inspección física

Defina el modo predominante de falla

 Pérdida de geometría del diente

 Pérdida parcial o total del diente

 Combinación de ambos casos

El modo de falla seleccionado se puede definir como el deterioro superficial que sufren los dientes del engrane en servicio, este tipo de fallas se manifiestan de diversas formas y razones. Las fallas más comunes de pérdida de geometría del diente se mencionan a continuación:

- Desgaste
Abrasivo
Corrosivo
- Fatiga Superficial
Picado
Astillado.


Los principales signos de este tipo de falla son:

- Deterioro general del perfil del diente
- Remoción de material superficial
- Aparición de crateres o cavidades superficiales


Inicie el análisis para determinar el tipo de falla más aproximado...

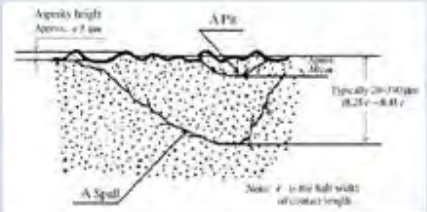
Figura B.23 Datos de inspección física

Análisis de pérdida de geometría Inicio

Presencia de picaduras 

Forma Aproximada Regular
 Irregular (Astillas)

Tamaño aproximado 2.5 micrometros
 de 25 a 50 micrometros 

Profundidad aproximada 10 micrometros
 desde 20 hasta 100 micrometros 


Ubicación de la falla Adendo
 Dedendo
 Línea de paso 

Figura B.24 Análisis de pérdida geométrica, primera parte.

Análisis de falla para pérdida de geometría del diente

Datos de inspección física

Presencia de marcas o estrias en dirección al deslizamiento 

Presencia de marcas de lagrimeo 

Reducción del espesor del diente

Destrucción del perfil del diente 

Superficies en contacto planas y desgastadas

Remoción de material 

Línea de paso predominante 

Transferencia de material entre superficies

Diagnostico

Figura B.25 Análisis de falla para pérdida de geometría del diente (segunda parte).

Análisis para pérdida total o parcial del diente Inicio

Datos de Inspección física

Presencia de grietas en la raíz del diente 

Presencia de punto focal de donde surge la falla 

Presencia de marcas de playa 

Geometría de la falla

Concava

Convexa 

Figura B.26 Análisis para pérdida total o parcial del diente, primera parte.

Análisis de falla para pérdida total o parcial del diente

Datos de inspección física

Apariencia Aspera en la superficie de la falla
Extensión de la apariencia aspera

En toda la superficie 

En una área pequeña 

Existencia de deformación 

Flujo de material sobre la punta del diente 

Cizallamiento aparente en los dientes 

Figura B.27 Análisis de falla para pérdida total o parcial de diente (segunda parte).

Sistema experto en tornillos

Se presentan las capturas realizadas al sistema experto en tornillos.

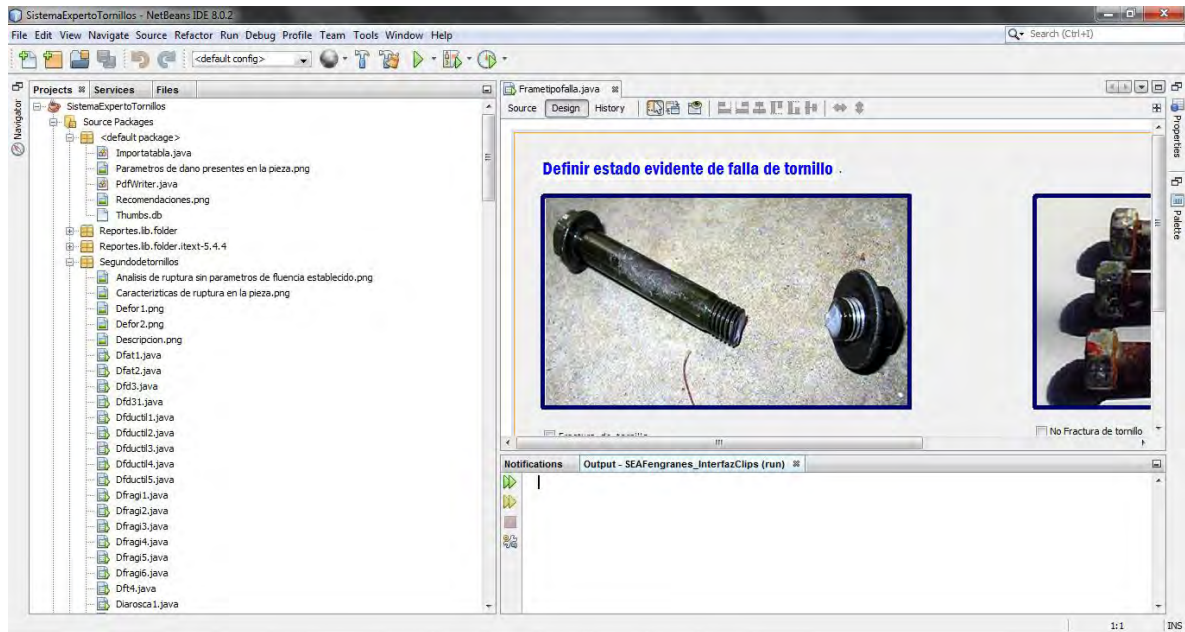


Figura B.28 Vista del proyecto dentro del entorno de Netbeans

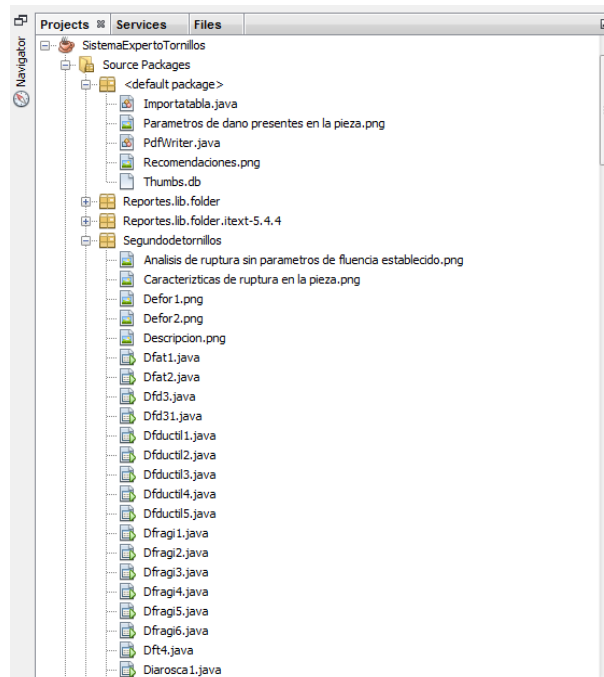


Figura B.29 Vista de los recursos utilizados para la ejecución del proyecto, claramente se observa que no existe una buena estructura.



Figura B.30 Pantalla de inicio del sistema, se inicia el análisis de falla sobre la existencia de fractura en el tornillo.



Figura B.31 Características de ruptura en la pieza, no se observa flujo de material en la pieza.



Figura B.32 Se definen los parámetros encontrados en la ruptura del tornillo

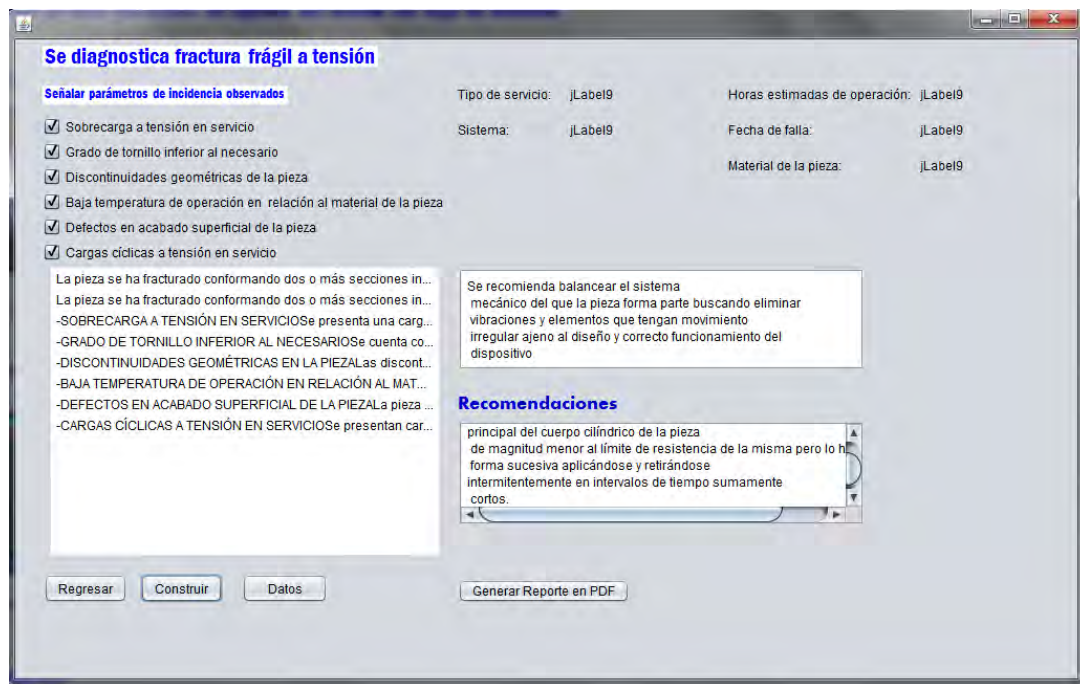


Figura B.33 El sistema da el diagnóstico y se realiza el vaciado de los datos dando clic sobre el botón "Construir".

Dando clic en el botón “Generar reporte en PDF” de la figura B.33. Se abre una venta, figura B.34, se especifica el nombre y la ruta del archivo.

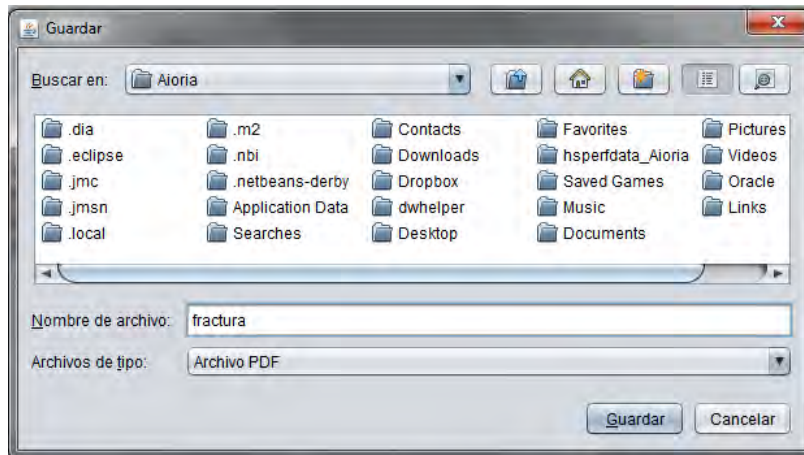


Figura B.34 Guardar reporte

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
UDIATEM
SISTEMA EXPERTO PARA ANALISIS DE FALLAS EN TORNILLOS

DATOS DE LA PIEZA

Tipo de Servicio:
jLabel9

Sistema:
jLabel9

Horas estimadas de operacion:
jLabel9

Fecha de falla:
jLabel9

Material de la pieza:
jLabel9

DIAGNOSTICO DE FALLA

[La pieza se ha fracturado conformando dos o más secciones independientes no se observa flujo de material remarcado, además se cuenta con una zona de fractura que presenta textura granular opaca con escalones minúsculos ajenos a la circunferencia del cuerpo cilíndrico del tornillo por lo cual se determina que el tipo de falla desarrollado es fractura frágil a tensión.

PARÁMETROS DE INCIDENCIA OBSERVADOS
, La pieza se ha fracturado conformando dos o más secciones independientes no se observa flujo de material remarcado, además se cuenta con una zona de fractura que presenta textura granular opaca con escalones minúsculos ajenos a la circunferencia del cuerpo cilíndrico del tornillo por lo cual se determina que el tipo de falla desarrollado es fractura frágil a tensión.

PARÁMETROS DE INCIDENCIA OBSERVADOS
,

-SOBRECARGA A TENSIÓN EN SERVICIO
Se presenta una carga en dirección paralela al eje

Figura B.35 Reporte generado, primera parte.

principal del cuerpo cilíndrico del tornillo y de magnitud superior a la resistencia del material que lo compone por lo cual después de un determinado periodo de tiempo la pieza se fracturara en función a las condiciones de operación.

Se recomienda añadir tornillos adicionales si es posible, con el objeto de distribuir la carga o en su caso disminuir la magnitud de dicha carga. También debe considerarse la selección de un tornillo compuesto por material de mayor resistencia al de la pieza actual.

-GRADO DE TORNILLO INFERIOR AL NECESARIO
Se cuenta con una pieza que presenta una resistencia menor a la solicitada por los parámetros de operación del sistema en cuestión.

Se recomienda seleccionar un grado de tornillo acorde a las dimensiones de la carga máxima que se soportaran en operación.

-DISCONTINUIDADES GEOMÉTRICAS EN LA PIEZA
Las discontinuidades geométricas en la pieza incluyen adelgazamientos del cuerpo cilíndrico del tornillo o de la sección de cuerda, así como muescas o irregularidades inherentes al diseño de fábrica.

Se recomienda seleccionar geometrías uniformes para la pieza que no incluyan alguna de las mencionadas irregularidades.

-BAJA TEMPERATURA DE OPERACIÓN EN RELACIÓN AL MATERIAL DE LA PIEZA
Algunos dispositivos mecánicos operan a temperaturas sumamente bajas en relación al material que compone al tornillo en cuestión, por lo que el mismo tiende a aumentar su dureza y por ende su fragilidad con lo que se pierde resistencia a cierto tipo de esfuerzos como es el caso de esfuerzos cortantes.

Se recomienda aislar térmicamente a la

Figura B.36 Reporte generado, segunda parte.

pieza en cuestión si esto es posible o en su defecto seleccionar un tornillo cuya composición sea de un material relativamente flexible.

-DEFECTOS EN ACABADO SUPERFICIAL DE LA PIEZA
La pieza presenta irregularidades en su superficie lo cual implica concentración de esfuerzos y por ende un punto de inicio de ruptura de la pieza. Se recomienda seleccionar piezas sin defectos en su superficie o corregir factores que dañen dicha superficie en condiciones de operación.

-CARGAS CÍCLICAS A TENSIÓN EN SERVICIO
Se presentan cargas con dirección paralela al eje principal del cuerpo cilíndrico de la pieza de magnitud menor al límite de resistencia de la misma pero lo hacen de forma sucesiva aplicándose y retirándose intermitentemente en intervalos de tiempo sumamente cortos.

Se recomienda balancear el sistema mecánico del que la pieza forma parte buscando eliminar vibraciones y elementos que tengan movimiento irregular ajeno al diseño y correcto funcionamiento del dispositivo]

Figura B.37 Reporte generado, tercera parte.

Anexo C: Operación del SIAFEM

En este apartado se anexan las capturas de pantalla del SIAFEM, el lector podrá ver el resultado del presente trabajo. Se omite la captura de pantalla de inicio del sistema, ya que fue presentada en el capítulo cuatro, ver figura 4.3.

Sistema experto en engranes

A continuación se presenta un ejemplo representativo desarrollado por Díaz (2014: 68), se resuelve un caso con las siguientes características:

- Presencia de picaduras (figura C.4).
- Picaduras en forma irregular (figura C.4).
- Tamaño aproximado de 20 a 50 μm (figura C.4).
- Ubicación de la falla: línea de paso (figura C.5).
- Presencia de grietas en la raíz del diente (figura C.7).
- Presencia de marcas de playa (figura C.7).
- Geometría de falla cóncava (figura C.7).
- Apariencia superficial áspera (figura C.7).
- Apariencia superficial áspera en una pequeña área (figura C.8).

El usuario podrá proporcionar a la aplicación web datos generales de la pieza obtenidos a través de una exanimación de campo realizada previamente. En las figuras C.1, C.2 y C.3 se puede observar el formulario que permite la captura de la información.

Sistema Integrado para Análisis de Fallas de Elementos Mecánicos







Datos Generales

Equipo al que pertenece: *

Fecha de falla: * m

Material ASTM: *

Tratamiento térmico * Carburizado Templado y revenido

Indique el tipo de engrane

Recto  Helicoidal 

Interno  Cónico Recto 

Datos de Mantenimiento/Servicio

Horas operativas: * Hrs.

Tipo de servicio: *

Condición del lubricante: * Buena Mala





Tipo de lubricante: Aceite mineral puro Aceite mineral AP







Figura C.1 Datos generales, primera parte.

Tipo de servicio: *

Condición del lubricante: * Buena Mala

Tipo de lubricante: Aceite mineral puro Aceite mineral AP Aceite compuesto Aceite Sintético Grasas

Reemplazo periódico de lubricante

Datos de desensamble

Condición del lubricante a la hora del desensamble: * Buena Mala


Partículas ajenas precipitadas o en suspensión en el lubricante


Condición de pernos y sellos: * Buena Mala

Fugas de lubricante

Datos de Inspección física

Defina el modo predominante de falla:

 Pérdida de geometría del diente

 Pérdida parcial o total del diente










Figura C.2 Datos generales, segunda parte.

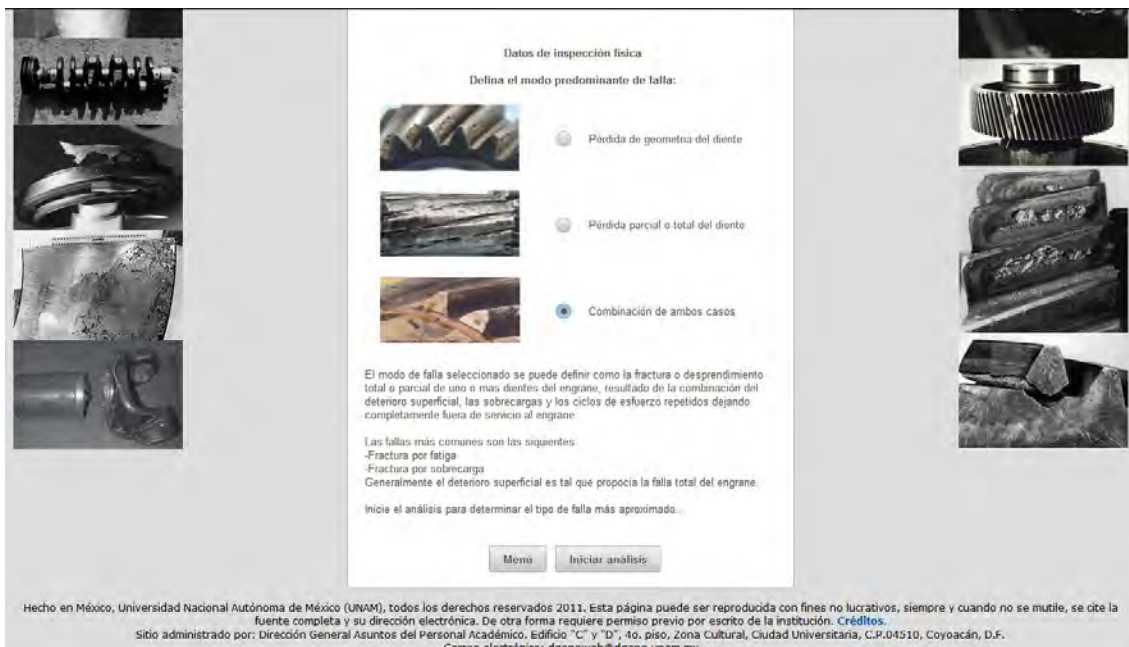


Figura C.3 Datos generales, tercera parte.



Figura C.4 Pérdida de geometría del diente, primera parte.

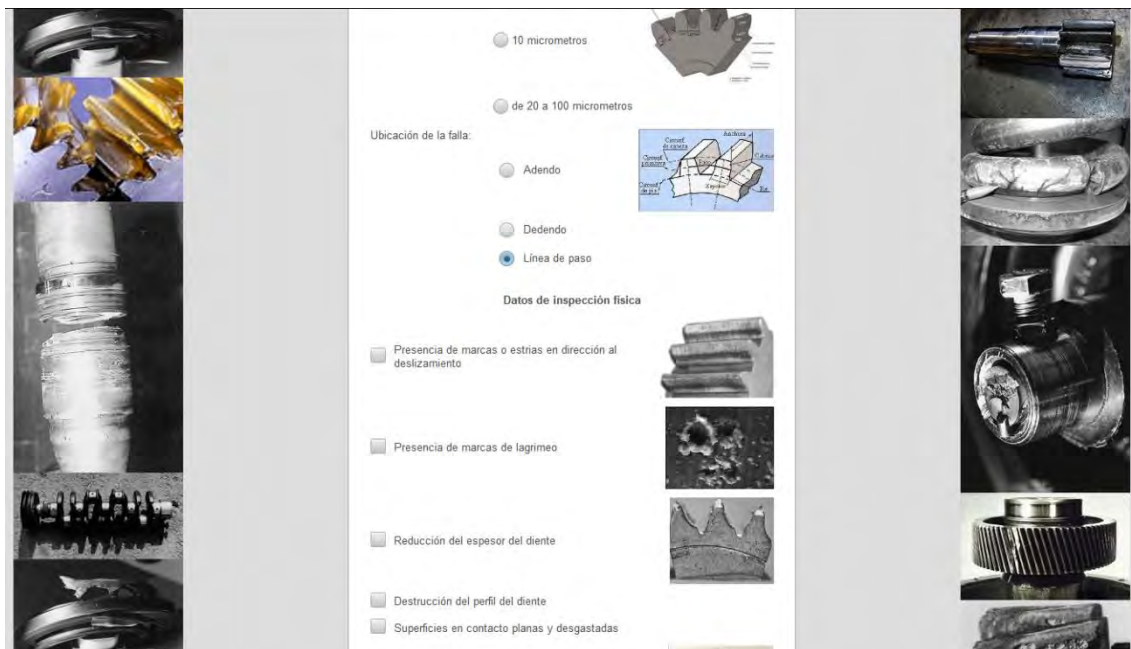
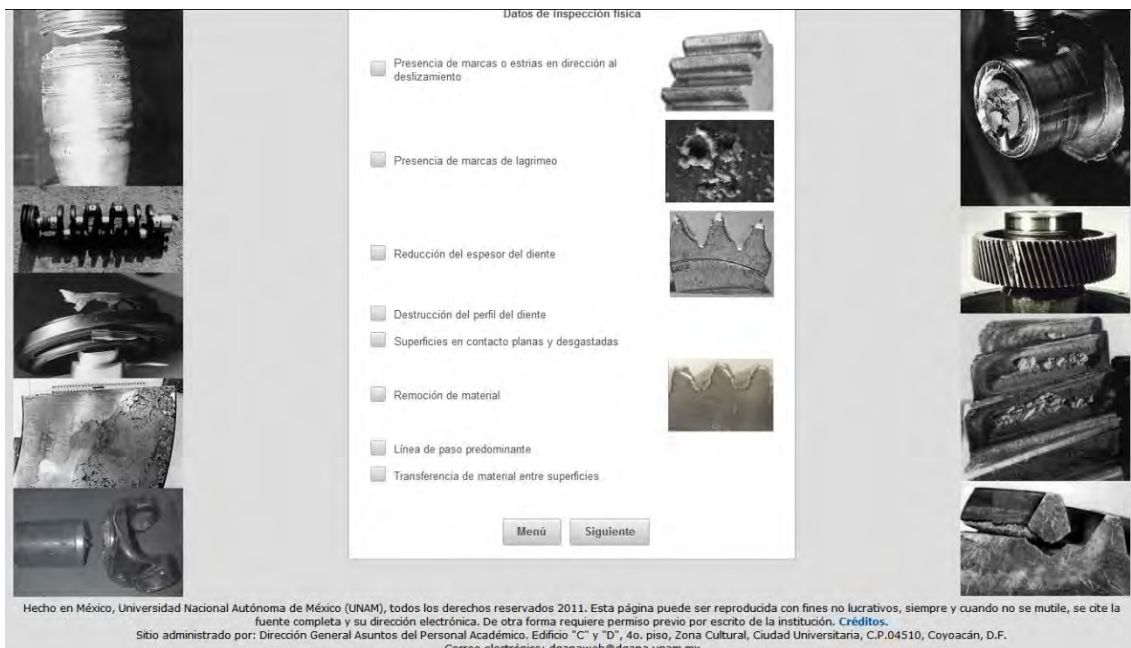


Figura C.5 Pérdida de geometría del diente, segunda parte.



Hecho en México, Universidad Nacional Autónoma de México (UNAM), todos los derechos reservados 2011. Esta página puede ser reproducida con fines no lucrativos, siempre y cuando no se mutile, se cite la fuente completa y su dirección electrónica. De otra forma requiere permiso previo por escrito de la institución. [Créditos](#).
 Sitio administrado por: Dirección General Asuntos del Personal Académico, Edificio "C" y "D", 4o. piso, Zona Cultural, Ciudad Universitaria, C.P.04510, Coyoacán, D.F.
 Correo electrónico: dgapaweb@dgapa.unam.mx

Figura C.6 Pérdida de geometría del diente, tercera parte.



Figura C.7 Pérdida total o parcial del diente, primera parte.

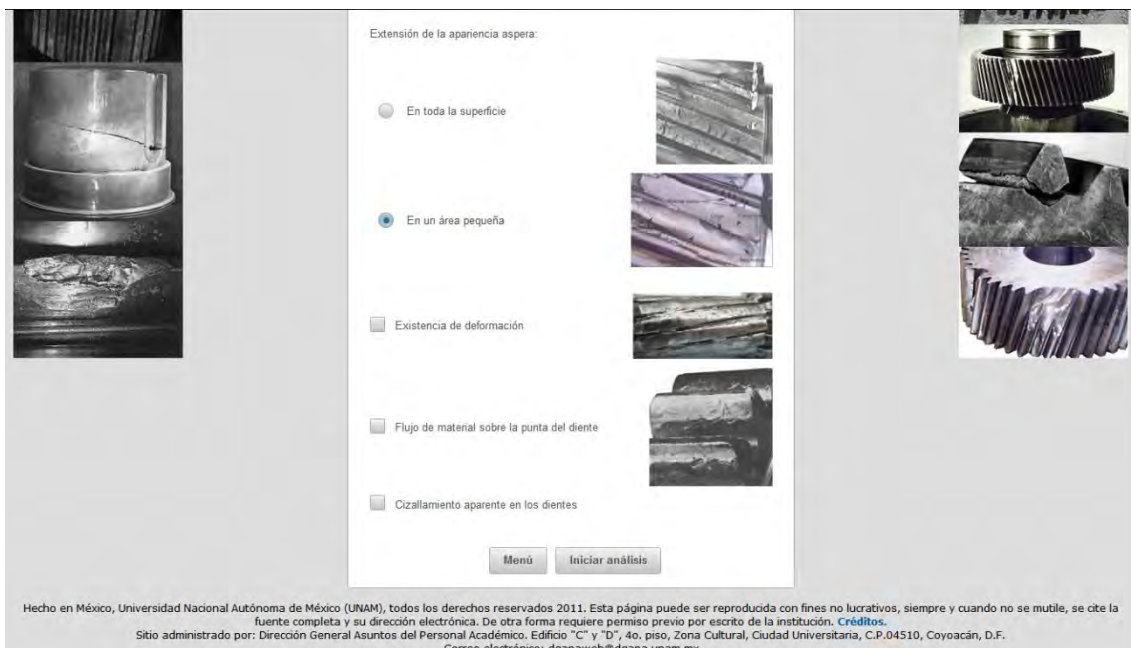


Figura C.8 Pérdida total o parcial del diente, segunda parte.

Sistema Integrado para Análisis de Fallas de Elementos Mecánicos



Fatiga superficial

Picado-Pitting

Descripción de la falla

La fatiga superficial es resultado de esfuerzos repetidos en la superficie del diente del engrane inherentes al funcionamiento del sistema, dichos esfuerzos forman grietas superficiales o subsuperficiales, este tipo de falla se presenta incluso con lubricación adecuada así como con una película ininterrumpida de aceite después de muchos millones de ciclos de esfuerzo, por consiguiente las fallas por fatiga resultan evidentes después de un tiempo prolongado de servicio.

Los engranes que funcionan bajo carga desarrollan esfuerzos superficiales constantes, solo si la carga tiene la suficiente intensidad y se repite con bastante frecuencia sobrevendrá la fatiga del material en donde fragmentos de metal son desprendidos de la superficie originando pequeñas picaduras.

Existen dos grados de fatiga superficial o picado, los cuales dependen directamente del tamaño de las picaduras, los grados en los que se presenta el picado se enuncian a continuación:

-Picado inicial

Este grado de picado se caracteriza por pequeñas picaduras de unos cuantos micrometros de diametro, el picado inicial se presenta en areas localizadas en donde existan irregularidades en el perfil que concentren esfuerzos.

Por tanto el picado inicial es causado por un mal acabado superficial o pequeños errores en el perfil del diente. Este grado de picado se le considera normal y rara vez se establece alguna acción para corregirlo.

- Picado destructivo

En este grado de picado, las picaduras son considerablemente de mayor diametro que en el picado inicial y se extienden a lo largo de todo el





Figura C.9 Resultado del análisis (Fatiga superficial), primera parte.



- Picado destructivo

En este grado de picado, las picaduras son considerablemente de mayor diametro que en el picado inicial y se extienden a lo largo de todo el diente con diversos tamaños y formas, generalmente el engrane conductor es el primero en picarse debido a que maneja una mayor carga además de ser el de menor diametro esta sometido a esfuerzos con mayor frecuencia en el caso de un sistema que incremente el torque.

Este grado de picado es producto de someter a una mayor carga al engrane una vez que se presenta el picado inicial.

Causas:

La fatiga superficial se produce primeramente por imperfecciones superficiales, además de errores mínimos en el perfil del diente dando lugar a esfuerzos de contacto por encima del limite de fatiga del material, por otra parte una mala alineación del sistema o un incremento en la carga o velocidad de servicio puede traer consigo el incremento en la intensidad de los esfuerzos sobre la superficie del diente, los cuales como se menciono anteriormente son los principales causantes de este tipo de falla.

Acciones preventivas y/o correctivas

Una vez que se observan las huellas características de la fatiga superficial y se identifica el grado de picado que se tiene se pueden considerar las siguientes acciones preventivas o correctivas según sea el caso.

- Picado inicial

Este tipo de falla puede ser evitada asegurando un buen acabado superficial en las areas en donde se lleva acabo el contacto, por otra parte mejorar la precision del perfil de evolvente o modificando el diseño a fin de reducir la carga dinámica a la que será sometido el diente.

-Picado Destructivo

Este tipo de picado se puede evitar manteniendo la carga por debajo del limite de fatiga del material por otra parte es posible incrementar la dureza del material a fin de incrementar el limite de fatiga del material en donde la fatiga superficial no




Figura C.10 Resultado del análisis (Fatiga superficial), segunda parte.

sistema que incrementa el torque.

Este grado de picado es producido de someter a una mayor carga al engrane una vez que se presenta el picado inicial.

Causas:

La fatiga superficial se produce primeramente por imperfecciones superficiales, además de errores mínimos en el perfil del diente dando lugar a esfuerzos de contacto por encima del límite de fatiga del material, por otra parte una mala alineación del eje o un incremento en la carga o velocidad de servicio puede traer consigo el incremento en la intensidad de los esfuerzos sobre la superficie del diente, los cuales como se mencionó anteriormente son los principales causantes de este tipo de falla.

Acciones preventivas y/o correctivas

Una vez que se observan las huellas características de la fatiga superficial y se identifica el grado de picado que se tiene se pueden considerar las siguientes acciones preventivas o correctivas según sea el caso.

- Picado inicial

Este tipo de falla puede ser evitada asegurando un buen acabado superficial en las áreas en donde se lleva a cabo el contacto, por otra parte mejorar la precisión del perfil de evolvente o modificando el diseño a fin de reducir la carga dinámica a la que será sometido el diente.

- Picado Destructivo

Este tipo de picado se puede evitar manteniendo la carga por debajo del límite de fatiga del material por otra parte es posible incrementar la dureza del material a fin de incrementar el límite de fatiga del material en donde la fatiga superficial no tenga lugar.


Hecho en México, Universidad Nacional Autónoma de México (UNAM), todos los derechos reservados 2011. Esta página puede ser reproducida con fines no lucrativos, siempre y cuando no se mutile, se cite la fuente completa y su dirección electrónica. De otra forma requiere permiso previo por escrito de la institución. [Créditos](#).
 Sitio administrado por: Dirección General Asuntos del Personal Académico, Edificio "C" y "D", 4o. piso, Zona Cultural, Ciudad Universitaria, C.P.04510, Coyoacán, D.F.
 Correo electrónico: dgapaweb@dgapa.unam.mx

Figura C.11 Resultado del análisis (Fatiga superficial), tercera parte.

Sistema experto en tornillos

En seguida se muestra la operación del sistema realizado por Martínez (2015), el funcionamiento se basa en seleccionar parámetros obtenidos mediante una inspección visual realizada previamente. El usuario podrá ingresar datos generales de la pieza a través del formulario de la figura C.12.

Sistema Integrado para Análisis de Fallas de Elementos Mecánicos



Parámetros generales de sistema

Los siguientes datos no afectarán el diagnóstico de falla emitido por el programa.

Tipo de servicio: *

Sistema: *

Material de la pieza: *

Fecha de falla:


Horas estimadas de operación: * Hra.

Fabricante: *

Tratamiento térmico: *

Norma reguladora de la pieza: *

Observaciones generales



Hecho en México, Universidad Nacional Autónoma de México (UNAM), todos los derechos reservados 2011. Esta página puede ser reproducida con fines no lucrativos, siempre y cuando no se modifique, se cite la fuente completa y su dirección electrónica. De otra forma requiere permiso previo por escrito de la institución. [Créditos](#).

Figura C.12 Parámetros generales del sistema.



Figura C.13 Definir estado evidente de la falla del tornillo.



Figura C.14 Características de ruptura en la pieza.

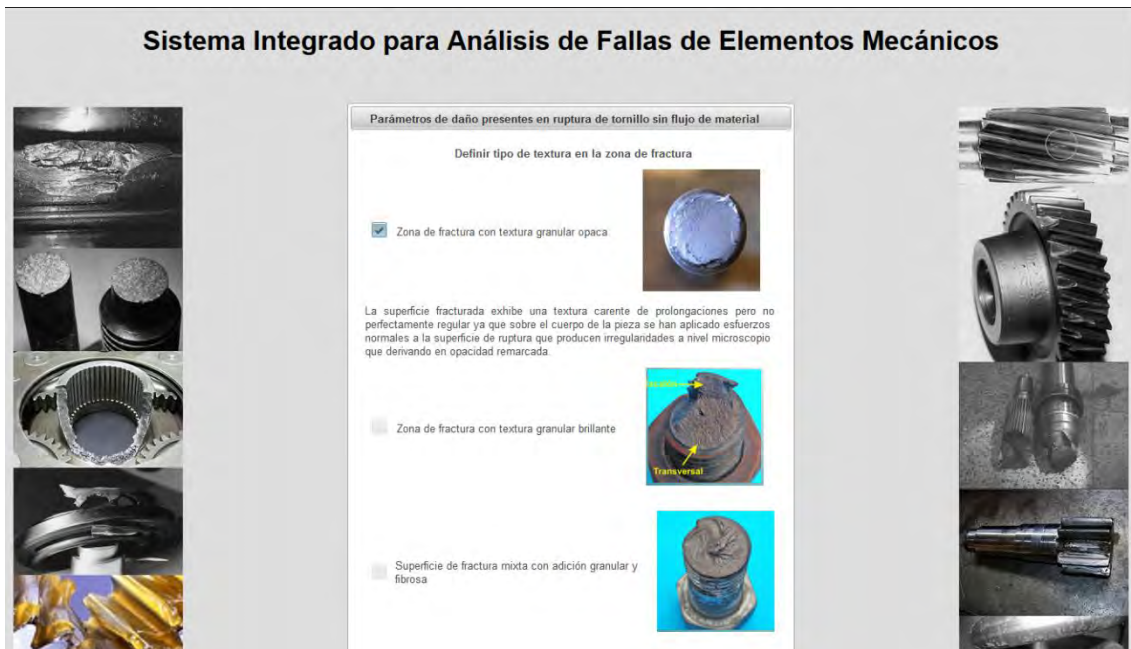


Figura C.15 Parámetros de daño presentes en ruptura de tornillo sin flujo de material, primera parte.

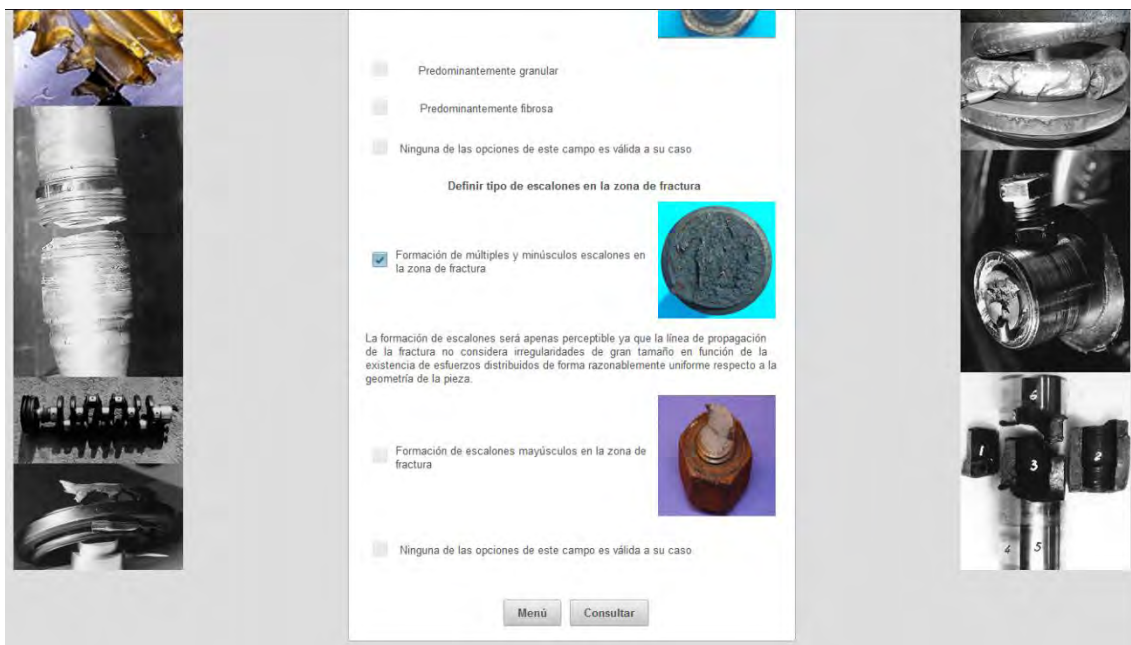


Figura C.16 Parámetros de daño presentes en ruptura de tornillo sin flujo de material, segunda parte.

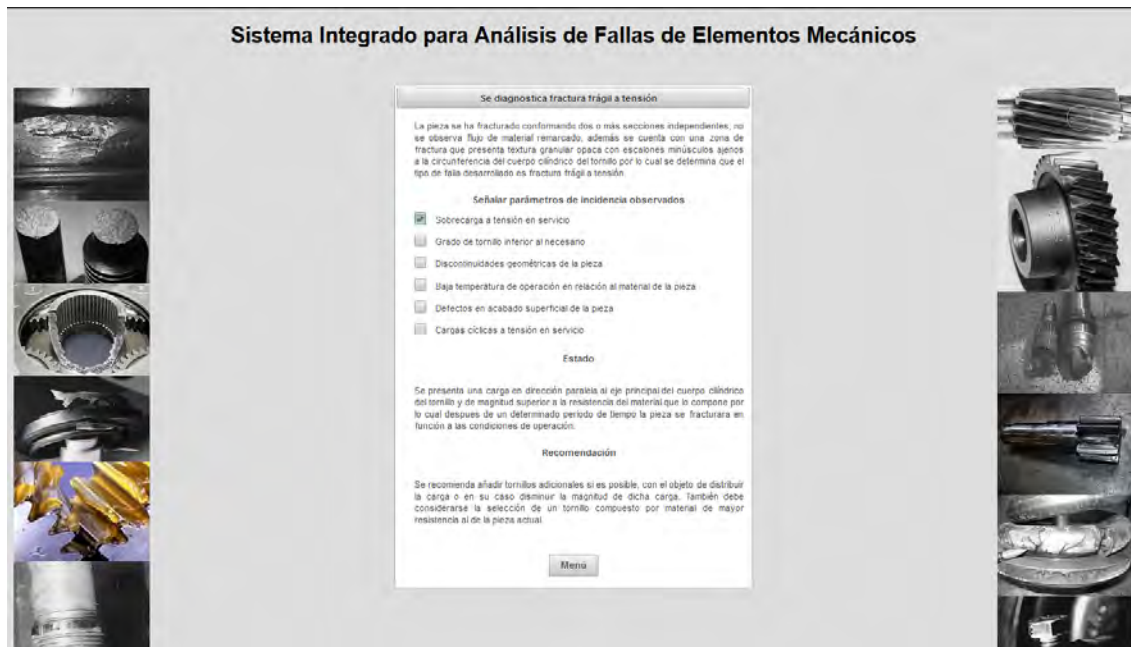


Figura C.17 Resultado del análisis de falla presente en tornillos.

Apuntes

En la figura C.18 se presenta el menú de los documentos relacionados con el análisis de falla de elementos mecánicos.

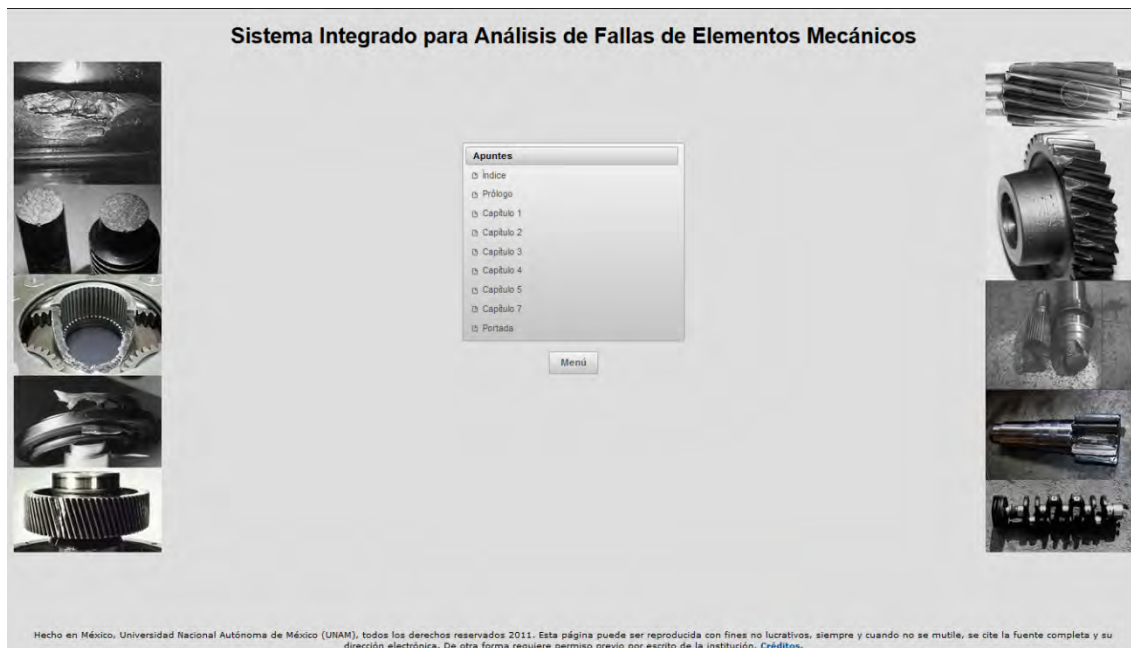


Figura C.18 Menú para consultar los documentos relacionados con el análisis de falla.

Para proporcionar un visor de archivos PDF, se hizo uso del *framework PrimeFaces* para mostrar contenido multimedia a través del componente *media*, el resultado se observa en la figura C.19.

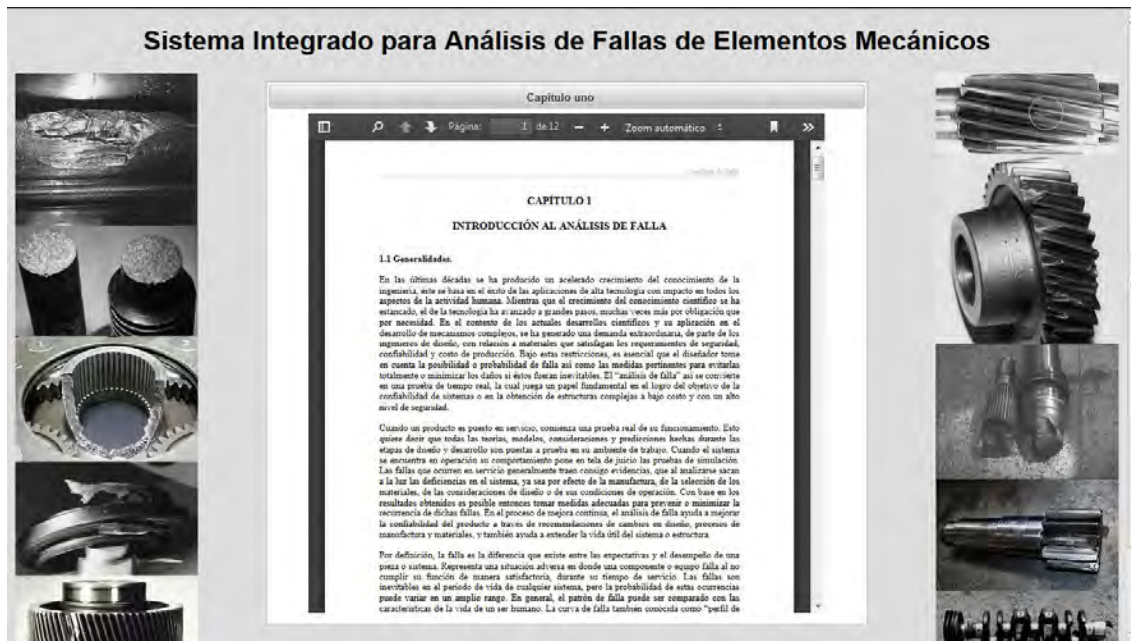


Figura C.19 Visor de documentos PDF.