UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

Sistema basado en conocimiento para verificar la trazabilidad de requisitos de un sistema de software.

T E S I S

QUE PARA OPTAR POR EL GRADO DE:

MAESTRA EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

P R E S E N T A:

GABRIELA GUADALUPE MORENO GUTIÉRREZ

Directora de Tesis:

M. en C. María Guadalupe Elena Ibargüengoitia González

Facultad de Ciencias

Codirector de Tesis:

Dr. Sergio Marcellin Jacques

Coordinación de Estudios de Posgrado

Ciudad Universitaria, CD. MX. marzo de 2018

Universidad Nacional Autónoma de México
UNAM
Biblioteca Central
Dirección General de Bibliotecas de la UNAM

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 esta 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.

Contenido

Capítulo 1. Marco teórico 1
1.1 Requisitos de software 1
1.2 Trazabilidad de requisitos 1
1.2.1 Objetivos de la trazabilidad de requisitos 2
1.2.2 Modos de trazabilidad de requisitos 2
1.2.3 Modelo de trazabilidad de requisitos 3
1.3 Estado del arte en trazabilidad de requisitos 4
1.4 Norma ISO/IEC 29110. Ingeniería de Software – Perfiles de Ciclo de Vida para Empresas Muy Pequeñas 6
1.5 Sistemas basados en conocimiento 9
1.5.1 Ventajas de los sistemas basados en conocimiento 10
1.5.2 Componentes de un sistema basado en conocimiento 11
1.5.3 Lenguajes de sistemas basados en conocimiento 12
1.5.4 CLIPS 12
1.5.5 Herramientas para el desarrollo de sistemas basados en conocimiento 15
Capítulo 2. Proceso de implementación de software en SEVETREQ 17
2.1 Descripción de SEVETREQ 17
2.2 Análisis de requisitos de SEVETREQ 18
2.2.1 Casos de uso 19
2.2.2 Diseño de la base de datos 22
2.3 Arquitectura y diseño de SEVETREQ 24
2.4 Casos de prueba 26
Capítulo 3. Estructura del sistema basado en conocimiento de SEVETREQ 28
3.1 Base de conocimiento de SEVETREQ 28
CAPÍTULO 4. APLICACIÓN DE SEVETREQ EN UN SISTEMA REAL 32
4.1 Inicio de aplicación de software de “adminsystem” 33
4.2 Actividad de Análisis de requisitos de software 35
4.3 Actividad de arquitectura y diseño detallado de software 37
4.4 Registro de trazabilidad 40
4.5 En resumen 41
CONCLUSIONES Y TRABAJO FUTURO 43
Bibliografía 45

Anexos.....47
A. Documento de plan del proyecto.....47
B.Documento de especificación de requisitos de SEVETREQ.....50
C. Documento de diseño de software para SEVETREQ.....72
D. Documento de casos de prueba.....79

Resumen

Uno de los objetivos en el desarrollo de software es automatizar ciertas tareas del ser humano, para conseguir más precisión y velocidad en sus actividades; en este contexto, el software utilizado debería tener la menor cantidad de errores y para que esto sea posible es necesario realizar pruebas suficientes. Dichas pruebas deben basarse en los requisitos cuya documentación incluye una descripción y el detalle de las necesidades del usuario que cumple.

El desarrollo de software se considera parte de un proceso de implementación de software en el que también se considera su documentación, esta incluye, entre otros, un documento de requisitos y uno de diseño; por un lado, en el documento de requisitos, se depositan los requisitos obtenidos del cliente y por otro, en el documento de diseño, los componentes abstraídos a partir de dichos requisitos. Para llevar un seguimiento de esta abstracción, realizar la trazabilidad de requisitos resulta útil; a grandes rasgos, la trazabilidad de requisitos permite conocer dónde "nació" el requisito y en qué componente se convirtió, de esta manera, el equipo de desarrollo será capaz de comparar lo solicitado contra lo implementado y detectar omisiones tanto de componentes como de requisitos.

Esta tesis de maestría describe un sistema web basado en conocimiento apoyado en la norma ISO/IEC 29110 Perfil básico que muestra la trazabilidad de requisitos de un sistema de software por medio de una matriz de requisitos generada a partir de hechos y reglas de inferencia de la base de conocimiento del sistema.

Introducción

Uno de los objetivos en el desarrollo de software es automatizar ciertas tareas del ser humano, con esto, también se consigue más precisión y velocidad en sus actividades.

Para el desarrollo de cualquier proyecto de software es fundamental que se haga una correcta definición de requisitos (IBM, 2008); requisitos confiables y precisos pueden ayudar a mejorar la calidad del software. Si se definen los requisitos comprometiendo a los involucrados pronto y a menudo, la precisión y completitud de los requisitos puede mejorar la calidad del producto significativamente.

Por otro lado, es necesario documentar los requisitos de manera que permita tanto al equipo de desarrollo como al cliente asegurar que se cumplan en el producto de software. En este contexto, es esencial mantener la trazabilidad de requisitos con los componentes producidos durante el proceso del diseño del sistema. (Ramesh, Stubbs, Powers, & Edwards, 1997)

Realizar trazabilidad de requisitos, permite a las organizaciones asegurar beneficios, los cuales son: entrega de software completo y correcto, así como asegurar que cumpla su cometido.

La trazabilidad de requisitos es un tema que se está investigando con el objetivo de que en el futuro el equipo de trabajo la vea como parte de su proceso de desarrollo y pueda llevarla a cabo fácilmente.

Se espera que la trazabilidad facilite tareas en todas las fases del ciclo de vida de la Ingeniería de software y sistemas, proveyendo ganancias tanto en calidad como productividad. En particular, ayudará con la definición de requisitos obteniendo el diseño, código y casos de prueba asociados. (Gotel, Cleland-Huang, & Hayes et al., The grand challenge of traceability (v1.0), 2012)

En esta tesis de maestría se propone un sistema web basado en conocimiento apoyado en el proceso de Implementación de Software (IS) de la norma ISO/IEC 29110 Perfil básico que implemente la trazabilidad de requisitos por medio de una matriz de trazabilidad donde se muestra la relación que existe entre requisitos y componentes del diseño ya sean clases y/o interfaces gráficas. Dicha matriz se genera a partir de hechos y reglas de inferencia de la base de conocimiento del sistema.

Además, el sistema web basado en conocimiento pretende ayudar al equipo de desarrollo en la administración de la documentación del proyecto de software.

Hasta el día de hoy se han propuesto diferentes técnicas y herramientas para hacer la trazabilidad de los requisitos. Una técnica, es la matriz de trazabilidad que muestra la relación que existe entre requisitos y casos de uso, en (Tabares, Arango, & Anaya, 2006). Esta matriz se describe como la técnica más común y aplicable a cualquier modelo de

desarrollo que permite verificar en qué casos de uso son representados y especificados los requisitos funcionales obtenidos desde el espacio del problema.

Objetivo

Es importante para un equipo de desarrollo de software contar con una herramienta que les permita administrar y validar los requisitos, componentes y documentación del sistema en el que están trabajando y que, además, muestre la trazabilidad hacia adelante de requisitos y componentes, para tener un panorama de lo que ya implementaron, lo que hace falta implementar o los componentes que no implementan algún requisito y podría servir para futuras implementaciones, pruebas e incluso validación con el cliente.

El trabajo de esta tesis consiste en el desarrollo de un producto de software el cual es un sistema web basado en conocimiento llamado SEVETREQ que sigue el proceso de Implementación de Software (IS) de la norma ISO/IEC 29110 Perfil básico para verificar la existencia de los productos de entrada y salida en las actividades Análisis de Requisitos del Software y Arquitectura y diseño detallado de software. Además, como producto final, realiza la trazabilidad de requisitos por medio de una matriz de trazabilidad.

A través de SEVETREQ el equipo de desarrollo podrá dar seguimiento a la documentación del proyecto en el que está trabajando, administrar sus requisitos y componentes (clases y/o interfaces gráficas) y visualizar la trazabilidad hacia adelante para que pueda detectar omisiones fácilmente.

Alcance

Desarrollar un sistema basado en conocimiento que guíe a un equipo de desarrollo de software en las actividades de análisis de requisitos del software y de arquitectura y diseño detallado de software del proceso de Implementación de Software (IS) del perfil básico de la norma ISO/IEC 29110. El sistema realizará el registro de la trazabilidad indicado en la norma por medio de una matriz de trazabilidad que tome los requisitos obtenidos que el cliente proporciona y los componentes de tipo clase y/o interfaz gráfica establecidos por el equipo de desarrollo.

La metodología utilizada en el desarrollo del sistema consistirá en realizar las actividades propuestas en dicho proceso comenzando con la actividad de análisis de requisitos en la que se generará un documento de especificación de requisitos, posterior a eso, se realizará el documento de Diseño de software, basándose en los diagramas de componentes del documento anterior, se procederá con la construcción del software y por último, se probará en un sistema desarrollado en el posgrado para un cliente real.

La estructura de este trabajo es la siguiente: *en el capítulo 1 se presenta el marco teórico que permitirá establecer los fundamentos sobre los que se sustenta la tesis; en el capítulo 2 se describen las actividades de análisis de requisitos, así como arquitectura y diseño detallado de software de SEVETREQ; en el capítulo 3 se detalla el desarrollo del sistema*

basado en conocimiento de SEVETREQ; el capítulo 4 contiene la actividad de integración y pruebas de software de SEVETREQ; conclusiones y trabajo a futuro y por último los anexos.

Capítulo 1. Marco teórico

1.1 REQUISITOS DE SOFTWARE

Un *requisito de software* es una propiedad que debe ser exhibida por algo y que expresa las necesidades y limitaciones para contribuir a la solución de un problema del mundo real. (Bourque & Fairley, 2014)

Los requisitos son estudiados por la Ingeniería de requisitos que se encarga de la identificación de objetivos para un sistema propuesto, la operación y conversión de estos objetivos en servicios y limitaciones, así como la asignación de responsabilidades para los requisitos resultantes como humanos, dispositivos y software. Es considerada como uno de los escenarios más importantes en el diseño y desarrollo de software ya que aborda el problema fundamental de diseñar el software correcto para el cliente. (Melorose, Perroy, & Careas, 2005)

Partiendo de las clasificaciones propuestas por (Dooley, 2011) y (Fernandes & Machado, 2016), los requisitos se pueden dividir en las siguientes categorías:

- Requisitos funcionales que se clasifican en:

  - Requisitos de usuario o explícitos. Enunciados sobre lo que el usuario espera del sistema.
  - Requisitos del dominio o implícitos. Son impuestos por el dominio del sistema y normalmente no están especificados por el usuario.
  - Requisitos de sistema. Se documentan en un lenguaje más técnico y sirven de apoyo para el diseño y construcción del sistema.

-Requisitos no funcionales

Afectan la experiencia del usuario (cuando se habla de usabilidad, apariencia, velocidad, etc.) y el desarrollo cuando se habla de seguridad, disponibilidad, etc. No cambian la esencia de las funcionalidades del sistema y se deben aplicar como un todo.

-Próximos requisitos

Requisitos de usuario que no serán implementados en el sistema al menos en la entrega pactada.

1.2 TRAZABILIDAD DE REQUISITOS

Un producto de software es maleable y tiene la capacidad de cambiar su estructura y composición durante su proceso de desarrollo, para conocer su evolución durante dicho proceso la trazabilidad juega un papel importante (Tabares, Arango, & Anaya, 2006), ya

que para el equipo de desarrollo es importante conocer las relaciones que existen entre la información proporcionada (Pinheiro, 2003) y lo que ya se ha desarrollado o se pretende desarrollar.

La *trazabilidad de requisitos* ha sido descrita como una medida de la calidad del sistema (Roetzheim, 1991) y se define como la habilidad de describir y seguir la vida de un requisito hacia atrás y hacia adelante (desde sus orígenes, a través de su desarrollo y especificación, hasta su despliegue y uso y a través de todos los periodos de refinamiento e iteración en cualquiera de estas fases) (Gotel & Finkelstein, An Analysis of the Requirements Traceability Problem., 1994).

Cuando un requisito es rastreado hacia atrás, significa que provino de un documento porque el requisito fue extraído directamente del documento o el documento contiene enunciados que apoyen al requisito. Cuando un requisito es rastreado hacia adelante, significa que existe un módulo que lo implementa o que se desarrolló un módulo para probar al requisito. (Pinheiro, 2003)

El estándar ANSI/IEEE 830-1984 establece que la especificación de requisitos es trazable si (ANSI/IEEE Standard, 1984):

a. El origen de cada requisito es claro
b. Si facilita la referencia de cada requisito en el desarrollo futuro o mejora de la documentación.

Según Edwards and Howell [1992] la trazabilidad es una técnica que provee una relación entre los requisitos, el diseño y la implementación final del sistema, permite a los desarrolladores demostrar al cliente que los requisitos han sido entendidos, el producto cumple con ellos y que no tiene características innecesarias; y los diseñadores pueden mostrar que el diseño cumple con los requisitos e identificar en etapas tempranas los requisitos que no se están satisfaciendo. (Ramesh, Stubbs, Powers, & Edwards, 1997)

1.2.1 Objetivos de la trazabilidad de requisitos

- Asegurar que el software producido cumple con las expectativas del usuario (Ramesh, Stubbs, Powers, & Edwards, 1997).
- Demostrar que cada requisito ha sido satisfecho. (Stehle, 1990)
- Demostrar que cada componente del sistema satisface un requisito. (Stehle, 1990)
- Indicar cómo se obtuvieron los requisitos y la razón del diseño donde se identifican no solo las decisiones, sino también las razones de apoyo y oposición detrás. (Ramesh & Dhar, 1992)
- Ayuda a evaluar el impacto que tendrá el desarrollo de los cambios al alcance ya que las relaciones entre productos de software pueden ser fácilmente identificadas.

1.2.2 Modos de trazabilidad de requisitos

En (Pinheiro, 2003) se presenta una clasificación para los modos de trazabilidad, con ellos es posible obtener diferente información de los requisitos obtenidos:

- Con respecto a la dirección de rastreo:
  - Trazabilidad hacia adelante. Es la habilidad de rastrear un requisito hacia sus componentes de diseño. Se obtiene contestando a la pregunta ¿De dónde viene?
  - Trazabilidad hacia atrás. Es la habilidad de rastrear un requisito hacia su fuente. Se obtiene contestando a la pregunta ¿Para cuál componente?
- Con respecto a su evolución:
  - Trazabilidad pre-RS (*pre-Requirements Specification*, en español, pre-especificación de requisitos). Se describen las razones y aspectos que generaron al requisito. Se obtiene respondiendo a la pregunta ¿Cómo fue producido?
  - Trazabilidad post-RS (*post-Requirements Specification*, en español, post-especificación de requisitos). Se describe el uso que se le dará al requisito. Se obtiene respondiendo a la pregunta ¿Cómo fue usado?
- Con respecto al tipo de objetos involucrados:
  - Trazabilidad inter-requisitos. Es la habilidad de rastrear las relaciones entre requisitos ya que al refinarlo y/o parafrasearlo, aparecen nuevos, derivados e incluso algunos son abandonados. Este modo de trazabilidad muestra los requisitos que provienen de otros requisitos.
  - Trazabilidad extra-requisitos. Muestra la relación entre requisitos y artefactos de software.

Para obtener una trazabilidad de requisitos completa, es útil combinar estos modos ya que puede no ser suficiente con saber quién proporcionó el requisito (trazabilidad hacia atrás), también sería bueno saber las razones por las que se incluye (trazabilidad pre-RS).

1.2.3 MODELO DE TRAZABILIDAD DE REQUISITOS

A pesar de que se conocen los beneficios de la trazabilidad y que los niveles de madurez dictan su uso, aún no hay un acuerdo oficial sobre sus aspectos principales. Por ejemplo, una organización puede definir que las líneas de trazado son extraídas manualmente y mantenidas en una hoja de cálculo, mientras que en otra se registran los requisitos y sus componentes relacionados en un servidor para que las líneas de trazado se actualicen automáticamente. Esta heterogeneidad impide que las prácticas de trazabilidad estén unificadas en las organizaciones y, en consecuencia, que el proceso de trazabilidad se lleve a cabo. (Marques, Ramalho, & Andrade, 2015)

Para ejecutar el proceso de la trazabilidad de una manera automatizada, se tienen que definir modelos de trazabilidad.

Un *modelo de trazabilidad* es el componente central de un ambiente de trazabilidad alrededor del cual los procedimientos de trazado, métodos y herramientas están organizados. (Pinheiro, 2003)

Se deben cubrir tres aspectos en un modelo de trazabilidad: (Pinheiro, 2003)

1. Definición. Un modelo de trazabilidad debe establecer los elementos que serán trazados (unidades de traza) y cómo se representarán en el modelo. También debe definir los diferentes tipos de trazas que lo componen, qué representan cada uno, qué unidades de traza relacionarán y en qué condiciones se deben registrar las trazas.
2. Producción. Se generan las trazas por medio de estrategias y técnicas definidas. La producción de trazas implica su comprensión, registro y mantenimiento. No es suficiente crear trazas y definir maneras en las que deben ser registradas; una traza debe ser entendida cuando suceda, de otra manera, pasará desapercibida y no será registrada.

   La descripción de las trazas por un modelo de trazabilidad debe ser similar a la manera en que las trazas ocurren en el mundo real. Si hay discrepancias entre lo que está especificado en el modelo y lo que se capturó, primero se podría terminar registrando lo que no hay y segundo, recuperando lo que nunca pasó.
3. Extracción. Un modelo de trazabilidad debe proveer diferentes y flexibles maneras de recuperar la información registrada en ella, para que se pueda elegir la más apropiada para cada ocasión. Existen tres modos de traza que proporcionan ayuda en términos de mecanismos de extracción:
   - Traza selectiva: para restringir la traza a ciertos patrones de objetos y relaciones o ciertos contextos, por ejemplo, restringir la traza a la fase de desarrollo o considerar solo clases de objetos y relaciones.
   - Traza interactiva: para permitir búsqueda interactiva a través de objetos relacionados con cada paso siendo guiado por las posibles relaciones. Se puede trazar hacia adelante o hacia atrás desde un objeto y probar diferentes rutas siguiendo las relaciones entre objetos.
   - Traza no guiada: para permitir al usuario ir de un objeto a otro a voluntad, revisando contenidos cuando se desee. Es conveniente cuando se tiene poca información en qué y cómo trazar.

1.3 ESTADO DEL ARTE EN TRAZABILIDAD DE REQUISITOS

Dada la importancia de la trazabilidad en el desarrollo de proyectos de software, se han hecho varias propuestas para ayudar a los equipos de trabajo a realizar un seguimiento de los requisitos y poder conocer el impacto que tiene cada uno en el proyecto a desarrollar.

Por ejemplo, en el trabajo de (Oh & Kang, 2014) se presenta un modelo formal de trazabilidad que le permite al arquitecto establecer y entender las trazas en diferentes niveles de una manera consistente. En el artículo, se establece que en un modelo de trazabilidad las decisiones de diseño deben ser expresadas explícitamente para mostrar cómo la arquitectura alcanza los atributos de calidad dados. Durante el diseño, estas decisiones de diseño descomponen los elementos arquitectónicos y asignan responsabilidades a los elementos descompuestos, con respecto a los atributos de calidad que deben ser realizados. A través de múltiples descomposiciones, una jerarquía de decisiones es creada, mostrando el proceso de cómo los elementos arquitectónicos de más bajo nivel son derivados de los requisitos a nivel sistema.

También se encuentra la metodología de (Agarwal, Chetwani, & Ravindra et al., 2015), SoRDeTT que está basada en plantillas para extraer requisitos y elementos de diseño y verificar la relación entre estos.

En el trabajo de (Sengupta, Kanjilal, & Bhattacharya, 2008) se propone un conjunto de reglas para establecer un trazado entre los requisitos funcionales en SRS (Especificación de Requisitos de Software, por sus siglas en inglés) y algunos de los diagramas UML comúnmente usados para modelar especificaciones de diseño usando un framework basado en XML y la notación Z.

Por otro lado, se encuentra la herramienta *End-To-End Industrial Software Traceability Tool* (Asuncion, Francois, & Taylor, 2007) que ayuda a realizar una trazabilidad general de los requisitos a través de todo el ciclo de vida (SDLC) de un proyecto de desarrollo. La herramienta minimiza la sobrecarga de trabajo que implica establecer y mantener trazas al proveer búsquedas por palabras clave entre artefactos y la actualización automática de las trazas; además ayuda a preservar la integridad de los documentos con una función en la que se hacen actualizaciones de artefactos y documentos de manera bidireccional, es decir, al agregar un documento, los artefactos son guardados en la base de datos y al realizar algún cambio en los artefactos, el documento es actualizado. También proporciona la búsqueda por palabras clave a través de los artefactos existentes para que el usuario pueda mantener las relaciones entre ellos.

En (Cleland-Huang, Zemont, Lukasik, & Wiktor, 2004) se propone un enfoque para la trazabilidad en el que se seleccionaron y aplicaron técnicas heterogéneas de acuerdo con las características y necesidades de trazabilidad para cada requisito. También se presenta un *framework* llamado TraCS para integrar las técnicas heterogéneas de trazabilidad.

En el trabajo (Filho, Lencastre, & Rodrigues, 2013) se presenta la herramienta RETRATOS que aborda cuatro cuestiones de la trazabilidad: ¿Qué información debería ser capturada?, ¿Cómo debería ser capturada la información de trazabilidad? ¿Cómo debería ser almacenada la información de trazabilidad? y ¿Cómo serán vistas y consultadas las relaciones de trazabilidad?

La herramienta realiza la generación automática de relaciones de trazabilidad entre modelos de software heterogéneos (BPMN, UML, código Java) e identifica elementos faltantes en dichos modelos, todo esto para sistemas multi agente.

Recientemente se desarrolló la herramienta MultiVisioTrace (Rodrigues, Lencastre, & De Cysneiros Filho, 2016) para proveer acceso a diferentes tipos de técnicas de visualización considerando el alcance de trazabilidad, ofrece flexibilidad en la elección de la técnica de visualización más apropiada para el contexto en el cual la trazabilidad está siendo aplicada.

Dadas las herramientas, modelos y metodologías anteriores, se encontró que SEVETREQ presenta algunas desventajas las cuales son: la incapacidad de realizar trazabilidad a lo largo del ciclo de vida del desarrollo ya que SEVETREQ proporciona trazabilidad en la fase de diseño; la necesidad de la acción humana para agregar los artefactos a trazar. Dentro de las similitudes se encuentran funciones para localizar artefactos y mantener las relaciones; y como ventajas se tiene que SEVETREQ permite realizar trazabilidad a cualquier tipo de software además de servir de apoyo en su documentación gracias a que está basado en el perfil básico de la norma ISO/IEC 29110.

1.4 NORMA ISO/IEC 29110. INGENIERÍA DE SOFTWARE – PERFILES DE CICLO DE VIDA PARA EMPRESAS MUY PEQUEÑAS

De acuerdo con la guía (CNFBCNA-INDECOPI, 2012) la industria de software reconoce el valor de las empresas muy pequeñas (VSEs por sus siglas en inglés) porque contribuyen con productos y servicios. Para el propósito de la ISO/IEC 29110, una VSE es una entidad con a lo más 25 personas.

Se ha encontrado que es difícil para las VSEs relacionar los estándares internacionales con sus necesidades de negocio y justificar su aplicación a las prácticas de negocio. Muchas VSEs tampoco pueden costear los recursos, en términos de número de empleados, presupuesto y tiempo, ni ven un beneficio neto en el establecimiento de procesos de ciclo de vida del software.

Para enmendar algunas de esas dificultades, se ha desarrollado un conjunto de guías de acuerdo con un conjunto de características de VSEs. Las guías están basadas en subconjuntos de elementos de normas apropiados, referidos como perfiles VSEs. El propósito de un perfil VSE es definir un subconjunto de estándares internacionales relevantes al contexto de la VSE.

Usando el Perfil básico de la norma, una VSE puede obtener los siguientes beneficios:

- Se realiza un proceso de administración disciplinado que provee visibilidad del proyecto y acciones correctivas a problemas y desviaciones del proyecto.
- Se obtiene un conjunto acordado de requisitos y productos esperados que se entrega al cliente.

- Se sigue un proceso sistemático de implementación de software que satisface las necesidades del cliente y asegura la calidad de los productos.

Para usar la guía, la VSE debe cumplir las siguientes condiciones de entrada:

- El enunciado de trabajo del proyecto está documentado.
- Se llevó a cabo la factibilidad del proyecto antes de comenzar.
- Se ha asignado y entrenado el equipo y líder del proyecto.
- Están disponibles para empezar el proyecto bienes, servicios e infraestructura.

El desarrollo de la solución expuesta en la presente tesis se enfocará en el proceso de Implementación de Software (IS) por lo que se revisará con mayor detalle.

El propósito del proceso de SI es la ejecución sistemática de las actividades de análisis, diseño, construcción, integración y pruebas para productos de software nuevos o modificados de acuerdo con los requisitos especificados.

La ejecución del proceso de SI es guiada por el plan del proyecto. El proceso empieza con una actividad de iniciación de la revisión del plan del proyecto. El plan del proyecto guiará la ejecución de las actividades de análisis de requisitos, diseño arquitectónico y detallado de software, construcción de software, integración de software, pruebas y entrega del producto.

A continuación, se muestra el diagrama que representa al proceso:

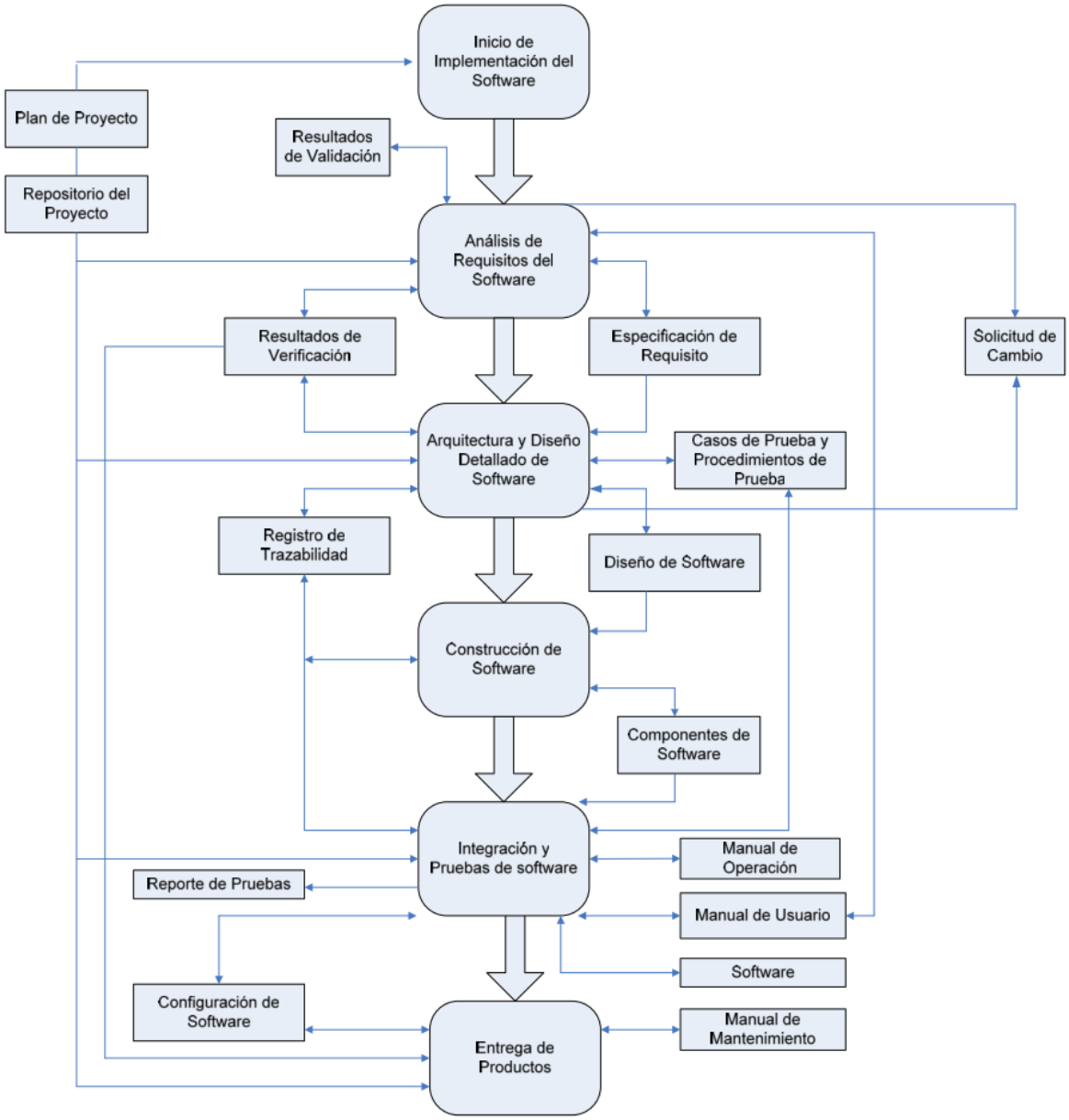


Figura 1.1. Diagrama del proceso de Implementación de software. (CNFBCNA-INDECOPI, 2012)

El proceso IS consiste en las siguientes actividades:

- IS.1 Inicio de la Implementación de Software. Asegura que el plan del proyecto establecido en la actividad planeación del proyecto es llevado a cabo por el equipo de trabajo.
- IS.2 Análisis de Requisitos del Software. Analiza los requisitos acordados con el cliente y establece los requisitos del proyecto validados.
- IS.3 Arquitectura y Diseño Detallado del Software. Transforma los requisitos de *Software* en la arquitectura *Software* del sistema y en el diseño detallado del *Software*.

- IS.4 Construcción del Software. Desarrolla el código y los datos del Software a partir del diseño de *Software*.
- IS.5 Integración y Pruebas del *Software*. Asegura que los componentes de *Software* integrados satisfacen los requisitos del *Software*. La
- IS.6 Entrega del Producto. Provee el producto de *Software* integrado al cliente.

Cada actividad cuenta con una lista de tareas, la presente tesis se concentra en las actividades IS2 e IS3, en las tablas 1 y 2 se muestran sus respectivas tareas.

Tareas de la actividad IS2 Análisis de Requisitos del Software
Asignar tareas a los miembros del equipo de trabajo
Comprender la especificación de requisitos
Verificar y obtener la aprobación de la especificación de requisitos
Validar y obtener la aprobación de la especificación de requisitos
Documentar la versión preliminar del manual de usuario o actualizar el manual existente
Verificar y obtener la aprobación del manual de usuario, si es apropiado
Incorporar la especificación de requisitos y el manual de usuario a la configuración de software

Tabla 1. Lista de tareas de la actividad de análisis de requisitos de software.

Tareas de la actividad IS3 Arquitectura y Diseño Detallado del Software
Asignar tareas a los miembros del equipo de trabajo
Comprender la especificación de requisitos
Documentar o actualizar el diseño de software
Verificar y obtener la aprobación del diseño de software
Establecer o actualizar los casos de prueba para pruebas de integración basadas en la especificación de requisitos y el diseño de software
Verificar y obtener la aprobación de los casos de prueba y procedimientos de prueba
Actualizar el registro de trazabilidad incorporando los casos de prueba y procedimientos de prueba
Incorporar el diseño de software, y el registro de trazabildiad a la configuración de software

Tabla 2. Lista de tareas de la actividad de Arquitectura y Diseño Detallado del Software

1.5 SISTEMAS BASADOS EN CONOCIMIENTO

En nuestra interacción con el mundo, hacemos uso de información adquirida por nuestros procesos perceptuales y que está almacenada en nuestra memoria a largo plazo.

Aunque los términos información y conocimiento no son sinónimos, podemos decir que la información hace referencia al mundo que es externo al sistema cognitivo, mientras que el conocimiento es la información que ha sido procesada y almacenada. (Saltiveri, Vidal, & Delgado, 2005; Saltiveri, Vidal, & Delgado, 2005)

La representación del conocimiento es el corazón del gran reto de la Inteligencia Artificial: para entender la naturaleza de la inteligencia y la cognición tan bien que las computadoras puedan ser hechas para exhibir las habilidades humanas. En 1958, John McCarthy pensó en sistemas de Inteligencia Artificial que podrían ejercitar el sentido común. A partir de este y otros trabajos, los investigadores se convencieron de que la Inteligencia (artificial) podría ser formalizada por razonamiento simbólico con representaciones explícitas del conocimiento y, que el reto es idear una manera de representar el conocimiento en computadoras y usarlo algorítmicamente para resolver problemas. (Frank, Vladimir, & Bruce, 2008)

Uno de los objetivos de la inteligencia artificial es desarrollar sistemas que exhiban el comportamiento inteligente de los humanos. Intentos iniciales en el campo (década de 1960) fue crear sistemas inteligentes que pudieran realizar tareas en una variedad de dominios. Sin embargo, los investigadores se dieron cuenta que desarrollar sistemas de propósito general era muy difícil y era mejor enfocarse en sistemas limitados por dominios. Edward Feigenbaum de la Universidad de Standford propuso la noción de sistemas basados en conocimiento. (Anjaneyulu, 1998)

Los Sistemas Basados en el Conocimiento (SBK) o Sistemas Expertos (SE) son una parte de la Inteligencia Artificial. Son sistemas basados en computadora que integran bases de datos, memorias, mecanismos de razonamiento, agentes, algoritmos, heurísticas, para adquirir representar, almacenar, generar y difundir conocimientos, inicialmente adquiridos a través de varios expertos humanos dentro de un dominio específico llamado nube. (Marcellin Jacques, 2014)

Los SBK recolectan pequeños fragmentos de conocimiento humano en un conocimiento base, el cual es usado para razonar un problema, usando el conocimiento que es apropiado. Una importante ventaja aquí es que dentro del dominio del conocimiento base, un problema diferente puede ser resuelto usando el mismo programa (Grosan & Abraham, 2011) sin tener que construirlo de nuevo a diferencia de un programa convencional cuyo conocimiento base está embebido en el código y si el conocimiento cambia, el código también tendría que ser modificado.

1.5.1 Ventajas de los sistemas basados en conocimiento

A continuación, se listan algunas ventajas de los sistemas basados en conocimiento (Grosan & Abraham, 2011) y (Marcellin Jacques, 2014):

- Habilidad para capturar y preservar experiencia humana irremplazable.
- Habilidad para desarrollar un sistema más consistente que los expertos humanos.
- Minimizar la pericia humana necesaria en un número de ubicaciones al mismo tiempo (especialmente en ambientes peligrosos para la salud humana).
- Se pueden desarrollar soluciones más rápido que los expertos humanos.
- Capacidad de explicar acciones y elecciones realizadas.
- Cuenta con la capacidad de manejar conocimiento en forma dinámica que se pueden obtener de diferentes fuentes (bases de datos, sensores, agentes externos, videos, textos, redes sociales, cámaras, etc.).
- Cuenta con uno a varios modelos para representar el conocimiento y experiencia sobre el área de domino (la nube).
- Cuenta con la posibilidad de aprender, es decir, de agregar conocimiento en base a sus operaciones y que no tenía al inicio de su operación.
- Cuenta con habilidad para razonar por medio de diferentes métodos como son (pero no limitados a) deducción, inducción y abducción.

1.5.2 COMPONENTES DE UN SISTEMA BASADO EN CONOCIMIENTO

Un sistema basado en conocimiento consiste en los siguientes componentes:

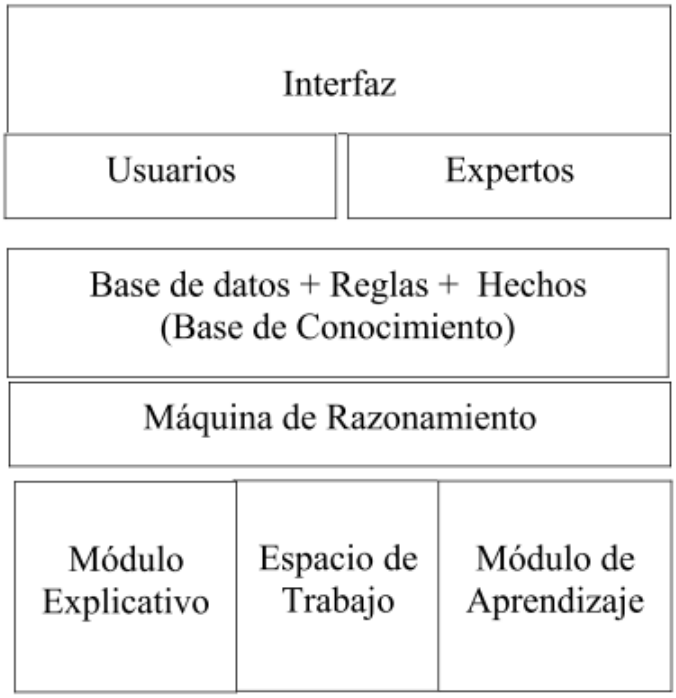


Figura 1.2. Componentes de un sistema basado en conocimiento. (Marcellin Jacques, 2014)

A continuación, se describe cada componente: (Marcellin Jacques, 2014)

- Interfaz con el usuario: permite aceptar y reconocer un lenguaje de comandos en forma intuitiva y los traduce en instrucciones y datos para que el sistema basado en conocimiento trabaje.
- Interfaz con el experto: permite captar información del exterior (proporcionada por el experto) e introducirla en forma adecuada en la base de conocimiento.
- Base de datos y reglas: contiene el saber específico en la disciplina de la cual el sistema es experto (nube). Consiste en un conjunto de hechos (datos) y de reglas programadas. Los datos plasmados en la base juegan un papel primordial en la calidad y en las habilidades de razonamiento del sistema.
- Máquina de razonamiento: también llamada máquina de inferencia, este módulo, permite controlar el sistema y manejar los razonamientos similares a los realizados por los expertos de la nube. Existen dos tipos de métodos de inferencia comúnmente usados:
    - Encadenamiento hacia atrás, es el proceso de empezar con las conclusiones e ir hacia atrás para apoyar los hechos. Por ejemplo, si se ve a alguien reparando un carro, se puede asumir que el carro no enciende. (Cong, Shin, & Salehnia)

- Encadenamiento hacia adelante, empieza con los hechos y trabaja hacia adelante para las conclusiones. Por ejemplo, si un carro no enciende, es necesario repararlo. (Cong, Shin, & Salehnia)

Las máquinas de inferencia basadas en sistemas de reglas consisten en reglas *if-then*, un conjunto de hechos y un intérprete que controla la aplicación de las reglas, dados los hechos. Estas reglas *if-then* son usadas para formular los enunciados condicionales que comprenden el conocimiento base completo. (Grosan & Abraham, 2011)

- Espacio de trabajo: Se almacenan resultados y el estado en que el problema se encuentra.
- Módulo explicativo: En este módulo se reflejan el o los porqués de las acciones del sistema. Explica sus decisiones.
- Módulo de aprendizaje: Permite agregar reglas y/o procesos que no estaban definidos inicialmente dentro del sistema.

1.5.3 LENGUAJES DE SISTEMAS BASADOS EN CONOCIMIENTO

Escribir un sistema basado en conocimiento requiere una gran cantidad de recursos en términos de tiempo y esfuerzo. Sería deseable usar programas existentes (paquetes) en los sistemas basados en conocimiento y modificarlos para nuevas aplicaciones. Pero algunas aplicaciones requieren características flexibles, los paquetes no pueden ofrecer soluciones a estos problemas. En este caso un lenguaje de sistemas basados en conocimiento poderoso sería mejor (Cong, Shin, & Salehnia) y la abstracción juega un papel importante en este sentido pues se puede crear un elemento que sea útil para diferentes aplicaciones, es decir, crear una herramienta de sistemas basados en conocimiento que pueda usarse para varios objetivos, la clave está en mantener la máquina de inferencia y el conocimiento base separados, este es el concepto para un s*hell* de sistemas basados en conocimiento; de esta manera, una persona que usa un s*hell*, no necesita ser un programador, y solo necesita agregar el conocimiento base. (Cong, Shin, & Salehnia)

OPS5, CLIPS, LOOPS, LISP y PROLOG son lenguajes de sistemas basados en conocimiento s populares, para este trabajo de tesis se dará énfasis a CLIPS ya que fue el lenguaje utilizado en el desarrollo del sistema.

1.5.4 CLIPS

Es una herramienta de sistema basado en conocimiento desarrollada por SBT (*Software Technology Branch,* por sus siglas en inglés, en la *NASA*). Está diseñado para facilitar el desarrollo de software para modelar el conocimiento o pericia humana. También ha sido diseñado para la integración con otros lenguajes como C y Java. (Giarratano, 2015)

En la guía de usuario (Giarratano, 2015), se presentan las tres maneras de representar el conocimiento en CLIPS:

- Reglas. Destinadas al conocimiento heurístico basado en la experiencia.
- *Deffunctions* y funciones generales. Destinadas principalmente al conocimiento procedimental.
- Programación orientada a objetos. También destinada a programación procedimental. Soporta las características de la programación orientada a objetos: clases, manejador de mensajes, abstracción, encapsulación, herencia y polimorfismo. Las reglas pueden coincidir con el patrón de objetos y hechos.

El s*hell* de CLIPS provee los elementos básicos de un sistema basado en conocimiento: (Giarratano, 2015)

- Lista de hechos (*fact-list*) y lista de instancias (*instance-list*): memoria global para datos.
- Conocimiento base (*knowledge-base*): contiene todas las reglas, las reglas base (*rule-base*):
- Máquina de inferencia: controla la ejecución general de las reglas.

Los programas desarrollados en CLIPS consisten en reglas, hechos y objetos. Para el desarrollo del sistema tratado en esta tesis, solo se utilizaron los dos primeros términos, por lo tanto, se cubrirán a continuación usando como referencia a (Cubero & Berzal), para mayor referencia sobre "objetos" de CLIPS, consultar (Giarratano, 2015).

Hechos

Se trata de los datos almacenados en la lista de hechos (*fact-list*), se pueden insertar como vectores ordenados de características o como registros.

Los vectores ordenados de características (VOC) son sentencias formadas por valores ordenados que corresponden a un atributo definido por el programador. Los VOC pueden llevar o no un nombre para indicar la relación que hay entre los atributos. Un ejemplo de un VOC sin nombre de relación sería:

(Pedro 45 M NO)

Donde el valor "Pedro" corresponde al atributo "Nombre", el valor "45" al atributo "Edad", el valor "M" al atributo "Sexo" y el valor "NO" al atributo "EstadoCivil".

El VOC anterior con nombre de relación se vería de la siguiente forma:

(Persona Pedro 45 M NO)

Donde el valor "Persona" es el nombre de relación, el resto de los valores tienen el significado explicado anteriormente.

Los registros permiten insertar hechos de una manera más ordenada, primero se debe definir el tipo de registro a insertar por medio de plantillas (*templates*), que se forman de un nombre de relación y uno o más nombres de atributos.

Plantilla:

```
(deftemplate nombreDeRelación
             (field nombreDeAtributo1)
             …
             (field nombreDeAtributoN)  )
```

Donde "deftemplate" es una palabra reservada para establecer el nombre de relación y "field" es una palabra clave para establecer un atributo del registro asignándole un nombre. Cuando el atributo se compone de más de un valor, se utiliza la palabra reservada "multifield.

Una vez definida la plantilla, se procede a insertar tantos registros como el programador lo requiera:

Registro:

```
(assert (nombreDeRelaciónEnPlantilla
          (nombreDeAtributo1  valor)
          …
          (nombreDeAtributoN  valor)))
```

Donde "assert" es una palabra reservada para insertar el registro en la lista de hechos.

Por ejemplo, para insertar un hecho Persona, primero se debe definir la plantilla Persona y después se generan los registros:

```
(deftemplate Persona
             (field Nombre)
             (field Edad)
             (field Sexo)
             (field EstadoCivil) )
(assert (Persona
        (Nombre Juan)
        (Edad 30)
        (EstadoCivil casado)
        (Sexo V)))
```

Reglas

Una regla en CLIPS es similar a los enunciados "IF THEN" en el lenguaje procedimental como JAVA, C o Ada, la sintaxis es la siguiente

```
(defrule <nombre>  <comentario>
             <patrón antecedente>
```

=>
<consecuente>)

La parte izquierda de la regla (donde se encuentra el patrón antecedente) suele llamarse LHS (*Left Hand Side*), mientras que la parte derecha es RHS (*Right Hand Side*).

La LHS está constituida por varios patrones. En ellos se establecen las condiciones que han de darse sobre los elementos de la memoria de trabajo para que la regla pueda activarse.

Los consecuentes pueden consistir en la adición o supresión de un elemento a la memoria de trabajo (ejecutando assert o retract respectivamente) o la ejecución de algún procedimiento (como, por ejemplo, printout, usado para imprimir mensajes en consola).

Por ejemplo, sea una regla en la que se buscan varones solteros, de 45 años y que se llamen "Pedro", como consecuente, deberá imprimir "Pedro está soltero":

```
(defrule ECivilPedro_Soltero
        (Persona Pedro 45 V NO)
        =>
        (printout t crlf "Pedro está soltero"))
```

1.5.5 Herramientas para el desarrollo de sistemas basados en conocimiento

Actualmente se encuentran disponibles varias herramientas y lenguajes para el desarrollo de aplicaciones web con sistemas basados en conocimiento. Estas herramientas usan técnicas de sistemas basados en conocimiento y tienen características útiles para crear sistemas web.

- Jess. Es un motor de reglas e intérprete para el lenguaje de reglas Jess, su sintaxis es similar a Lisp, fácil de aprender y muy adecuado para definir reglas y realizar programación procedimental. Fue desarrollado en *Sandia National Laboratories* en Livermore, California a finales de la década de 1990.
  Jess es principalmente una biblioteca que puede ser embebida en otro software basado en Java, sin embargo, también cuenta con un ambiente interactivo.
  Se puede descargar una licencia de la página web de Jess http://herzberg.ca.sandia.gov/jess ahí se encuentra una versión de prueba o se puede solicitar la versión completa, esta es gratis para uso académico, pero también se puede adquirir para uso comercial con un cargo especial. (Friedman-Hill, 2003)
- ERESYE. Herramienta para la realización de sistemas basados en conocimiento inteligentes usando el lenguaje Erlang. ERESYE es un sistema de producción de reglas que permite escribir reglas como cláusulas de función Erlang, proveyendo apoyo para su ejecución. Soporta conceptos orientados a objetos y ontologías gracias a que su herramienta de manejo de ontología provee medios para traducir conceptos basados en objetos a la forma Erlang. (Di Stefano, Gangemi, & Santoro, 2005)
- ExSys. Incorpora razonamiento basado en reglas y difuso. Una componente de ejecución basado en internet permite el diseño de tipo cliente-servidor a través de

scripts CGI y puede integrar otros componentes para el servidor. (Grosan & Abraham, 2011)

Capítulo 2. Proceso de implementación de software en SEVETREQ

En este capítulo y los siguientes se describirá el sistema SEVETREQ, así como la forma en la que se llevaron a cabo las tareas del proceso IS para su desarrollo.

2.1 DESCRIPCIÓN DE SEVETREQ

SEVETREQ es un sistema web basado en conocimiento cuyo objetivo es mostrar al equipo de desarrollo una matriz de trazabilidad con los requisitos y componentes (puede ser de tipo clase o interfaz) que previamente ingresaron siguiendo las tareas de las actividades de análisis de requisitos y diseño de software del IS que se explica en los siguientes párrafos.

El proceso IS comienza con la actividad de "Inicio de la implementación de software", SEVETREQ da por hecho que las tareas indicadas en esta actividad ya se han realizado, es decir, SEVETREQ asume que el equipo ya cuenta con un plan de proyecto el cual ya fue revisado por ellos, además, ya se se estableció un ambiente de implementación. Es necesario que el equipo lleve a cabo dichas tareas pues esto permitirá ejecutar ordenadamente en SEVETREQ las siguientes actividades del proceso.

Análisis de requisitos

La segunda actividad es Análisis de requisitos de software, el equipo de desarrollo podrá indicar a SEVETREQ dónde almacenará el documento de Especificación de requisitos, posterior a eso realizará el registro de todos los requisitos identificados, ingresando un alias, su descripción y si ya fue validado o no. La validación puede realizarse en cualquier momento.

Arquitectura y diseño detallado de software

La siguiente actividad es Arquitectura y diseño detallado de software, el equipo de desarrollo podrá indicar a SEVETREQ dónde almacenará el documento de Arquitectura y diseño de software, para después dar de alta los componentes que desarrollarán, la información necesaria para crear un componente en SEVETREQ es un alias, su descripción, el tipo (clase o interfaz) y si ya fue validado o no. La validación puede realizarse en cualquier momento.

Solicitudes de cambio

En proceso IS se hace referencia a la *solicitud de cambio* para los documentos de Especificación de requisitos, Diseño y arquitectura de software y manual de usuario. SEVETREQ contempla esta situación permitiendo generar solicitudes de cambio para los documentos de: Especificación de requisitos en donde se permitirá realizar cambios o

generar nuevos requisitos; Diseño de software para aplicar cambios o generar nuevos componentes y el documento de Manual de usuario, en el anexo "B. Documento de especificación requisitos de SEVETREQ" se explican las reglas de negocio de esta actividad.

En la figura 2.1 se encuentra un formato de solicitud de cambio sugerido para llevar el control de cada solicitud.

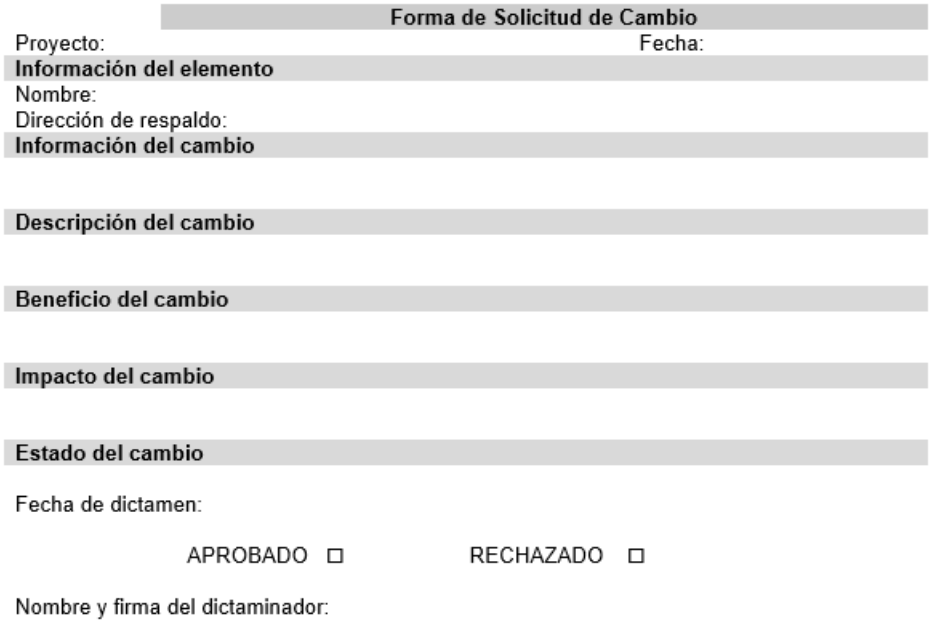


Figura 2.1. Formato de solicitud de cambios.

Registro de trazabilidad

Para llevar a cabo el registro de trazabilidad, el sistema SEVETREQ solicita los requisitos y componentes del proyecto en el que el equipo de desarrollo se encuentre trabajando para después, pedirle que indique los requisitos que cada componente implementa y así poder crear una matriz de trazabilidad donde se pueda ver si existe algún requisito sin implementar o algún componente que no implemente requisitos. Además, SEVETREQ muestra la lista de requisitos y/o componentes que le precedieron a cada uno.

2.2 ANÁLISIS DE REQUISITOS DE SEVETREQ

El desarrollo comenzó con el *Plan del proyecto* ubicado en el anexo "A.Documento de plan del proyecto" en donde se plasmó el enunciado de trabajo para definir el objetivo de SEVETREQ, además se estableció el cronograma de trabajo que guiaría el desarrollo del sistema.

Se continuó llevando a cabo la actividad de análisis de requisitos, que generó el documento de *Especificación de requisitos de SEVETREQ* donde se encuentran los casos de uso que describen su funcionamiento. A continuación, se presenta cada uno.

2.2.1 CASOS DE USO

En la figura 2.2 se muestra el Diagrama general de casos de uso y más adelante se describirá cada caso.

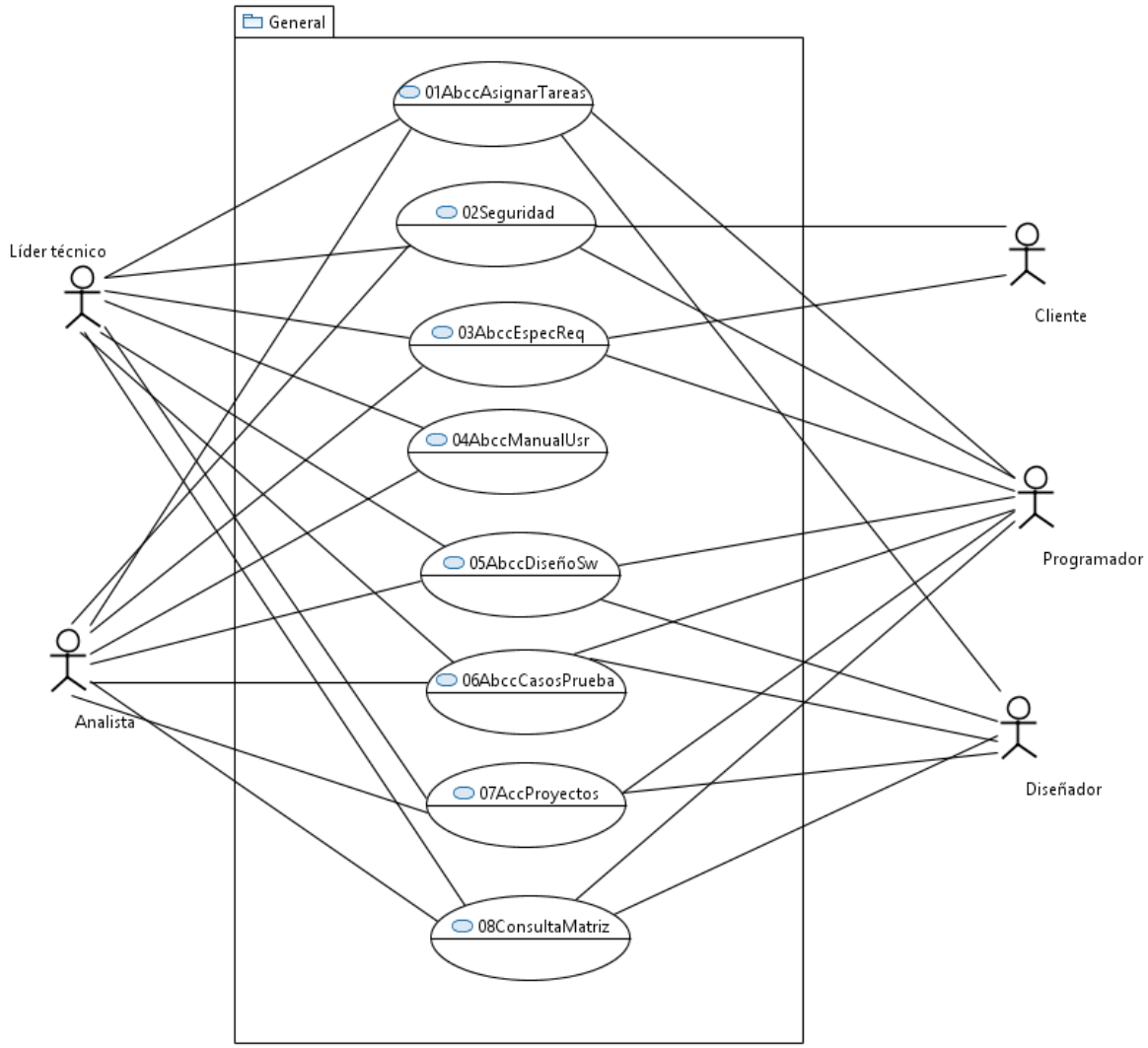


Figura 2.2. Diagrama general de casos de uso de SEVETREQ

A continuación, se realizará una breve descripción de los casos de uso generales de SEVETREQ, para una mayor referencia, se pueden consultar en el anexo "B.Documento de especificación de requisitos de SEVETREQ" en donde se encuentra el detalle de cada caso de uso ya que los que se encuentran en este diagrama no representan la totalidad de los requisitos de SEVETREQ.

Actores del sistema

Los actores considerados en SEVETREQ están basados en los propuestos por la norma y son:

- Líder técnico. Persona que forma parte del equipo de desarrollo con conocimiento y experiencia en el dominio del proceso de software.

- Analista. Integrante del equipo de desarrollo con conocimiento y experiencia que permita obtener, especificar y analizar los requisitos, diseño de interfaces de usuario y criterios ergonómicos, además de experiencia en desarrollo y mantenimiento de software.
- Cliente. Persona que cuenta con conocimiento y experiencia en el dominio de la aplicación, conoce los usuarios representativos para asegurar que el entorno operacional sea dirigido de forma correcta. Tiene autoridad para aprobar requisitos y sus cambios.
- Diseñador. Individuo del equipo de desarrollo que tiene conocimiento y experiencia en componentes de software y diseño de arquitectura, en la planificación y ejecución de pruebas de integración, así como experiencia en desarrollo y mantenimiento de software.
- Programador. Miembro del equipo de desarrollo que se encarga de realizar la programación de los componentes, debe tener experiencia en desarrollo y mantenimiento de software. En la norma se habla del rol de "equipo de trabajo", sin embargo, en esta tesis dicho rol se cambió por el de "programador" para delimitar a los actores.

Casos de uso del sistema

01AbccAsignarTareas

En este caso de uso, se realiza la administración[1] y asignación de tareas de análisis de requisitos y diseño detallado de software que el líder técnico y el gestor del proyecto establecieron en el plan del proyecto previamente generado. El líder técnico será quien capture y asigne aquellas que corresponden al área de requisitos, el líder técnico, analista y diseñador capturarán y asignarán las tareas del área de diseño para que el resto del equipo pueda consultar ambas.

02Seguridad

Se llevan a cabo acciones sobre la administración de permisos, así como la del inicio y cierre de sesión. Todos los actores tienen acceso en este caso de uso.

03AbccEspeqReq

Se indica la ruta del repositorio donde el equipo almacena el *Documento de especificación de requisitos* y se administran los requisitos que el analista extrae del o los documentos. Estos pueden estar en lenguaje natural y se sugiere que tengan un nivel de granularidad alto pues más adelante deberán ser relacionados con los componentes extraídos del documento de diseño. El analista será quien ingrese la ruta y los requisitos, posteriormente,

[1] La palabra "administración", en el contexto de esta tesis, se refiere a las acciones de alta, modificación, consulta y eliminación.

el cliente, líder técnico y él podrán consultarlos y validarlos. El resto del equipo solo podrá consultarlos.

Si existiera alguna solicitud de cambio para el documento de requisitos, SEVETREQ permite agregarla; cuando se hace un cambio en algún requisito, el sistema basado en conocimiento detecta qué solicitud realizó dicho cambio y genera un rastro para saber qué requisito fue modificado en dicha solicitud.

04AbccManualUsr

Se realizan las acciones de administración de la ubicación del repositorio donde el equipo almacena el manual de usuario (en caso de que exista) que el analista genera[2]. El analista y el líder técnico tendrán acceso a las acciones de este caso de uso.

05AbccDiseñoSw

Se indica la ruta del repositorio donde el equipo almacena el documento principal de *Diseño de software* y se administran los componentes que el diseñador y el analista extraen del documento. SEVETREQ pide al usuario un alias del componente que servirá para identificarlo en la matriz de trazabilidad. Una vez creado el componente, el usuario puede elegir los requisitos que este componente implementa. Tanto el analista como el diseñador serán los encargados de realizar las altas y modificaciones, el resto del equipo podrá consultar los componentes generados.

Si existiera alguna solicitud de cambio para el documento de diseño y/o componentes, SEVETREQ permite agregarla; cuando se hace un cambio en algún componente, el sistema basado en conocimiento detecta qué solicitud realizó dicho cambio y genera un rastro para saber qué componente fue modificado en dicha solicitud.

06AbccCasosPrueba

Se realizan las acciones de la administración del *Documento de casos de prueba* que el diseñador y analista generaron. Podrá ser consultado por el resto del equipo

07AltaProyectos

El líder técnico da de alta el proyecto del que se requiere trazabilidad de requisitos y seguimiento de documentos. El resto del equipo podrá consultarlo.

08ConsultarMatriz

Muestra la matriz de trazabilidad con todos los requisitos y componentes registrados previamente por los integrantes del equipo para identificar los requisitos sin implementar o los componentes que no tienen relación con ningún requisito. Cualquier integrante del equipo puede consultarla.

---

[2] En esta primera versión de SEVETREQ únicamente se solicitará la ubicación del documento en el repositorio del equipo, más adelante se pretende que el sistema almacene el documento.

2.2.2 Diseño de la base de datos

Para la base de datos se utilizó el gestor MySql Workbench 6.3, en la figura 2.3 se muestra el diagrama de la base de datos de SEVETREQ. Se explicarán las tablas a continuación, es importante indicar que la mayoría de las tablas (señaladas con *) contienen campos que para efectos prácticos se llamarán "básicos" y son: usuario que creó, usuario que modificó, fecha de creación, fecha de modificación; estos campos sirven para la trazabilidad de la información del propio SEVETREQ:

- **proyecto** *. Almacena la información básica del proyecto que incluye su nombre y los documentos de requisitos, diseño, casos de prueba y manual de usuario.
- **tipo_documento**. Contiene un catálogo de los tipos de documento sugeridos por la norma ISO/IEC 29110, estos son: Requisitos, Diseño, Manual de usuario, Casos de prueba. Si el equipo de desarrollo contara con más tipos de documento, se podrían agregar directo en la base de datos sin afectar el código.
- **sol_cambios***. En esta tabla se guardan las solicitudes de cambio para los documentos de requisitos, diseño y manual de usuario de cada proyecto.
- **usuario.** Esta tabla contiene nombre, nombre de usuario, contraseña y tipo de usuario de las personas que tendrán acceso al sistema.
- **documento***. Se guarda la ruta en el repositorio del equipo de los documentos generados para cada proyecto, indicando de qué tipo de documento se trata.
- **tipo_componente_diseni**o. Es un catálogo que almacena los tipos de componente que se manejarán en el desarrollo de un proyecto, en SEVETREQ se establecieron interfaz gráfica y clase, pero el sistema está preparado para almacenar otros tipos de componente.
- **tarea***. Contiene las tareas de análisis y diseño (y alguna etapa que el equipo detecte) que se planearon para un proyecto y las personas a las que se asigna cada una.
- "actividades_iso29110". Catálogo que guarda las actividades de la norma para ser relacionadas con las tareas guardadas en la tabla "tarea".
- **tipo_usuario**. Tabla que contiene el catálogo de roles de SEVETREQ, están basados en la norma y tienen diferentes niveles de accesibilidad dentro del sistema.
- **componente***. Tabla donde se guardan los componentes de cualquier tipo del proyecto en el que el equipo de desarrollo se encuentre trabajando.
  requisito*. Esta tabla almacena los requisitos de cada proyecto.
- **estado_validacion**. Tabla que sirve para almacenar el catálogo de los estados de validación por los que pasan los documentos y requisitos de un proyecto. Estos son: Sin validar, Validado.
- **rel_comp_req**. Se almacenan los requisitos asociados a los componentes, el bloque del sistema basado en conocimiento se apoya en esta tabla para generar la matriz de requisitos.

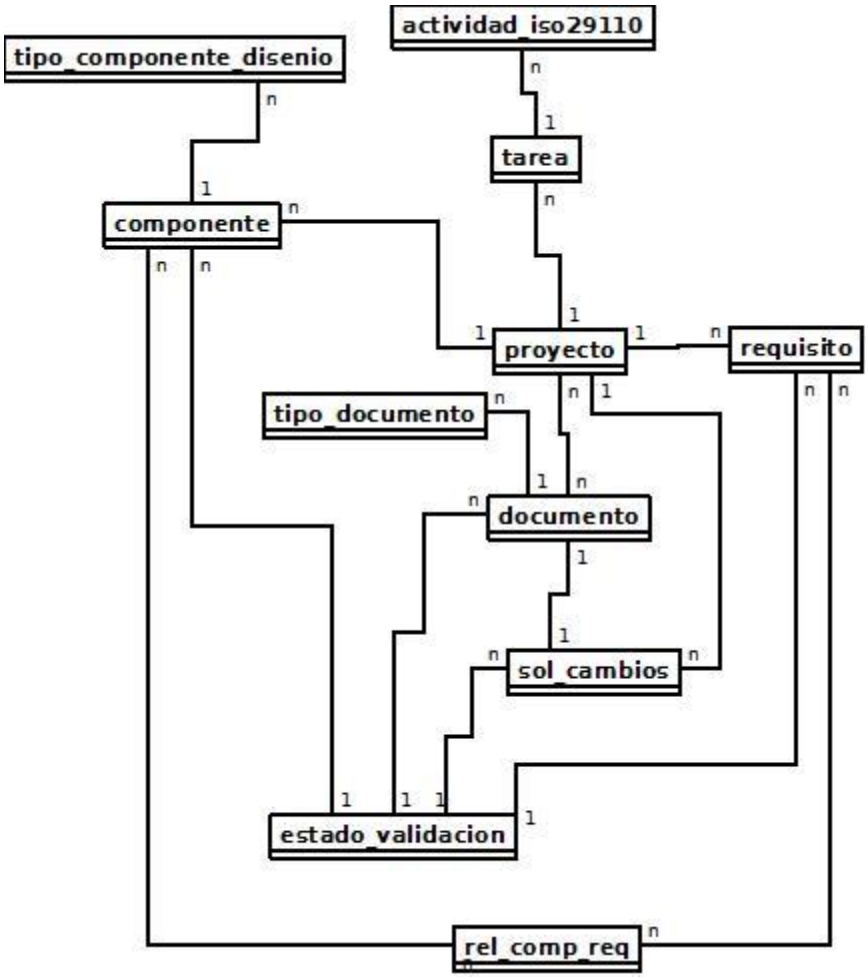


Figura 2.3. Diagrama de la base de datos del sistema SEVETREQ.

2.3 ARQUITECTURA Y DISEÑO DE SEVETREQ

Se continuó con lo establecido en la actividad de Arquitectura y diseño detallado de software, iniciando con el documento de *Diseño de software para SEVETREQ* en el que se establece la arquitectura a utilizar, las tecnologías, los diagramas de componentes de SEVETREQ y los prototipos de las interfaces. Se describe dicho documento de la siguiente manera.

Debido a que se trata de un sistema web se decidió desarrollar la parte de Java por medio del framework JSF que implementa la arquitectura de Modelo-Vista-Controlador (MVC).

En el diagrama de componentes se crearon paquetes para las clases del Modelo, para las clases del Controlador, para las clases del núcleo del sistema basado en conocimiento y para los archivos de las Vistas. En la figura 2.4 se muestra el Diagrama general de paquetes, más adelante se mostrarán los diagramas de cada paquete.

El desarrollo del sistema basado en conocimiento se hará por medio del lenguaje "CLIPS" utilizando la biblioteca "Jess". En el capítulo 3 se detalla el desarrollo de este paquete.

La capa del modelo se encontrará representada por un paquete llamado "modelos" y las clases que ahí se encuentren tendrán el prefijo "modelos". Estas clases son el mapeo de las tablas de la base de datos, por cada tabla, existirá una clase con el mismo nombre de la tabla que mapea.

Además, se ubicarán las clases cuyos métodos contienen las consultas a la base de datos. Los nombres de éstas deberán tener como prefijo la palabra "Repo".

Las clases de la capa del controlador se alojarán en el paquete "controladores" y se realizarán por medio de jsf, contendrá la lógica de negocios de SEVETREQ y se corresponden con las vistas de la capa de vista, es decir, cada vista tendrá una clase de controlador. El nombre de las clases de esta capa deberá tener como prefijo la palabra "controlador".

La capa de vista se desarrollará utilizando el framework Primefaces. Se considerarán los siguientes tipos de vistas:

1. Vista para agregar. Servirá para el código de los formularios de alta. Deberá nombrarse de la siguiente manera: agregarElemento, donde "Elemento" se refiere al elemento que agrega.
2. Vista para modificar. Servirá para el código de los formularios de modificación. Deberá nombrarse de la siguiente manera: modificarElemento, donde "Elemento" se refiere al elemento que agrega.
3. Vista para listar. Servirá para mostrar listas de elementos. Deberá nombrarse de la siguiente manera: "principalElemento", donde "Elemento" se refiere al elemento que lista.

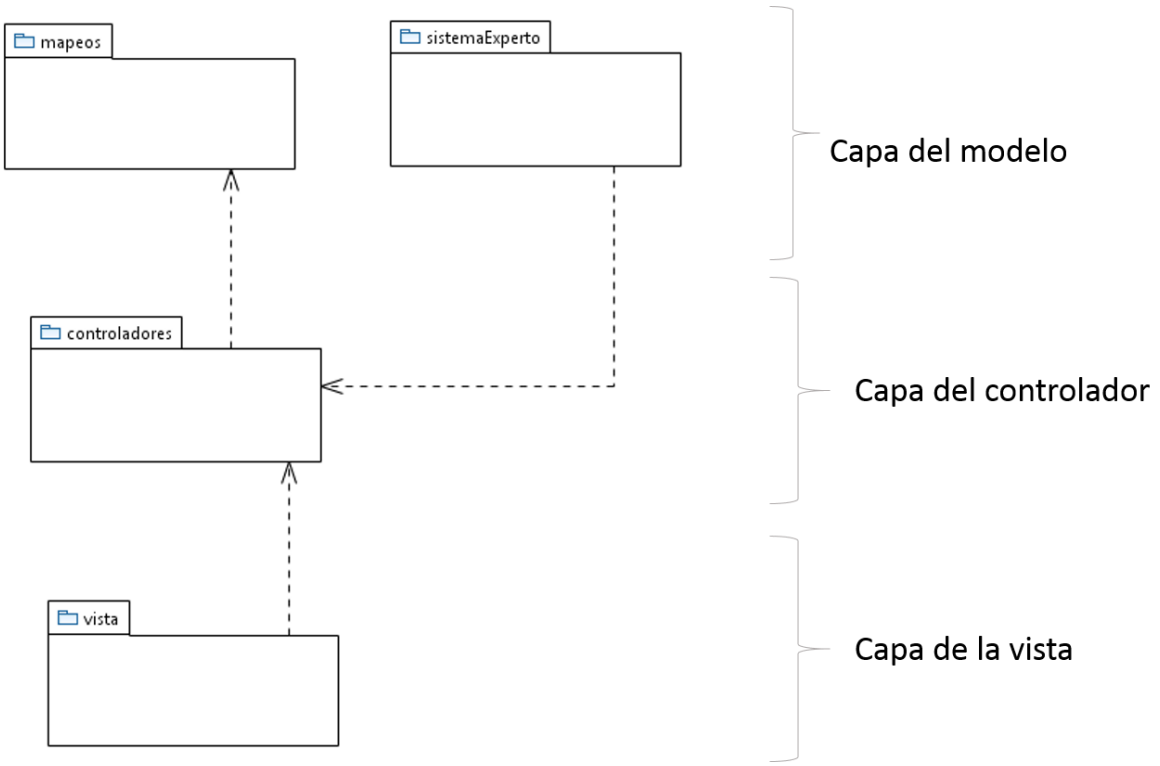


Figura 2.4. Diagrama general de paquetes del sistema SEVETREQ.

Paquete **mapeos**. Las clases generadas en este paquete fueron generadas por medio de la herramienta Hibernate[3], son mapeos de cada una de las tablas que se encuentran en la base de datos, por lo tanto, sus atributos se corresponden con los que se encuentran en las tablas. En la figura 2.5 se encuentran las clases correspondientes.

Figura 2.5. Diagrama de clases del paquete “mapeos”.

[3] Framework libre basado en ORM (object-relational mapping), el cual transfiere los datos entre la aplicación y la base de datos en forma de objetos.

Paquete **sistemaExperto**. En la figura 2.6 se encuentran las únicas clases y métodos que se usan para generar las reglas de inferencia del sistema basado en conocimiento que permiten la creación de la matriz de trazabilidad.

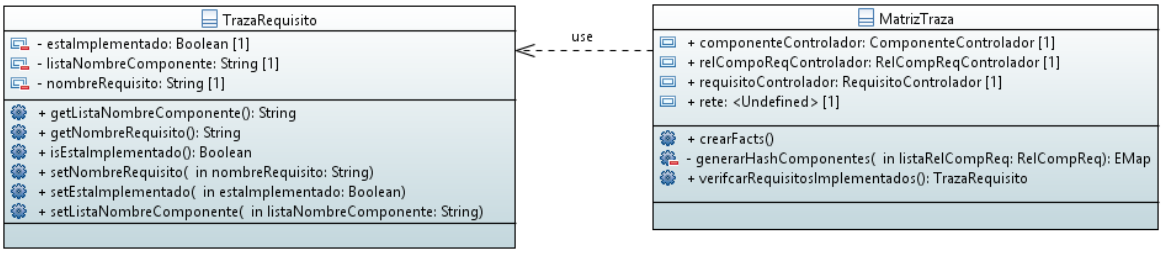


Figura 2.6. Diagrama de clases del paquete **sistemaExperto**. Las reglas se generan "al vuelo" en el método verificarRequisitosImplementados y los hechos en crearFacts también "al vuelo", este último guarda los hechos en la base de conocimiento.

En la clase **MatrizTraza** se encuentra el núcleo de la trazabilidad de SEVETREQ, exactamente en el método **verifcarRequisitosImplementados**, que envía una lista de objetos de tipo **TrazaRequisito**. Este objeto indica el nombre del requisito y los componentes en los que ha sido implementado. En el capítulo 3 se explicará el funcionamiento de estas clases.

Para conocer con más detalle el diseño de SEVETREQ, se puede consultar el anexo "Documento de diseño de software para SEVERTEQ".

2.4 CASOS DE PRUEBA

Según el proceso IS, otra de las tareas en la actividad de Arquitectura y diseño detallado es realizar casos y procedimientos de prueba. En esta fase del proceso de SEVETREQ se diseñaron casos de prueba con base en los casos de uso, a continuación, se muestra una tabla con los casos de pruebas más relevantes:

Caso de uso	**Descripción de la prueba**
01AbccAsignarTareas	Crear una tarea de la actividad de análisis ingresando un nombre de la tarea, nombre de la persona a quien se asigna y descripción.
01AbccAsignarTareas	Crear una tarea de la actividad de diseño agregando nombre de la tarea, nombre de la persona a quien se asigna y descripción.
02Seguridad	Iniciar sesión con un usuario registrado
03AbccEspeqReq	Capturar un requisito indicando su alias y descripción.
04AbccManualUsr	Para un proyecto que requiere manual de usuario, agregar la ruta del repositorio donde se encuentra alojado el documento de manual de usuario.

05AbccDiseñoSw	Crear un componente con un alias, descripción, tipo de componente y estado de validación.
05AbccDiseñoSw	Para un componente previamente agregado, indicar la lista de requisitos que implementa.
06AbccCasosPrueba	Agregar la ruta del repositorio donde se encuentra alojado el documento de casos de prueba.
07AccProyectos	Agregar un proyecto estableciendo un nombre, las rutas de la ubicación de los documentos de análisis y diseño, además indicar si contará o no con manual de usuario.
08ConsultaMatriz	Mostrar la matriz de trazabilidad de requisitos donde se observe la relación previamente establecida entre estos y los componentes.

De acuerdo con la norma los casos de prueba deben ser diseñados antes de la construcción del software, pero por estrategia, para el desarrollo de SEVETREQ, se decidió realizarlos al término de la construcción y así ejecutarlos inmediatamente.

Se realizaron pruebas de caja blanca y caja negra, ambas se resolvieron al momento de encontrarlas, por lo que no se llevó un conteo de las pruebas de caja blanca, sin embargo, sí se contaron las de caja negra y se encontraron alrededor de 20 errores.

En el anexo “Casos de prueba” se encuentra la lista completa de los casos de prueba que están agrupados por caso de uso. Cabe mencionar que todos los errores encontrados ya fueron solucionados.

Capítulo 3. Estructura del sistema basado en conocimiento de SEVETREQ

En este capítulo se presenta la estructura del sistema basado en conocimiento y de las reglas de inferencia diseñadas para darle a SEVETREQ la capacidad de generar la matriz de trazabilidad que muestra la relación entre componentes y requisitos.

3.1 Base de conocimiento de SEVETREQ

La base de conocimiento de SEVETREQ se compone de hechos los cuales son los datos de entrada del sistema basado en conocimiento y reglas de inferencia, enunciados condicionales que enlazan condiciones dadas con acciones o salidas.

Con base en la figura 3.1, el diagrama de los componentes básicos de los sistemas basados en conocimiento, se realiza un mapeo con los componentes de SEVETREQ para explicar su estructura.

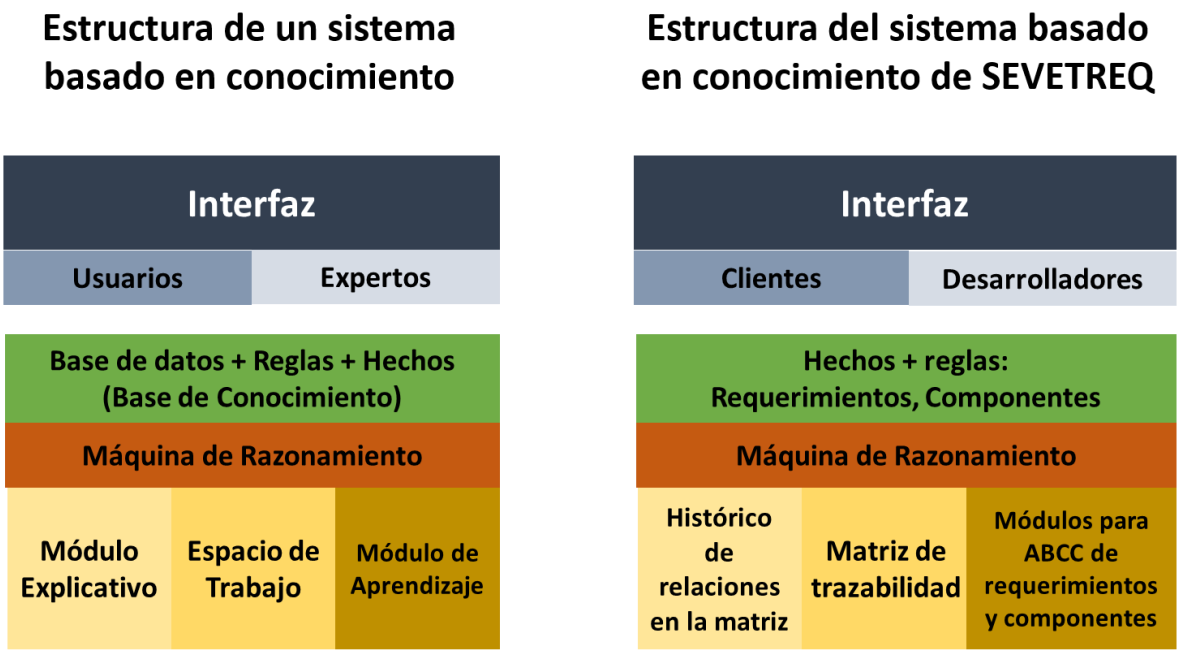


Figura 3.1. Mapeo de los componentes de un sistema basado en conocimiento con los componentes de SEVETREQ.

Esta parte del sistema se desarrolló con la biblioteca Jess, esto permitió realizar la programación con CLIPS por medio de Java.

En la siguiente tabla se describe el mapeo de la figura 3.1.

Componente de SE	Componente de SEVETREQ	Descripción del componente de SEVETREQ
Interfaz con el usuario	Interfaz de los clientes	SEVETREQ presenta una interfaz donde se listan todos los requisitos que solicitaron para el funcionamiento del sistema.
Interfaz con el experto	Interfaz de los desarrolladores	Permite al equipo de desarrollo ingresar todos los elementos que formarán parte del conocimiento de SEVETREQ, a saber: requisitos, componentes y la relación entre estos.
Base de conocimiento	Hechos + reglas: requisitos, componentes	En este bloque se generan los hechos y reglas de SEVETREQ, éstos se explicarán en la siguiente sección. En la figura 2.6 donde se encuentra el diagrama de clases para el SE de SEVETREQ, se puede encontrar el método" crearFacts" en la clase "MatrizTraza", el cual realiza la generación de hechos. En la misma clase se encuentra el método "verifcarRequisitosImplementados" que se encarga de establecer las reglas de inferencia.
Máquina de razonamiento	Máquina de razonamiento	La máquina de razonamiento de SEVETREQ se encuentra dentro del método "verifcarRequisitosImplementados" de la clase "MatrizTraza" de la figura 2.6, aquí se revisa cada una de las reglas parar generar una lista de tipo TrazaRequisito que contiene la información necesaria para crear la matriz de trazabilidad, es decir, el nombre de un requisito y la lista de componentes en los que se encuentra implementado.
Módulo explicativo	Histórico de las relaciones en la matriz	En cada requisito y componente que se encuentra en el renglón respectivo de la matriz, se aprecia la evolución que ha tenido dicho elemento durante el proceso de desarrollo.
Espacio de trabajo	Matriz de trazabilidad	Se trata de la generación de la matriz de trazabilidad, el cual se considera el resultado final, aunque no debe verse como un elemento estático ya que su composición puede cambiar si componentes, requisitos y/o las relaciones entre ellos cambian.
Módulo de aprendizaje	Módulos ABCC de requisitos y componentes	En este módulo se pueden agregar, eliminar y/o modificar requisitos y componentes para alimentar al sistema basado en conocimiento.

Tabla. Descripción del mapeo de los componentes de un sistema basado en conocimiento con los componentes de SEVETREQ.

La base de conocimiento de SEVETREQ que permite la generación de la matriz de trazabilidad se compone de hechos y reglas creados a partir de los requisitos, componentes y las relaciones entre éstos.

La generación de la matriz de trazabilidad se realiza de la siguiente manera: SEVETREQ forma una regla por cada requisito y un hecho por cada componente que contiene la lista de requisitos asociados.

A continuación, se explicará la estructura de los hechos y reglas:

Hechos. Primero se creó una plantilla que define su estructura: el nombre de la plantilla (Implementacion), un campo simple que almacena el nombre del componente (NombreComponente) y un campo múltiple donde se almacenará la lista de requisitos asociados al componente (Requisitos). La plantilla queda de la siguiente manera:

```
(deftemplate Implementacion
        (field NombreComponente)
        (multifield Requisitos))
```

Un ejemplo de algún hecho sería:

```
(Implementacion
        (nombreComponente "Compo7")
        (Requisitos "Requisito4" "Req1_SolCambioReq1" "reqDeSol" "ReqDeSolConTabla1"
        "reqNuevaSol1" "reqDeSol2" "Req1_Proy2" "reqDeSol3" "Req1_Proy3"
        "ReqSolNueva1" "Req1_Proy1ModifGaby" "Req2" "ReqSolNueva2"))
```

Entonces, el hecho anterior diría algo como:

el componente que se llama "Compo7", implementa los requisitos: "Requisito4" "Req1_SolCambioReq1" "reqDeSol" "ReqDeSolConTabla1" "reqNuevaSol1" "reqDeSol2" "Req1_Proy2" "reqDeSol3" "Req1_Proy3" "ReqSolNueva1" "Req1_Proy1ModifGaby" "Req2" "ReqSolNueva2"))

Reglas. SEVETREQ forma una regla por cada requisito almacenado en la base de datos y busca si se encuentra en alguno de los hechos almacenados en la base de conocimiento. Si lo encuentra, le indica a la vista que ese requisito está implementado por cierto componente. Una regla se vería de la siguiente manera:

```
(defrule BuscarReqSolNueva2
        (Implementacion
                (nombreComponente ?N)
                (Requisitos $? "ReqSolNueva2" $?))
                =>
                (printout t ?N": ReqSolNueva2: true;"))
```

Por convención, se estableció que el nombre de la regla se compondría de la palabra "Buscar" y concatenado el nombre del requisito; por la tanto, para el ejemplo anterior, el nombre de la regla es BuscarReqSolNueva2.

La traducción al español de la regla "BuscarReqSolNueva2" sería de la siguiente manera:

"Busca cualquier componente que contenga al requisito que se llama "ReqSolNueva2", si lo encuentras, imprime ReqSolNueva2: true;"

Los componentes diseñados para implementar lo anterior se encuentran en el paquete de la figura 3.2.

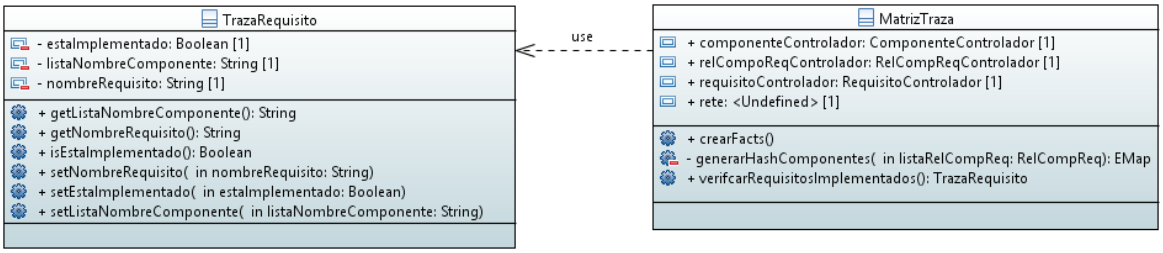


Figura 3.2. Diagrama de clases del paquete “sistemaExperto”.

El método “verificarRequisitosImplementados” llama al método “crearFacts” el cual realiza las operaciones necesarias para crear los hechos con la ayuda de un hashMap por medio del método “generarHashComponentes” y después depositarlos en la base de conocimiento, por último, el método “verificarRequisitosImplementados” crea las reglas de inferencia que consultan los hechos y, posteriormente, mandar a la vista una lista de tipo “TrazaRequisito” que le servirá para desplegar la matriz de trazabilidad.

CAPÍTULO 4. APLICACIÓN DE SEVETREQ EN UN SISTEMA REAL

El sistema se probó usando la documentación de un proyecto de software real llamado "AdminSystem" realizado en el curso Ingeniería de software Orientada a Objetos del semestre 2017-1 del posgrado en Ciencia e Ingeniería de la Computación de la UNAM, dadas las dimensiones de éste y el corto tiempo para terminarlo, el equipo de desarrollo acordó con el cliente realizar una parte, es decir, no entregarían el proyecto completo y el resto lo terminaría alguien externo.

"AdminSystem" es un sistema de alta y control de clientes de la empresa Sistemas MIG. Se registran todos los clientes y la información de los sistemas adquiridos, módulos habilitados, usuarios, características y funcionalidades específicas contratadas según el tipo de póliza que tienen. Adicionalmente controla las versiones de actualizaciones de los sistemas de los clientes, así como las llamadas de soporte para llevar el registro de tiempos utilizados para solucionar los problemas que se presenten al utilizar el sistema.

El desarrollo del sistema se llevó a cabo siguiendo el proceso IS de la norma ISO/IEC29110 Perfil básico. A continuación, se describe la manera en la que se ejecutó cada actividad de SEVETREQ para probarlo usando la documentación generada para AdminSystem.

4.1 INICIO DE APLICACIÓN DE SOFTWARE DE "ADMINSYSTEM"

La ejecución de SEVETREQ en el sistema "AdminSystem" que sirvió de prueba se detalla en los siguientes apartados.

Se ingresa a SEVETREQ por medio del inicio de sesión través de la interfaz gráfica que se muestra en la figura 4.1.

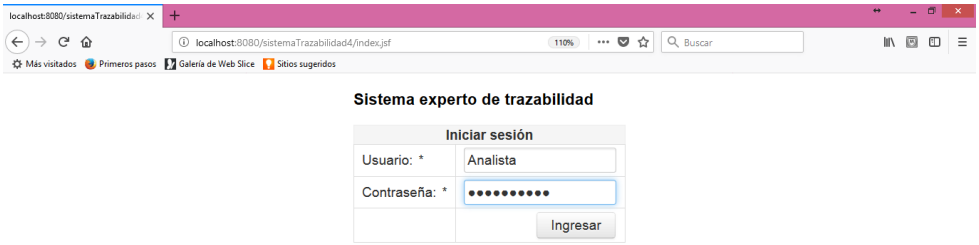


Figura 4.1. Interfaz para inicio de sesión en SEVETREQ.

Después de iniciar sesión, SEVETREQ muestra una interfaz gráfica con dos opciones: elegir un proyecto creado anteriormente o crear uno nuevo. La interfaz aparece en la figura 4.2.

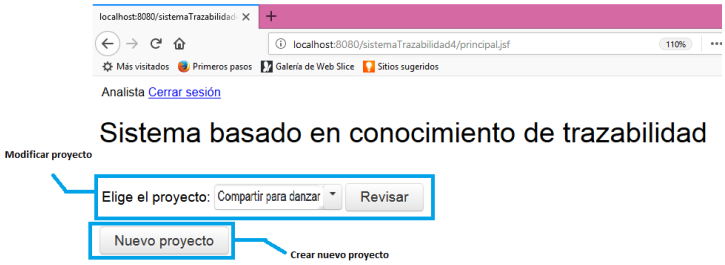


Figura 4.2. Interfaz gráfica para crear o modificar proyectos.

Para crear un nuevo proyecto para el caso de prueba AdminSystem, se presiona el botón *Nuevo* proyecto y aparece el formulario de *Agregar proyecto*; AdminSystem se crea inicialmente, sin agregar ninguna información sobre sus documentos de requisitos, diseño y casos de prueba. La información de los documentos se fue agregando en cada actividad, es decir, cuando se comenzó con la actividad de análisis de requisitos, se agregó la información de su documento principal, lo mismo sucedió con el documento de diseño principal. Dicho formulario se muestra en la figura 4.3. El campo "¿Tiene manual de usuario?" se dejó vacío ya que la documentación no contemplaba uno.

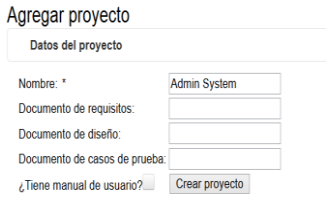


Figura 4.3. Formulario para crear un proyecto

Figura 4.4. Interfaz gráfica de los datos del proyecto.

Al presionar el botón *Crear proyecto,* aparece la interfaz gráfica de la figura 4.4, esta interfaz contiene los datos del proyecto. En la parte de arriba tiene dos botones, uno para crear un proyecto nuevo que no tiene que ver con el proyecto en cuestión (botón *Nuevo proyecto*, al presionarlo se muestra un formulario como el de la figura 4.3) y otro para construir la matriz de trazabilidad (botón "Matriz de trazabilidad") que se explicará más adelante.

Debajo de esos botones se encuentran se encuentran los datos principales del proyecto que son el nombre y las rutas de los documentos de requisitos, diseño y casos de prueba; para agregarlos o modificarlos, basta con ingresar la ruta del documento en el repositorio del equipo y presionar el botón "Modificar proyecto".

4.2 ACTIVIDAD DE ANÁLISIS DE REQUISITOS DE SOFTWARE

Para esta actividad el equipo creó un documento de especificación de requisitos de AdminSystem que contiene en los requisitos funcionales el diagrama general de casos de uso con su detalle (figura 4.5); como requisitos no funcionales se consideraron: la mantenibilidad, seguridad, usabilidad y estilo arquitectónico.

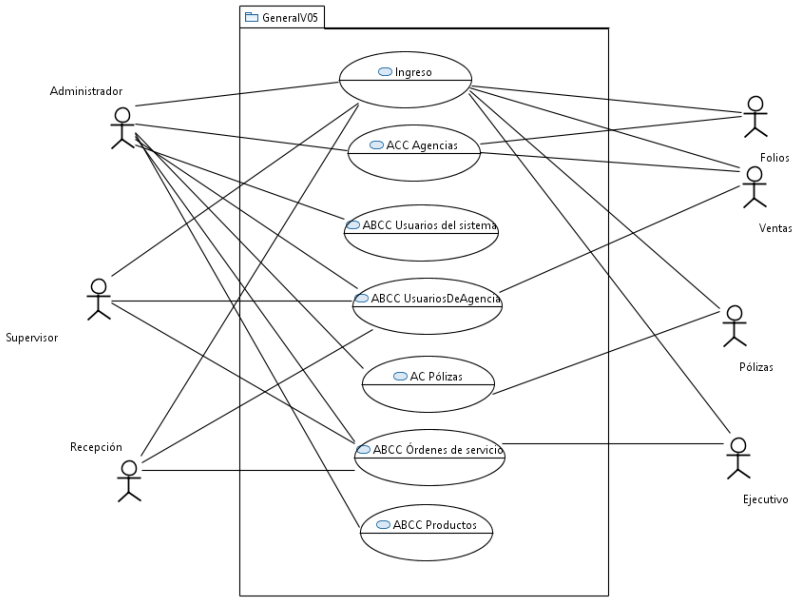
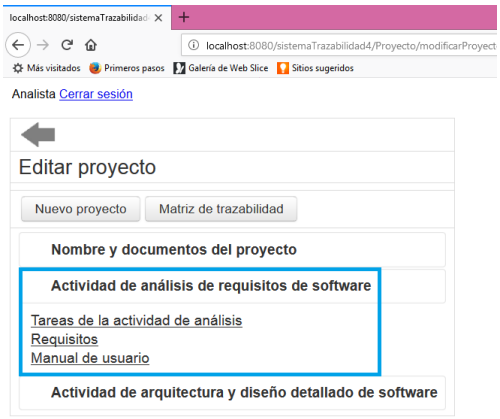


Figura 4.5 Diagrama general de casos de uso del sistema "AdminSystem".[4]

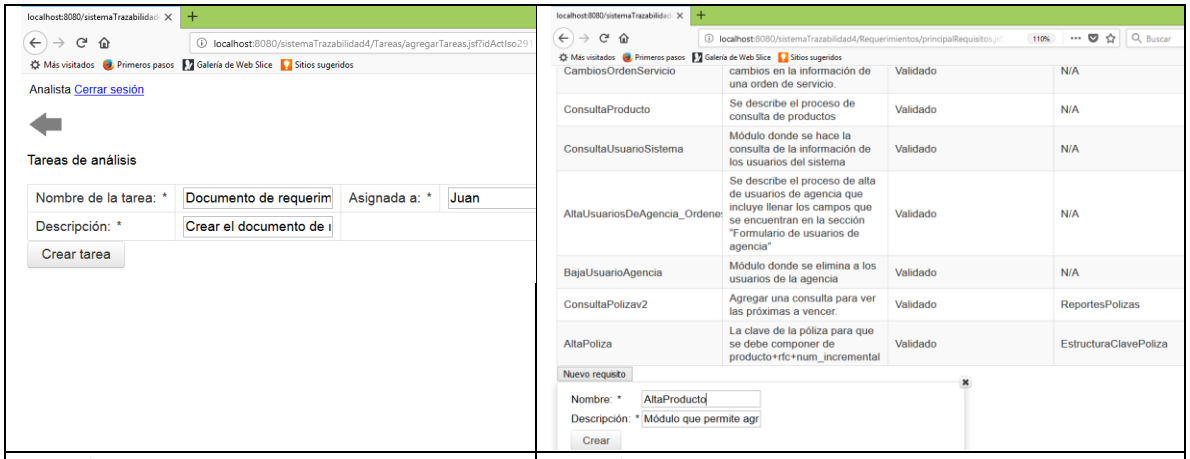
De este documento se obtuvo la lista de requisitos, y la lista de responsabilidades que conciernen a la actividad de análisis de requisitos, fueron ingresadas en la sección "Actividad de análisis de requisitos de software" como se muestra en la figura 4.6.



[4] El diagrama solo es una representación general de las funciones del sistema, la totalidad de los requisitos fue representada en diagramas detallados de cada caso de uso.

Figura 4.6. Lista de apartados que corresponden a las tareas de la actividad de análisis de requisitos de software.

La lista de responsabilidades se capturó en el formulario en la opción “Tareas de la actividad de análisis”; la lista de requisitos se añadió por medio del formulario en la opción “Requisitos”; en la opción “Manual de usuario” no se agregó nada ya que no se generó. En la figura 4.7 se apreciarán los formularios mencionados.



Formulario para agregar tareas | *Formulario para agregar requisitos.*

Figura 4.7. Formularios para agregar tareas y requisitos.

El formulario para agregar tareas o responsabilidades contiene el nombre de la tarea, una descripción y la persona encargada de ejecutarla.

El formulario para agregar requisitos solicita un nombre de requisito y una descripción, después de ingresar esos datos y presionar el botón *Crear*.

El requisito es agregado a una tabla que contiene nombre, descripción, el estado de validación, la solicitud de cambios asociada al requisito (en caso de que exista una) y las acciones a realizar en él, en este caso modificación. Dicha tabla permite filtrar los requisitos por nombre o descripción. Se muestra en la figura 4.8.

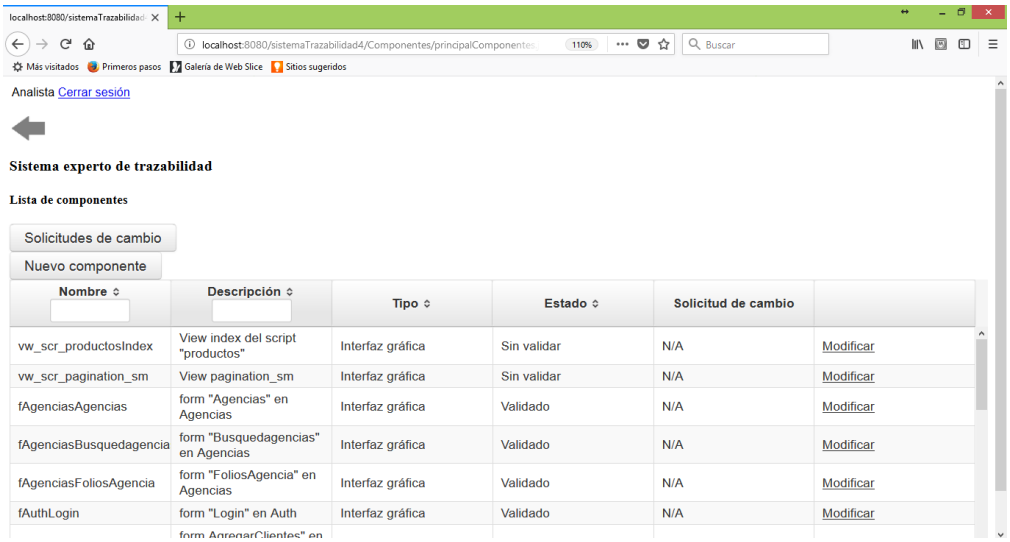


Figura 4.8. Tabla de requisitos del sistema “AdminSystem”.

El formulario para la modificación es similar al que se encuentra en la parte derecha de la figura 4.7.

4.3 ACTIVIDAD DE ARQUITECTURA Y DISEÑO DETALLADO DE SOFTWARE

En esta actividad el equipo realizó diagramas de componentes del AdminSystem que agregó en un documento de diseño en el que, además, se establecieron los atributos de calidad, el estilo arquitectónico, ambiente de implementación, diagrama de distribución y de bases de datos. En la figura 4.9 se encuentra uno de los diagramas de componentes mencionados.

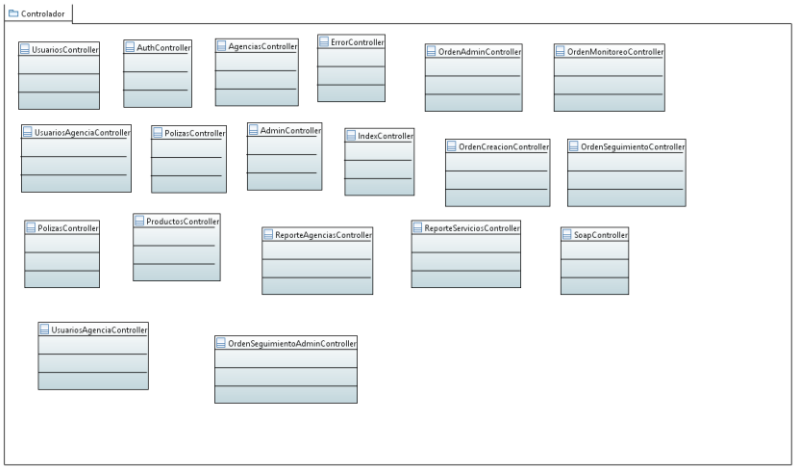


Figura 4.9. Diagrama de componentes del paquete del controlador de AdminSystem.

Para la actividad de arquitectura y diseño detallado de software, las responsabilidades del proyecto AdminSystem para capturarlas en el apartado “Tareas de la actividad de diseño” de la sección “Actividad de arquitectura y diseño detallado de software”; la lista de

componentes obtenidas de los diagramas, se capturaron en el apartado “Componentes” de la misma sección. Estos apartados se ven en la figura 4.10.

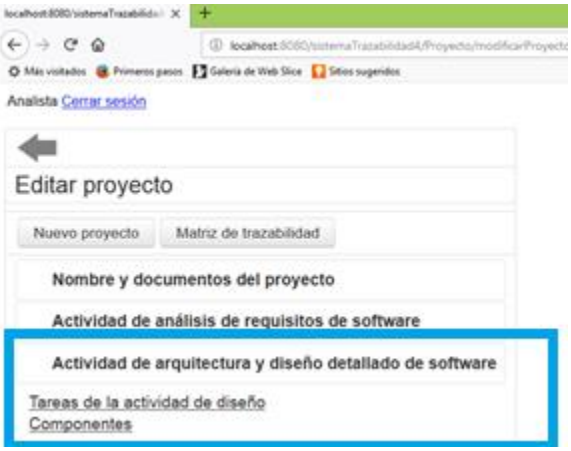


Figura 4.10. Lista de apartados que corresponden a las tareas de la actividad de arquitectura y diseño detallado de software.

La lista de responsabilidades se añadió en el formulario de la opción “Tareas de la actividad de diseño”. Figura 4.11

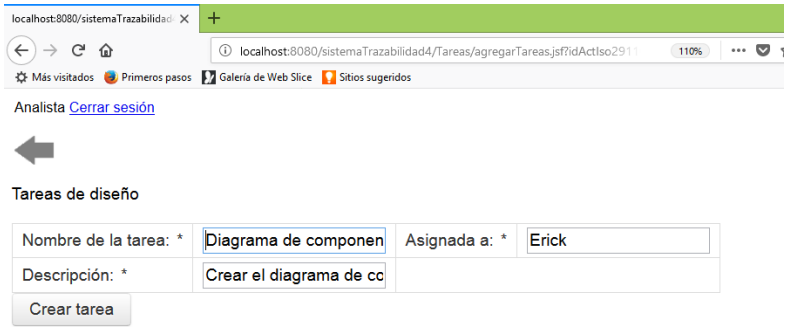


Figura 4.11. Formulario para agregar responsabilidades de diseño.

La lista de componentes se añadió en el formulario en la opción *Componentes*. En esta última opción, también se indicaron los requisitos que cada componente implementó. En la figura 4.12 se apreciará el formulario de los componentes.

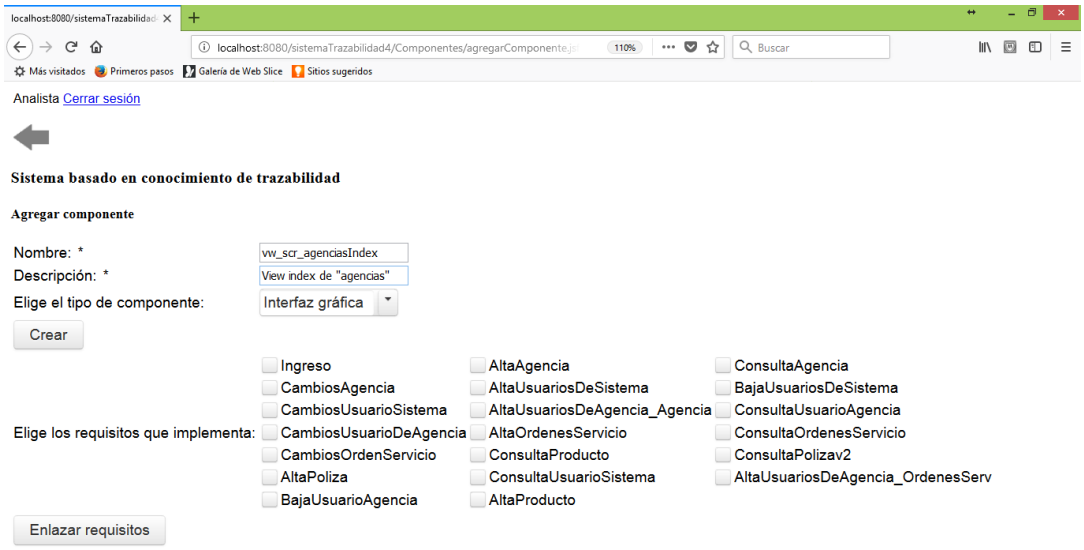


Figura 4.12. Formulario para agregar un componente.

En el campo "Nombre" se recomienda poner el nombre del archivo para poder identificarlo rápidamente en la matriz. En el campo "Descripción" se detalla lo que contiene el archivo y en el campo "Elige el tipo de componente" se indica si el archivo se trata de una interfaz o una clase. Todos estos campos son obligatorios y una vez ingresados, se presiona el botón "Crear", posterior a eso, en la sección "Elige los requisitos que implementa", se hace la selección de todos los requisitos que dicho componente implementará y se presiona el botón "Enlazar requisitos".

La lista de los componentes agregados se despliega en una tabla que contiene nombre, descripción, tipo de componente, el estado de validación, la solicitud de cambios asociada al componente (en caso de que exista una) y las acciones a realizar en él, en este caso modificación. Dicha tabla permite filtrar los componentes por nombre o descripción. Se muestra en la figura 4.13.

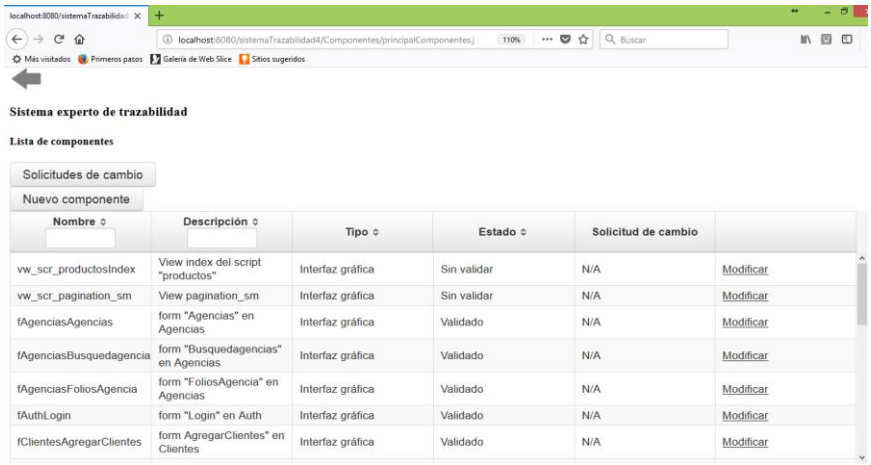


Figura 4.13. Tabla de componentes del sistema "AdminSystem".

El formulario para la modificación es similar al de la figura 4.12.

4.4 REGISTRO DE TRAZABILIDAD

El equipo de AdminSystem realizó el análisis de la trazabilidad por medio de archivos de Excel en los que se indicaba la lista de componentes a desarrollar y la persona encargada de hacerlo, pero no se establecía la lista de requisitos que implementaban.

Para generar la matriz de Trazabilidad se va a la página de los datos del proyecto (figura 4.4) se presiona el botón "Matriz de trazabilidad" para que SEVETREQ genere hechos y reglas del sistema basado en conocimiento, con la información ingresada previamente y despliegue la matriz de trazabilidad de requisitos mostrada en la figura 4.14.

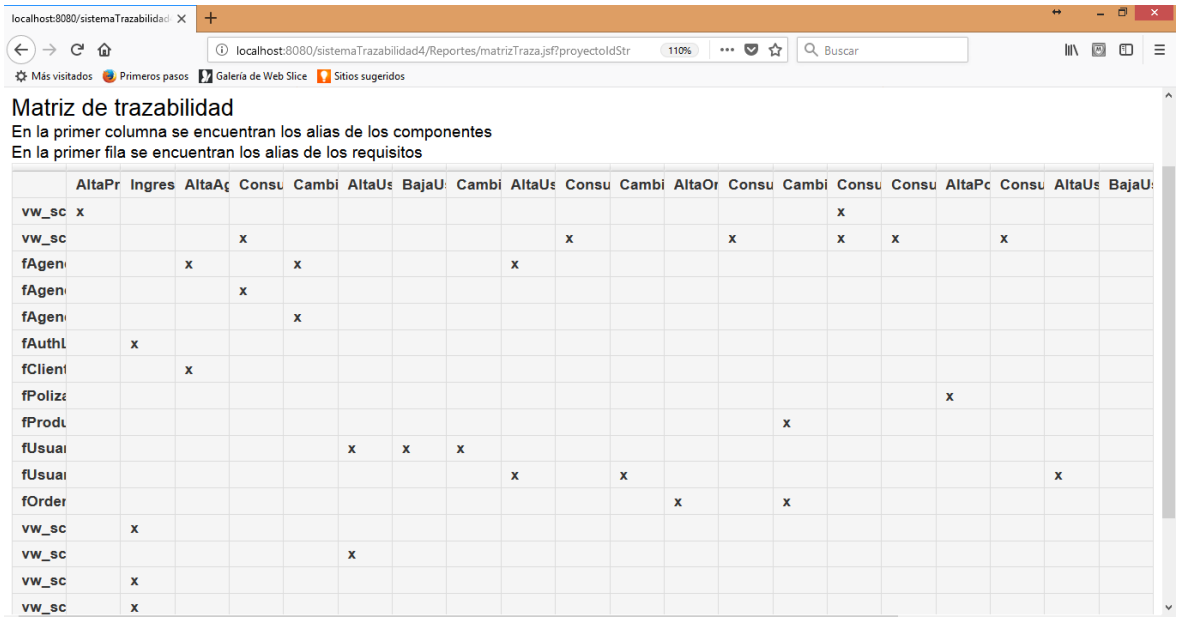


Figura 4.14. Matriz de trazabilidad de requisitos.

La primera fila contiene los nombres de los componentes y la primera columna los nombres de los requisitos, cuando un componente implementa algún requisito, SEVETREQ pone una "x" en la celda donde se cruzan. De esta manera, si una columna no tiene "x" significa que no ha sido implementado; si una fila no tiene "x" significa que el componente no implementa ningún requisito.

Las personas encargadas de terminar el desarrollo de "AdminSystem" conocerán el estado actual de los requisitos y componentes. En este punto la matriz es útil, ya que sabrían en dónde comenzar debido a que el equipo acordó que no entregaría el sistema completo.

Además, SEVETREQ permite realizar modificaciones a las tareas de análisis y diseño, así como a los requisitos y componentes. Por otro lado, también contempla las solicitudes de cambio para requisitos y componentes permitiendo agregar la ruta del repositorio donde se encuentra alojado el documento.

4.5 EN RESUMEN

SEVETREQ sirve de apoyo para generar la documentación de un proyecto de software siguiendo las actividades sugeridas en el proceso IS de la ISO/IEC29110, concluyendo con el registro de trazabilidad que en este caso se realiza por medio de la matriz de trazabilidad de requisitos, la cual se construyó por medio del sistema basado en conocimiento generando hechos y reglas creadas a partir de la información actual del sistema, esto permite representar automáticamente la información proporcionada por los integrantes del equipo sobre los requisitos y componentes de un sistema; y así, tener claridad de la relación que existe entre estos. Además es posible identificar omisiones en el desarrollo, por ejemplo, cuando un requisito no ha sido implementado, retrabajos en el desarrollo como componentes que no implementan ningún requisito o en el mejor de los casos, sirve de ayuda para conocer el impacto en las futuros cambios o mejoras al sistema.

En las secciones anteriores de este capítulo se aplicaron las funcionalidades principales de SEVETREQ, como resultado, se obtuvo una matriz de trazabilidad de requisitos cuya relación entre los requisitos establecidos y los componentes diseñados no mostró omisiones pues todos los requisitos fueron implementados por al menos un componente y todos los componentes implementaban al menos un requisito.

Paralelamente se realizó una matriz de trazabilidad de una manera manual, es decir, en una hoja de cálculo se dispusieron los mismos requisitos y componentes agregados en SEVETREQ y se indicó la relación entre estos. La imagen 4.15 muestra la matriz creada en la hoja cálculo.

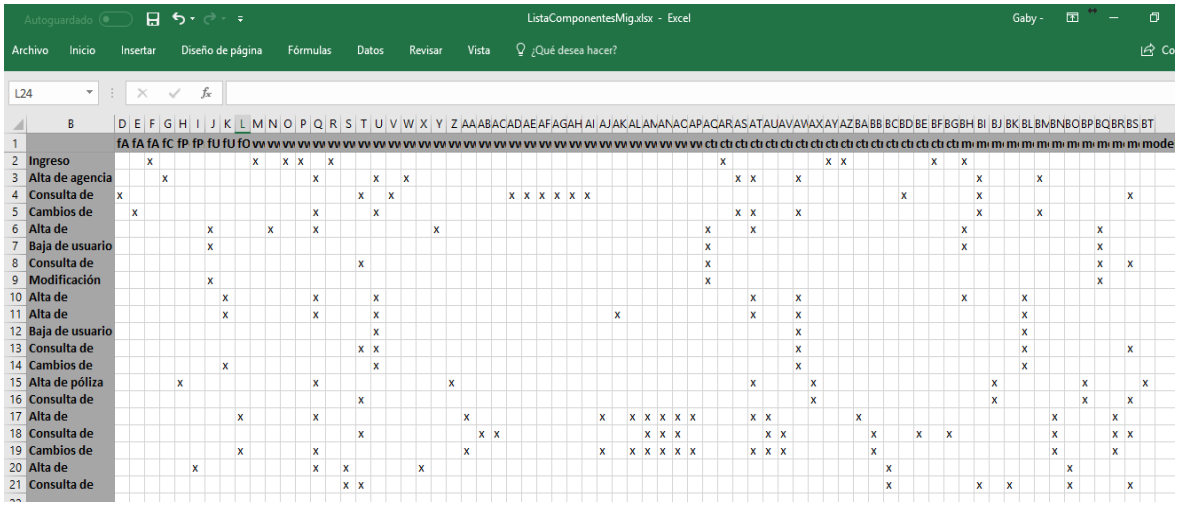


Imagen 4.15. Hoja de cálculo para la matriz de trazabilidad.

En total, se relacionaron 70 componentes y 19 requisitos. Se encontró que, dada la cantidad de información resulta complicado marcar la relación entre los elementos, pues en esa disposición no es fácil identificar los 70 componentes para relacionarlos con los requisitos. Al igual que con SEVETREQ, en esta matriz tampoco se encontraron omisiones en la relación entre los requisitos establecidos y los componentes diseñados pues todos los requisitos

fueron implementados por al menos un componente y todos los componentes implementaban al menos un requisito. Con esto se observa que el equipo de trabajo realizó un buen diseño del sistema.

CONCLUSIONES Y TRABAJO FUTURO

Resulta útil para los equipos de desarrollo conocer si todos los requisitos solicitados se están implementando en el sistema que están desarrollando, esto, además, permite deducir el impacto que tendría desarrollar nuevo código en el mantenimiento o nuevas implementaciones, así como asegurar que cada requisito ha sido implementado.

A través de SEVETREQ un equipo de desarrollo puede administrar los requisitos y componentes por medio de formularios y tablas destinados para ello. Las tablas que ofrece la interfaz facilitan las búsquedas por nombre y ordenamiento por cada una de las columnas lo que permite al equipo ubicarlos rápidamente.

Por medio de la matriz de trazabilidad de requisitos que se genera por medio del sistema basado en conocimiento a través de la interpretación de hechos y reglas generadas con la información actual del sistema, el equipo puede visualizar la trazabilidad en ambas direcciones para que pueda detectar omisiones o analizar el estado actual del sistema que desarrollan y así, facilitar el diseño de nuevas implementaciones.

También pueden dar seguimiento a la documentación del proyecto en el que están trabajando por medio de la interfaz gráfica que contiene los documentos relacionados a él. Ya que en esta versión solo se muestra la ruta del documento, sería útil que el equipo pudiera almacenar en SEVETREQ esos documentos sirviendo como repositorio y después visualizarlos e incluso modificarlos ahí mismo.

Además, ya que SEVETREQ es un sistema web, el equipo de desarrollo puede actualizar y consultar la información de su proyecto desde cualquier lugar con una conexión a internet.

SEVETREQ fue presentado en el Conferencia Internacional de Investigación e Innovación en Ingeniería de Software CONISOFT 2017 como artículo y fue aceptado y publicado en las memorias de ese evento. (Juárez-Ramírez, y otros, 2017).

TRABAJO FUTURO

Aunque en esta fase de SEVETREQ no se aprecian todas las bondades de usar un sistema basado en conocimiento pues solo indica si existe o no una relación entre componentes y requisitos. En una siguiente versión, con la información de la base de conocimiento, podría proporcionar diversas consejos al usuario con base en dicha relación para apoyar la reutilización de código y las reglas no solo verificarían el estado actual del sistema, sino que generarían sugerencias de desarrollo apoyado en un sistema de recomendación para que, dado un conjunto de requisitos, SEVETREQ pueda sugerir una lista de componentes a implementar y sea utilizado también como un complemento de apoyo para el diseño.

Algo que SEVETREQ podría añadir para apoyar en la trazabilidad, es llevar un registro de los componentes utilizados en proyectos anteriores para que, en proyectos futuros, le recuerde

a la persona encargada del diseño los componentes más utilizados y el diseñador decida si los reutiliza o no.

En cuanto a la matriz de trazabilidad, SEVETREQ podría indicar el estado de cada componente, es decir, mostrar si ya fue implementado o no, si se encuentra en desarrollo, en pruebas o tiene errores.

Bibliografía

Agarwal, R., Chetwani, R., & Ravindra et al., M. (2015). Novel methodology for requirements to design traceability of onboard software. *2014 International Conference on Advances in Electronics, Computers and Communications, ICAECC 2014*.

Anjaneyulu, K. (1998). Expert Systems: An Introduction . *Resonance*, 46-58.

ANSI/IEEE Standard. (1984). En *IEEE Guide to Software Requirements Specification* (pág. 830). New York, USA.

Asuncion, H., Francois, F., & Taylor, R. (2007). An end-to-end industrial software traceability tool. *Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering - ESEC-FSE '07*, 115.

Bourque, P., & Fairley, R. (2014). *Guide to the Software Engineering Body of Knowledge, Version 3.0.*

Cleland-Huang, J., Zemont, G., Lukasik, G., & Wiktor. (2004). A heterogeneous solution for improving the return on investment of requirements traceability. *Proceedings of the IEEE International Conference on Requirements Engineering*, 230-239.

CNFBCNA-INDECOPI. (2012). *INGENIERÍA DE SOFTWARE. Perfiles del ciclo de vida para las pequeñas organizaciones (PO). Parte 5-1-2: Guía de gestión e ingeniería: Grupo de perfil genérico. Perfil básico.*

Cong, B., Shin, S. Y., & Salehnia, A. R. (s.f.). Expert systems and expert system languages.

Di Stefano, A., Gangemi, F., & Santoro, C. (2005). ERESYE: artificial intelligence in Erlang programs. 62-71.

Dooley, J. (2011). *Software Development and Professional Practice.* Estados Unidos: Apress.

F. v., V. L., & B. P. (2008). *Handbook of Knowledge Representation.* ELSEVIER B. V.

Fernandes, J., & Machado, R. (2016). Requirements in Engineering Projects. *Lecture Notes in Management and Industrial Engineering,.* Springer International Publishing.

Filho, G., Lencastre, M., & Rodrigues, A. (2013). RETRATOS: Requirement traceability tool support. *CEUR Workshop Proceedings, 1005*, 1-6.

Friedman-Hill, E. (2003). *Jess in action. Rule-Based systems in java.*

Giarratano, J. (2015). CLIPS User's guide. Version 6.30.

Gotel, O., & Finkelstein, A. (1994). An Analysis of the Requirements Traceability Problem. *In Proceedings of the 1st IEEE International Conference on Requirements Engineering*, 94-101.

Gotel, O., Cleland-Huang, J., & Hayes et al., J. (2012). The grand challenge of traceability (v1.0). *Software and Systems Traceability*, 343-409.

Grosan, C., & Abraham, A. (2011). Rule-Based Expert Systems. 149-185.

IBM. (Diciembre de 2008). Requirements definition as the foundation for effective software delivery.

Juárez-Ramírez, R., Jadwiga Oktaba, H., Fernández y Fernández, C., Licea Sandoval, G., Jiménez, S., Aguilar Vera, R., . . . Ucán Pech, R. (2017). 5ta. Conferencia Internacional de Investigación e Innovación en Ingeniería de Software. *5ta Conferencia Internacional en Investigación e Innovación en Ingeniería de Software 2017, CONISOFT 17*, (pág. 114). Mérida Yucatán.

Marcellin Jacques, S. (2014). Conocimiento para compartir Sistemas Basados en el Conocimiento o de Sistemas Expertos.

Marques, A., Ramalho, F., & Andrade, W. (2015). Towards a Requirements Traceability Process Centered on the Traceability Model. *Proceedings of the 30th Annual ACM Symposium on Applied Computing*.

Melorose, J., Perroy, R., & Careas, S. (2005). *Engineering and Managing Software Requirements.*

Oh, J., & Kang, S. (2014). A hierarchical model for traceability between requirements and architecture. *Proceedings of the ACM Symposium on Applied Computing*, 1035-1042.

Pinheiro, F. (2003). Requirements Traceability. En *Perspectives on software requirements* (págs. 91-113).

Ramesh, B., & Dhar, V. (1992). Supporting Systems Development by Capturing Deliberations During Requirements Engineering. *IEEE Transactions On Software Engineering 18*, 498-510.

Ramesh, B., Stubbs, C., Powers, T., & Edwards, M. (1997). Requirements traceability: Theory and practice. *Annals of Software Engineering*, 397-415.

Rodrigues, A., Lencastre, M., & De Cysneiros Filho, G. (2016). Multi-VisioTrace: Traceability visualization tool. *Proceedings - 2016 10th International Conference on the Quality of Information and Communications Technology, QUATIC 2016*, 61-66.

Roetzheim, W. (1991). *Developing Software to Government Standards.* Englewood Cliffs, NJ.: Prentice Hall.

Saltiveri, T. G., Vidal, J. L., & Delgado, C. J. (2005). *Diseño de sistemas interactivos centrados en el usuario.* Barcelona: Editorial UOC.

Sengupta, S., Kanjilal, A., & Bhattacharya, S. (2008). Requirement Traceability in Software Development Process: An Empirical Approach. *2008 The 19th IEEE/IFIP International Symposium on Rapid System Prototyping*, 105-111.

Stehle, G. (1990). *Requirements Traceability for Real Time Systems.* In Proceedings of EuroCASE II, Londres.

Tabares, M. S., Arango, F., & Anaya, R. (2006). Una revisión de modelos y semánticas para la trazabilidad de requisitos. *Revista EIA*, 33-42.

Anexos

A. DOCUMENTO DE PLAN DEL PROYECTO

Introducción

La trazabilidad de requisitos es un tema que se está investigando con el objetivo de que en el futuro el equipo de trabajo la vea como parte de su proceso de desarrollo y pueda llevarla a cabo fácilmente.

Se espera que la trazabilidad facilite tareas en todas las fases del ciclo de vida de la Ingeniería de software y sistemas, proveyendo ganancias tanto en calidad como productividad. En particular, ayudará con la definición de requisitos obteniendo el diseño, código y casos de prueba asociados.

Realizar un sistema basado en conocimiento web apoyado en la norma ISO/IEC 29110 Perfil básico que implemente la trazabilidad de requisitos por medio de una matriz de trazado donde se muestra la relación que existe entre requisitos y componentes del diseño.

Para que ayude al equipo de desarrollo en la administración de la documentación del proyecto de software.

Objetivo

Realizar un sistema basado en conocimiento apoyado en el proceso de Implementación de software (IS) de la norma ISO/IEC 29110 Perfil básico que indique el estado de la documentación para las fases de análisis y diseño, y que, además, muestre la matriz de trazabilidad para la relación entre componentes y requisitos.

Alcance

Desarrollar un sistema basado en conocimiento que guíe a un equipo de desarrollo de software en las actividades de análisis de requisitos del software y de arquitectura y diseño detallado de software del proceso de Implementación de Software (IS) del perfil básico de la norma ISO/IEC 29110. El sistema realizará el registro de la trazabilidad indicado en la norma por medio de una matriz de trazabilidad que tome los requisitos obtenidos de los requisitos de los clientes y los componentes establecidos por el equipo de desarrollo.

La metodología utilizada en el desarrollo del sistema consistirá en realizar las actividades propuestas en dicho proceso comenzando con la actividad de análisis de requisitos en la que se generará un documento de especificación de requisitos, posterior a eso, se realizará el documento de Diseño de software, basándose en los diagramas de componentes del documento anterior, se procederá con la construcción del software y por último, se probará en un sistema desarrollado en el posgrado para un cliente real.

Recursos necesarios

A continuación, se listan los recursos que se necesitarán para el desarrollo del sistema.

Recursos humanos:

Un desarrollador con conocimientos en tecnologías web y bases de datos y lenguajes de sistemas basados en conocimiento.

Equipo:

Un equipo de cómputo para el desarrollo del sistema.

Herramientas:

Tecnologías de lado del servidor: Java eclipse, biblioteca Jess, gestor de base de datos MySql.

Tecnologías del lado del cliente: Primefaces y HTML5.

Capacitaciones:

Capacitación para el uso del lenguaje para sistemas basados en conocimiento CLIPS.

Capacitación para el uso del lenguaje para el uso de la biblioteca jess.

Diagrama general de casos de uso

En la figura 1 se observará el diagrama general de casos de uso diseñado para el sistema.

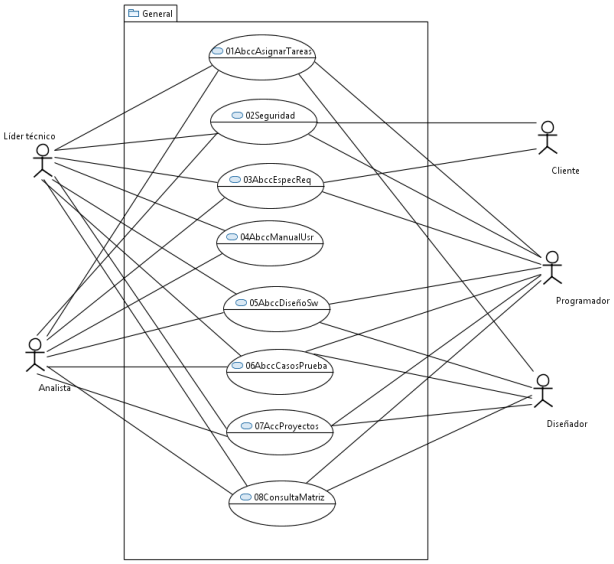


Figura 1. Diagrama general de casos de uso del sistema.

Product backlog

Se apreciará una tabla con la lista de tareas a realizar a lo largo de todo el proyecto, cada tarea está basada en el diagrama general de casos de uso de la sección anterior.

	Funcionalidad o Caso de Uso	**Prioridad**	**Iteración**
AbccAsignarTareas	AltaTareasAnalisis	Alta	5
	BajaTareasAnalisis	Alta	5
	ConsultaTareasAnalisis	Alta	5
	ModificacionTareasAnalisis	Alta	5
	AltaTareasDiseño	Alta	6
	BajaTareasDiseño	Alta	6
	ConsultaTareasDiseño	Alta	6
	ModificacionTareasDiseño	Alta	6
Seguridad	IniciarSesion	Alta	0
	CerrarSesion	Alta	0
	Consulta usuario	Media	4
AbccEspeqReq	AltaEspeqReq	Alta	1
	BajaEspeqReq	Alta	1
	ConsultaEspeqReq	Alta	1
	ModificacionEspeqReq	Alta	1
	ValidacionEspeqReq	Alta	1
AbccManualUsr	AltaManualUsr	Alta	8
	BajaManualUsr	Alta	8
	ConsultaManualUsr	Alta	8
	ModificacionManualUsr	Alta	8
	ValidacionManualUsr	Alta	8
ConsultaMatrizTraza	ConsultaMatrizTraza	Alta	8
AbccDiseñoSw	AltaDiseñoSw	Alta	2
	BajaDiseño	Alta	2
	ConsultaDiseño	Alta	2
	ModificacionDiseño	Alta	2
	ValidacionDiseño	Alta	2
AbccCasosPrueba	AltaCasosPrueba	Media	7
	ConsultaCasosPrueba	Media	7
	ModificacionCasosPrueba	Media	7
	ValidacionCasosPrueba	Media	7
ACCProyectos	AltaProyectos	Alta	0
	CambiosProyectos	Alta	0
	ConsultasProyectos	Alta	0

Riesgos

En esta sección se colocan los posibles riesgos que se detectaron a la hora de realizar el proyecto, así mismo se coloca su impacto y el plan de contingencia realizar en caso de que sucedan, ello con la finalidad de poder mitigar su impacto.

R1	Enfermedad	Se enfermó el desarrollador.	Baja	Alto	Reprogramar las actividades para poder realizar otras actividades que pueda realizar la persona para no afectar los tiempos de entrega	**Identificado**
R2	Subestimación del tamaño de los módulos	Se generó un cronograma que no se puede cumplir debido a la complejidad del módulo a realizar.	Baja	Medio	Generar una buena administración del tiempo para balancear la carga de trabajo de acuerdo con su complejidad.	**Identificado**
R3	**Falla en el Diseño**	**Diseño inadecuado (hay que volver a diseñar)**	**Media**	**Medio**	**Generar propuestas para la adecuación de las fallas tomándolas en cuenta para poder alinearlas con los objetivos del proyecto.**	**Identificado**

B. DOCUMENTO DE ESPECIFICACIÓN DE REQUISITOS DE SEVETREQ

Descripción general

Objetivo del software

Realizar un sistema basado en conocimiento apoyado en el proceso de Implementación de software (IS) de la norma ISO/IEC 29110 Perfil básico que indique el estado de la documentación para las fases de análisis y diseño, y que, además, muestre la matriz de trazabilidad para la relación entre componentes y requisitos.

Glosario de términos

Matriz de trazabilidad. Tabla donde se indica si existe relación entre los componentes y requisitos de un sistema de software.

Norma ISO/IEC 29110 Perfil básico. Denominada Ingeniería de Software – Perfiles de Ciclo de Vida para Empresas Muy Pequeñas, se definen los procesos a implementar a través de perfiles y guías para las características de las VSE (Very Small Entities, pequeñas empresas por sus siglas en inglés). (CNFBCNA-INDECOPI, 2012)

Proceso de Implementación de software (IS). Proceso del perfil básico de la norma ISO/IEC 29110 cuyo objetivo es la ejecución sistemática de las actividades de análisis, diseño, construcción, integración y pruebas a un producto de software nuevo o modificado, de acuerdo con sus requisitos especificados. (CNFBCNA-INDECOPI, 2012)

Sistema basado en conocimiento. Sistemas basados en computadora que integran bases de datos, memorias, mecanismos de razonamiento, agentes, algoritmos, heurísticas, para adquirir

representar, almacenar, generar y difundir conocimientos, inicialmente adquiridos a través de varios expertos humanos dentro de un dominio específico llamado nube. (Marcellin Jacques, 2014)

Tarea. Actividad previamente acordada en la fase de análisis o diseño de un proyecto que realizará algún miembro del equipo de trabajo.

Requisitos Funcionales

Identificación de los casos de uso

A continuación, se muestra una imagen del diagrama general de casos de uso para el sistema.

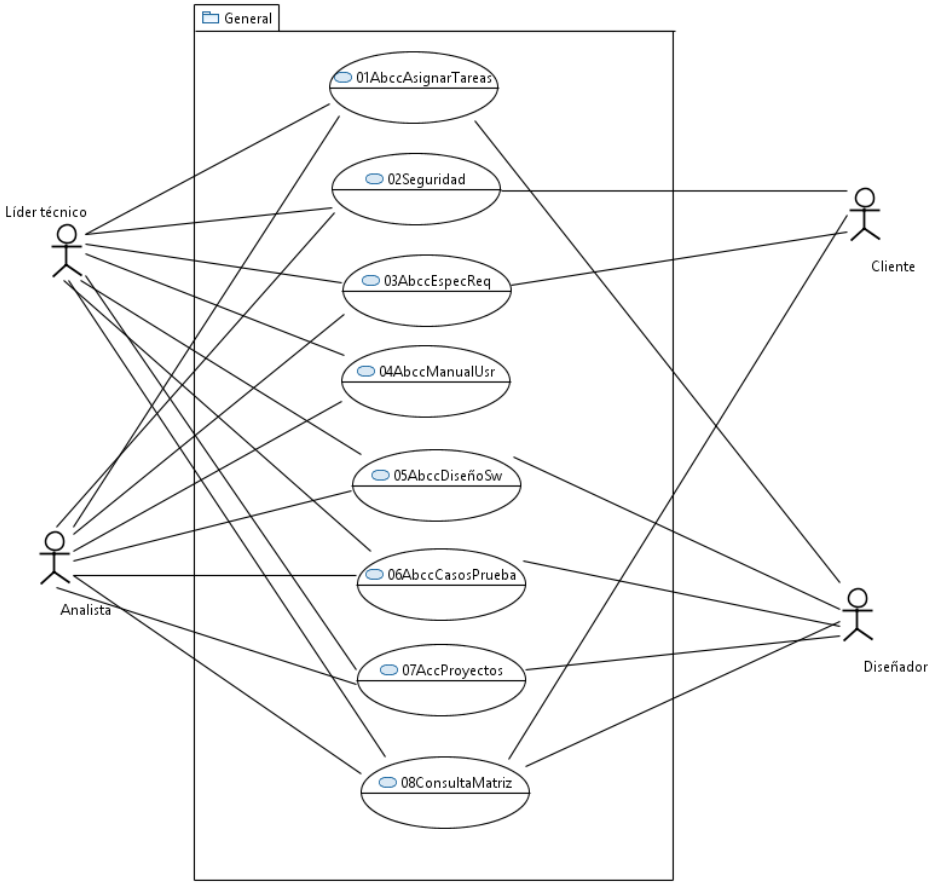


Figura Req01. Diagrama de casos de uso de SEVETREQ

Detallado de los casos de uso

01AbccAsignarTareas

Id:	01AbccAsignarTareas
Nombre:	Abcc de asignar tareas
Actores:	Líder técnico, analista, diseñador
Descripción:	Se realizarán las operaciones de altas, bajas, cambios y consultas de las tareas para las faes de análisis y diseño. Se asignarán tareas a los miembros del equipo de trabajo de acuerdo con cada rol, basado en el plan del proyecto actual. Se realizará llenando una tabla que contenga el nombre del miembro del equipo y la tarea asignada.
Precondiciones:	- Iniciar sesión en el sistema.

011AbccTareasAnalisis

Id:	011AbccTareasAnalisis
Nombre:	Abcc de tareas de análisis
Actores:	Líder técnico
Descripción:	Se realizarán las operaciones de altas, bajas, cambios y consultas de las tareas para la fase deanálisis.
Precondiciones:	- Iniciar sesión en el sistema.

0111AltaTareasAnalisis

Id:	0111AltaTareasAnalisis
Nombre:	Alta de tareas de análisis
Actores:	Líder técnico
Descripción:	Se realizará la creación de una o más tareas de análisis. El usuario deberá ingresar el nombre de la persona asignada y su tarea correspondiente.
Precondiciones:	- Iniciar sesión en el sistema.

Flujo de eventos

Flujo básico

Actor			Sistema		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para agregar.		**2**	Muestra un formulario para ingresar nombre de la persona a asignar y tarea asignada.	
3	Llena el formulario y presiona el botón "Guardar".	**E1, E2**	**4**	Muestra un mensaje de operación exitosa.	**E3**
Postcondición:		Se han creado una o más tareas.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Agregar más tareas	El usuario presiona el botón "Agregar más" para agregar más tareas y se repiten los pasos 3 y 4.
E3	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor, ingresa datos incorrectos o ingresa un nombre de tarea que ya existe para ese proyecto.

0112BajaTareasAnalisis

Id:	0112BajaTareasAnalisis
Nombre:	Baja de tareas de análisis
Actores:	Líder técnico
Descripción:	Se realizará la eliminación lógica de una o más tareas de análisis. La tarea eliminada solo cambia del estado "activo" al estado "inactivo", no se eliminará de la base de datos.
Precondiciones:	- Tener al menos una tarea creada previamente.

Flujo de eventos

Flujo básico

Actor			Sistema		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para eliminar tareas.		**2**	Muestra la lista de tareas ingresadas.	
3	Elige la tarea a eliminar y presiona el botón "Eliminar".		**4**	Muestra un mensaje de confirmación.	
5	Confirma la acción.	**E1**	**6**	Cambia la tarea del estado "activo" al estado "inactivo.	
Postcondición:		La tarea desaparece de la lista.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".

0113ConsultaTareasAnalisis

Id:	0113ConsultaTareasAnalisis
Nombre:	Consulta de tareas de análisis
Actores:	Líder técnico
Descripción:	Se realizará la consulta de una o más tareas de análisis.
Precondiciones:	- Tener al menos una tarea creada previamente.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para consultar una tarea.		**2**	Muestra el formulario de la tarea seleccionada.	
Postcondición:		Se muestra el detalle de la tarea seleccionada.			

0114ModificaciónTareasAnalisis

Id:	0124ModificacionTareasAnalisis
Nombre:	Modificación de tareas de análisis
Actores:	Líder técnico
Descripción:	Se realizará la modificación de una o más tareas de análisis.
Precondiciones:	Tener al menos una tarea creada previamente.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige alguna la opción para modificar una tarea.		**2**	Muestra un formulario para modificar nombre de la persona a asignar y tarea asignada.	
3	Modifica el formulario y presiona el botón “Guardar”.	**E1**	**4**	Muestra un mensaje de operación exitosa.	**E2**
Postcondición:		Se han modificado la tarea seleccionada.			

Excepciones

Id	**Nombre**	**Acción**
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón “Cancelar”.
E2	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor, ingresa datos incorrectos o ya existe un nombre de tarea para ese proyecto.

012AbccTareasDiseño

Id:	011AbccTareasDiseño
Nombre:	Abcc de tareas de diseño
Actores:	Líder técnico, analista y diseñador
Descripción:	Se realizarán las operaciones de altas, bajas, cambios y consultas de las tareas para la fase de diseño.
Precondiciones:	- Iniciar sesión en el sistema. - Haber completado la documentación de la fase de análisis. - Tener al menos un rol creado previamente.

0121AltaTareasDiseño

Id:	0121AltaTareasDiseño
Nombre:	Alta de tareas de diseño
Actores:	Líder técnico, analista y diseñador
Descripción:	Se realizará la creación de una o más tareas de diseño. El usuario deberá ingresar el rol, nombre de la persona asignada y su tarea correspondiente.
Precondiciones:	Iniciar sesión en el sistema. Tener al menos un rol creado previamente.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para agregar.	**E1**	**2**	Muestra un formulario para ingresar nombre de la persona a asignar y tarea asignada.	
3	Llena el formulario y presiona el botón "Guardar".	**E2, E3**	**4**	Muestra un mensaje de operación exitosa.	**E4**
Postcondición:		Se han creado una o más tareas.			

Excepciones

Id	**Nombre**	**Acción**
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Agregar más tareas	El usuario presiona el botón "Agregar más" para agregar más tareas y se repiten los pasos 2 y 3
E3	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor o ingresa datos incorrectos.

0122BajaTareasDiseño

Id:	0122BajaTareasDiseño
Nombre:	Baja de tareas de diseño
Actores:	Líder técnico, analista y diseñador
Descripción:	Se realizará la eliminación lógica de una o más tareas de diseño.

	La tarea eliminada solo cambia del estado "activo" al estado "inactivo", no se eliminará de la base de datos.
Precondiciones:	Tener al menos una tarea creada previamente.

Flujo de eventos

Flujo básico

Actor			Sistema		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para eliminar tareas.		2	Muestra la lista de tareas ingresadas.	
3	Elige la tarea a eliminar y presiona el botón "Eliminar".		4	Muestra un mensaje de confirmación.	
5	Confirma la acción.	E1	6	Cambia la tarea del estado "activo" al estado "inactivo.	
Postcondición:		La tarea desaparece de la lista.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".

0123ConsultaTareasDiseño

Id:	0123ConsultaTareasDiseño
Nombre:	Consulta de tareas de diseño
Actores:	Líder técnico, analista y diseñador
Descripción:	Se realizará la consulta de una o más tareas de diseño.
Precondiciones:	Tener al menos una tarea creada previamente.

Flujo de eventos

Flujo básico

Actor			Sistema		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para consultar una tarea.		**2**	Muestra el formulario de la tarea seleccionada.	
Postcondición:		Se muestra el detalle de la tarea seleccionada.			

0124ModificaciónTareasDiseño

Id:	0124ModificacionTareasDiseño
Nombre:	Modificación de tareas de diseño
Actores:	Líder técnico, analista y diseñador

Descripción:	Se realizará la modificación de una o más tareas de diseño.
Precondiciones:	Tener al menos una tarea creada previamente.

Flujo de eventos

Flujo básico

Actor			Sistema		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para modificar una tarea.		**2**	Muestra un formulario para modificar nombre de la persona a asignar y tarea asignada.	
3	Modifica el formulario y presiona el botón "Guardar".	**E1**	**4**	Muestra un mensaje de operación exitosa.	**E2**
Postcondición:		Se han modificado la tarea seleccionada.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor o ingresa datos incorrectos.

02Seguridad

Id:	02Seguridad
Nombre:	Seguridad
Actores:	Líder técnico, analista, diseñador
Descripción:	Se realizan acciones de inicio y cierre de sesión.
Precondiciones:	Tener un usuario registrado en el sistema.

021IniciarSesion

Id:	021IniciarSesion
Nombre:	Iniciar sesión
Actores:	Líder técnico, analista, diseñador
Descripción:	Se realizará el inicio de sesión a la aplicación.
Precondiciones:	Tener un usuario registrado en el sistema.

Flujo de eventos

Flujo básico

Actor			Sistema		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Muestra un formulario con dos campos: uno para ingresar el nombre		**2**	Ingresa usuario y contraseña, presionar el botón para ingresar.	

	de usuario; otro para ingresar su respectiva contraseña.				
3	Muestra la pantalla principal en la que indica el nombre de usuario y el tipo de usuario				
Postcondición:		El sistema le permite el acceso al usuario.			

Excepciones

Id	Nombre	Acción
E1	Claves incorrectas	El sistema no le permite el acceso al usuario porque el nombre de usuario y/o contraseña son incorrectos.

022CerrarSesion

Id:	021CerrarSesion
Nombre:	Cerrar sesión
Actores:	Líder técnico, analista, diseñador
Descripción:	Se realizará el cierre de sesión a la aplicación.
Precondiciones:	Iniciar sesión en la aplicación

Flujo de eventos

Flujo básico

Actor			Sistema		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Presiona el botón para cerrar sesión.		**2**	Termina la sesión del usuario y muestra la pantalla con el formulario para inicio de sesión.	
Postcondición:		El sistema muestra el formulario para iniciar sesión			

03AbccEspeqReq

Id:	03AbccEspeqReq
Nombre:	Abcc de especificación de requisitos
Actores:	Líder técnico, analista, cliente
Descripción:	Se realizarán las operaciones de altas, bajas, cambios y consultas del documento de requisitos y de cada uno los requisitos del proyecto en el que el equipo se encuentra trabajando.
Precondiciones:	Crear un proyecto.

031AltaEspeqReq

Id:	031Alta EspeqReq
Nombre:	Alta de documento de especificación de requisitos y lista de requisitos
Actores:	Líder técnico, analista
Descripción:	Se realizará la creación de los requisitos y del documento de especificación de requisitos. El sistema solicitará el documento de especificación de requisitos y mostrará una tabla para vaciar los requisitos cuya estructura se encuentra en "Anexo requsitos".
Precondiciones:	Contar con un documento de requisitos.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para agregar requisitos.		**2**	Muestra un formulario donde solicita el documento de requisitos y la lista de requisitos.	
3	Indica la ubicación del archivo , ingresa los requisitos y presiona el botón "Guardar".	**E1**	**4**	Muestra un mensaje de operación exitosa.	**E2**
Postcondición:		Se guarda en base la ubicación del documento, los requisitos, fecha, hora y usuario que realizó la operación.			

Excepciones

Id	**Nombre**	**Acción**
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor o ingresa datos incorrectos.

032BajaEspeqReq

Id:	032Baja EspeqReq
Nombre:	Baja de documento de especificación de requisitos y lista de requisitos
Actores:	Líder técnico, analista
Descripción:	Se realizará la eliminación lógica de uno o más requisitos. El requisito eliminado solo cambia del estado "activo" al estado "inactivo", no se eliminará de la base de datos.
Precondiciones:	Tener al menos un requisito creado previamente.

Flujo de eventos

Flujo básico

Actor			Sistema		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para eliminar requisitos.		**2**	Muestra la lista de requisitos ingresados.	
3	Elige el requisito a eliminar y presiona el botón "Eliminar".		**4**	Muestra un mensaje de confirmación.	
5	Confirma la acción.	**E1, E3**	**6**	Cambia el requisito del estado "activo" al estado "inactivo.	**E2**
Postcondición:		El requisito desaparece de la lista.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Operación inválida	Si el usuario intenta eliminar un requisito asociado a un componente, el sistema muestra un mensaje de error.

033ConsultaEspeqReq

Id:	033ConsultaEspeqReq
Nombre:	Consulta de documento de especificación de requisitos y lista de requisitos
Actores:	Líder técnico, analista, cliente
Descripción:	Se realizará la consulta del documento de especificación de requisitos y lista de requisitos.
Precondiciones:	Haber creado un documento de especificación de requisitos o al menos un requisito en la lista de requisitos.

Flujo de eventos

Flujo básico

Actor			Sistema		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para consultar especificación de requisitos.		**2**	Muestra la ubicación del documento de requisitos y la lista de requisitos.	
Postcondición:		El sistema muestra la ubicación del documento de requisitos y la lista de requisitos.			

034ModificaciónEspeqReq

Id:	034ModificaciónEspeqReq
Nombre:	Modificación de documento de especificación de requisitos y lista de requisitos
Actores:	Líder técnico, analista
Descripción:	Se realizará la modificación del documento de especificación de requisitos y lista de requisitos.
Precondiciones:	Haber creado un documento de especificación de requisitos o al menos un requisito en la lista de requisitos.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para modificar la especificación de requisitos.		**2**	Muestra la ubicación del documento de requisitos y la lista de requisitos.	
3	Modifica la ubicación del documento de requisitos y presiona el botón "Guardar".	**E1**	**4**	Muestra un mensaje de operación exitosa.	
3.1	Elige el requisito a modificar.		**4.1**	Muestra el formulario con el detalle del requisito.	
5.1	Modifica el requisito y presiona el botón "Guardar".	**E1**	**6.1**	Muestra un mensaje de operación exitosa.	**E2**
Postcondición:		Se ha modificado el requisito y/o la ubicación del documento de requisitos, además, los campos de la fecha de modificación y usuario que modificó son actualizados.			

Excepciones

Id	**Nombre**	**Acción**
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor o ingresa datos incorrectos.

0341ValidacionEspeqReq

Id:	0341ValidacionEspeqReq
Nombre:	Validación de la especificación de requisitos.
Actores:	Líder técnico, analista, cliente
Descripción:	Se realizará la validación de la especificación de requisitos. Mientras el documento no esté validado, se podrán realizar modificaciones. Este documento puede estar en uno de los estados del anexo "Estados de documento". Consultar el "Anexo solicitud de cambios" para ver el procedimiento a seguir.
Precondiciones:	Haber creado la especificación de requisitos que incluye el documento y al menos un requisito.

Flujo de eventos

Flujo básico

Actor			Sistema		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para validar la especificación de requisitos.		**2**	Muestra un campo donde se solicita su validación.	
3	Marca la especificación de requisitos como validada y presiona el botón "Guardar".	**E1**	**4**	Muestra un mensaje de operación exitosa.	
3.1	Marca la especificación de requisitos como "Solicitud de cambio", llena la descripción de la solicitud y presiona el botón "Guardar".	**E1, E3**	**4.1**	Muestra un mensaje de operación exitosa.	**E2**
Postcondición:		El estado de la solicitud de cambio es modificado.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor o ingresa datos incorrectos.
E3	Creación inválida	El usuario no puede crear una nueva solicitud de cambio si existe alguna sin validar.

04AbccManualUsr

Id:	04AbccManualUsr
Nombre:	Abcc del manual de usuario
Actores:	Líder técnico, analista
Descripción:	Se realizarán las operaciones de altas, bajas, cambios y consultas del manual de usuario. Este paso es opcional en el flujo del sistema, el usuario puede elegir entre integrar un manual de usuario o no en su documentación.
Precondiciones:	Haber agregado un proyecto.

041AltaManualUsr

Id:	041AltaManualUsr
Nombre:	Alta del manual de usuario
Actores:	Analista
Descripción:	Se solicitará la ubicación del documento del manual de usuario.
Precondiciones:	El usuario deberá indicar al sistema que incluirá un manual de usuario en su documentación.

Flujo de eventos

Flujo básico

Actor			Sistema		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Selecciona la opción para agregar el manual de usuario.		**2**	Muestra la pantalla para la administración de manual de usuario.	
3	Indica la ubicación del archivo en la sección de "Versión preliminar" y presiona el botón "Guardar".	**E1**	**4**	Muestra un mensaje de operación exitosa.	**E2**
Postcondición:		Se genera un manual de usuario asociado al proyecto seleccionado.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor o ingresa datos incorrectos.

042BajaManualUsr

Id:	042BajaManualUsr
Nombre:	Baja de documento de manual de usuario
Actores:	Analista
Descripción:	Se realizará la eliminación lógica del manual de usuario El manual eliminado solo cambia del estado "activo" al estado "inactivo", no se eliminará de la base de datos.
Precondiciones:	Haber indicado que la documentación sí incluirá manual de usuario.

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para eliminar manual de usuario.		**2**	Muestra un mensaje de confirmación. En caso de que haya solicitudes de cambio, indicarle al usuario de su existencia.	
3	Confirma la acción.	**E1**	**4**	Cambia el requisito del estado "activo" al estado "inactivo del manual y las solicitudes de cambio si las hubiera.	
Postcondición:		El manual ya no aparece en el proyecto.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".

043ConsultaManualUsr

Id:	043ConsultaManualUsr
Nombre:	Consulta de documento del manual de usuario
Actores:	Analista
Descripción:	Se realizará la consulta del documento del manual de usuario.
Precondiciones:	Haber agregado la ubicación del documento del manual de usuario.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para consultar manual de usuario.		**2**	Muestra la ubicación del documento del manual de usuario.	
Postcondición:		El sistema muestra la ubicación del documento del manual de usuario.			

044ModificaciónManualUsr

Id:	044ModificaciónManualUsr
Nombre:	Modificación de documento del manual de usuario
Actores:	Analista
Descripción:	Se realizará la modificación de la ubicación documento del manual de usuario. Ver el anexo "Solicitud de cambios para el manual de usuario" para consultar las reglas de negocio de la solicitud de cambios.
Precondiciones:	Haber agregado una ubicación para el documento del manual de usuario. No haber marcado como "Validado" el manual.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para modificar la ubicación del manual de usuario.		**2**	Muestra la ubicación del documento del manual de usuario.	
3	Modifica la ubicación del documento del manual de usuario y presiona el botón "Guardar".	**E1**	**4**	Muestra un mensaje de operación exitosa.	
Postcondición:		Se ha modificado la ubicación del documento del manual de usuario.			

Excepciones

Id	**Nombre**	**Acción**
E1	Documento validado	El usuario intenta modificar la ubicación del documento validado, el sistema le impide realizar la operación.

0441ValidacionManualUsr

Id:	0441ValidacionManualUsr
Nombre:	Validación del documento del manual de usuario
Actores:	Líder técnico, analista
Descripción:	Se realizará la validación del documento del manual de usuario. Mientras el documento no esté validado, se podrán realizar modificaciones. Este documento puede estar en uno de los estados del anexo "Estados de documento". Consultar el "Anexo solicitud de cambios" para ver el procedimiento a seguir.
Precondiciones:	Haber agregado una ubicación para el documento del manual de usuario.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para validar el manual de usuario.		**2**	Muestra un campo donde se solicita su validación.	
3	Marca el manual como validado y presiona el botón "Guardar".	**E1**	**4**	Muestra un mensaje de operación exitosa.	
3.1	Marca el manual como "Solicitud de cambio", llena la descripción de la solicitud y presiona el botón "Guardar".	**E1**	**4.1**	Muestra un mensaje de operación exitosa.	**E2**
Postcondición:		Se almacena el estado del manual de usuario.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor o ingresa datos incorrectos.

05AbccDiseñoSw

Id:	05AbccDiseñoSw
Nombre:	Abcc del diseño de software
Actores:	Diseñador, analista
Descripción:	Se realizarán las operaciones de altas, bajas, cambios y consultas del documento de diseño de software y de cada uno los componentes.
Precondiciones:	Haber agregado un proyecto.

051AltaDiseñoSw

Id:	051AltaDiseñoSw
Nombre:	Alta de documento de diseño de software y lista de componentes.
Actores:	Diseñador, analista
Descripción:	Se realizará la creación de la lista de componentes, de la ubicación del documento de diseño y el emparejamiento de requisitos y componentes. El sistema solicitará el documento de diseño de software y mostrará una tabla para vaciar los componentes que contendrá las columnas indicadas en el "Anexo diseño".
Precondiciones:	Contar con un documento de diseño. Tener los componentes y los requisitos que cubren.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para agregar diseño.		**2**	Muestra un formulario donde solicita el documento de diseño y los componentes.	
3	Indica la ubicación del archivo, ingresa los componentes, establece la relación de cada componente con uno o más requisitos y presiona el botón "Guardar".	**E1**	**4**	Muestra un mensaje de operación exitosa.	**E2**
Postcondición:		Se asocia al proyecto seleccionado el documento, los componentes, fecha, hora y usuario que realizó la operación.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor o ingresa datos incorrectos.

052BajaDiseño

Id:	052BajaDiseño
Nombre:	Baja del documento de diseño y la lista de componentes.
Actores:	Líder técnico, analista
Descripción:	Se realizará la eliminación lógica de uno o más componentes. El componente eliminado solo cambia del estado "activo" al estado "inactivo", no se eliminará de la base de datos.
Precondiciones:	Tener al menos un componente creado previamente.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para eliminar componentes.		**2**	Muestra la lista de componentes ingresados.	
3	Elige el componente a eliminar y presiona el botón "Eliminar".		**4**	Muestra un mensaje de confirmación.	
5	Confirma la acción.	**E1**	**6**	Cambia el componente del estado "activo" al estado "inactivo.	
Postcondición:		El requisito desaparece de la lista.			

Excepciones

Id	**Nombre**	**Acción**
E1	Cancelar acción	El usuario cancela la operación solicitada dando clic a un botón "Cancelar".

053ConsultaDiseño

Id:	053ConsultaDiseño
Nombre:	Consulta del documento de diseño y la lista de componentes
Actores:	Diseñador, analista
Descripción:	Se realizará la consulta del documento de diseño y la lista de componentes.
Precondiciones:	Haber creado un documento de diseño o al menos un componente en la lista de componentes.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para consultar diseño.		**2**	Muestra la ubicación del documento de diseño y la lista de componentes.	
Postcondición:		Muestra la ubicación del documento de diseño y la lista de componentes.			

054ModificaciónDiseño

Id:	054ModificaciónDiseño
Nombre:	Modificación del documento de diseño y la lista de componentes
Actores:	Diseñador, analista
Descripción:	Se realizará la modificación del documento de diseño, la lista de componentes y la relación de estos con los requisitos.
Precondiciones:	Haber creado un documento de diseño y la lista de componentes.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para modificar el diseño.		**2**	Muestra la ubicación del documento de diseño y la lista de componentes.	
3	Modifica la ubicación del documento de diseño y presiona el botón "Guardar".	**E1**	**4**	Muestra un mensaje de operación exitosa.	
3.1	Elige el componente a modificar.		**4.1**	Muestra el formulario con el detalle del componente.	
5.1	Modifica el componente y la relación con uno o más requisitos y presiona el botón "Guardar".	**E1**	**6.1**	Muestra un mensaje de operación exitosa.	**E2**
Postcondición:		Se ha modificado el componente y/o la ubicación del documento de diseño.			

Excepciones

Id	**Nombre**	**Acción**
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor o ingresa datos incorrectos.

0541ValidacionDiseño

Id:	0541ValidacionDiseño
Nombre:	Validación de diseño
Actores:	Diseñador, analista
Descripción:	Se realizará la validación de diseño. Mientras el documento no esté validado, se podrán realizar modificaciones. Este documento puede estar en uno de los estados del anexo "Estados de documento". Consultar el "Anexo solicitud de cambios" para ver el procedimiento a seguir.
Precondiciones:	Haber creado un documento de diseño y la lista de componentes.

Flujo de eventos

Flujo básico

Actor			Sistema		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para validar el diseño.		**2**	Muestra un campo donde se solicita su validación.	
3	Marca el diseño como validado y presiona el botón "Guardar".	**E1**	**4**	Muestra un mensaje de operación exitosa.	
3.1	Marca el diseño como "Solicitud de cambio", llena la descripción de la solicitud y presiona el botón "Guardar".	**E1**	**4.1**	Muestra un mensaje de operación exitosa.	**E2**
Postcondición:		Se almacena el estado del diseño.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor o ingresa datos incorrectos.

06AbccCasosPrueba

Id:	06AbccCasosPrueba
Nombre:	Abcc del documento de casos de prueba
Actores:	Diseñador, analista
Descripción:	Se realizarán las operaciones de altas, bajas, cambios y consultas del documento de casos de prueba.
Precondiciones:	Haber agregado un proyecto.

071AltaCasosPrueba

Id:	061AltaCasosPrueba
Nombre:	Alta del documento de casos de prueba
Actores:	Diseñador, analista
Descripción:	Se solicitará la ubicación del documento de casos de prueba
Precondiciones:	

Flujo de eventos

Flujo básico

Actor			Sistema		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elegir la opción para agregar el del documento de casos de prueba		**2**	Muestra un campo que solicita la ubicación del documento.	
3	Indica la ubicación del archivo y presiona el botón "Guardar".	**E1**	**4**	Muestra un mensaje de operación exitosa.	**E2**
Postcondición:		Se guarda en base la ubicación del documento.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor o ingresa datos incorrectos.

062BajaCasosPrueba

Id:	062BajaCasosPrueba
Nombre:	Baja del documento de casos de prueba
Actores:	Diseñador, analista
Descripción:	Se realizará la eliminación lógica de la ubicación del documento de casos de prueba.
Precondiciones:	Haber ingresado una ubicación del documento.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Desactiva la opción donde se indica que el documento se encuentra activo.		**2**	Muestra un mensaje de confirmación.	
3	Confirma la acción.	**E1**	**4**	Cambia	
Postcondición:		El campo de la ubicación se encuentra vacío.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".

063ConsultaCasosPrueba

Id:	063ConsultaCasosPrueba
Nombre:	Consulta de documento de casos de prueba
Actores:	Diseñador, analista
Descripción:	Se realizará la consulta del documento de casos de prueba.
Precondiciones:	Iniciar sesión en el sistema. Haber agregado la ubicación del documento de casos de prueba

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para consultar el documento de casos de prueba		**2**	Muestra la ubicación del documento de casos de prueba.	
Postcondición:		El sistema muestra la ubicación del documento de casos de prueba.			

064ModificaciónCasosPrueba

Id:	064ModificaciónCasosPrueba
Nombre:	Modificación de documento de casos de prueba.
Actores:	Diseñador, analista
Descripción:	Se realizará la modificación de la ubicación documento de casos de prueba.
Precondiciones:	Iniciar sesión en el sistema. Haber agregado una ubicación para el documento de casos de prueba.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para modificar la ubicación del documento de casos de prueba.		**2**	Muestra la ubicación del documento de casos de prueba.	
3	Modifica la ubicación del documento de casos de prueba y presiona el botón “Guardar”.	**E1**	**4**	Muestra un mensaje de operación exitosa.	
Postcondición:		Se ha modificado la ubicación del documento de casos de prueba.			

Excepciones

Id	**Nombre**	**Acción**
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón “Cancelar”.

0641ValidacionCasosPrueba

Id:	0641ValidacionCasosPrueba
Nombre:	Validación del documento de casos de prueba.
Actores:	Diseñador, analista
Descripción:	Se realizará la validación del documento de casos de prueba. Mientras el documento no esté validado, se podrán realizar modificaciones. Este documento puede estar en uno de los estados del anexo “Estados de documento”. Consultar el “Anexo solicitud de cambios” para ver el procedimiento a seguir.
Precondiciones:	Haber agregado una ubicación para el documento de casos de prueba.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para validar el documento de casos de prueba.		**2**	Muestra un campo donde se solicita su validación.	
3	Marca el documento como validado y presiona el botón “Guardar”.	**E1**	**4**	Muestra un mensaje de operación exitosa.	
Postcondición:		Se cambia el estado del documento de casos de prueba.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor o ingresa datos incorrectos.

07AltaProyectos

Id:	07AltaProyectos
Nombre:	Alta de proyectos
Actores:	Líder técnico, analista
Descripción:	Se realizará la creación de los proyectos de software a documentar. La información que se almacenará será la indicada en el "Anexo campos proyectos".
Precondiciones:	Tener un usuario registrado en el sistema.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para agregar proyecto.		**2**	Muestra un formulario para los datos del proyecto.	
3	Llena el formulario y presiona el botón "Guardar".	**E1**	**4**	Muestra un mensaje de operación exitosa.	**E2**
Postcondición:		Se han creado una o más tareas.			

Excepciones

Id	Nombre	Acción
E1	Cancelar acción	El usuario cancela la operación solicitada dando a un botón "Cancelar".
E2	Información inválida	El sistema muestra un mensaje de error cuando el usuario no ingresa algún valor o ingresa datos incorrectos.

08ConsultaMatriz

Id:	08ConsultaMatriz
Nombre:	Consulta matriz
Actores:	Líder técnico, analista, diseñador, cliente
Descripción:	Se realizará la consulta de la matriz de trazabilidad
Precondiciones:	Establecer la relación entre al menos un componente con al menos un requisito.

Flujo de eventos

Flujo básico

Actor			**Sistema**		
Paso	**Acción**	**Excepción**	**Paso**	**Acción**	**Excepción**
1	Elige la opción para desplegar la matriz de trazabilidad.		**2**	Despliega en pantalla la matriz de trazabilidad.	
Postcondición:		Se despliega en pantalla la matriz de trazabilidad.			

C. DOCUMENTO DE DISEÑO DE SOFTWARE PARA SEVETREQ

1. Representación de la arquitectura

SEVETREQ se creará utilizando el patrón de diseño de modelo-vista-controlador (MVC) debido a que se usará el framework Java Server Faces (jsf) el cual usa dicho patrón.

El desarrollo del sistema basado en conocimiento se hará por medio del lenguaje "CLIPS" utilizando la biblioteca "Jess".

La capa del modelo se encontrará representada por un paquete llamado "modelos" y las clases que ahí se encuentren tendrán el prefijo "modelos". Estas clases son el mapeo de las tablas de la base de datos, por cada tabla, existirá una clase con el mismo nombre de la tabla que mapea.

Además, se ubicarán las clases cuyos métodos contienen las consultas a la base de datos. Los nombres de estas deberán tener como prefijo la palabra "Repo".

Las clases de la capa del controlador se alojarán en el paquete "controladores" y se realizarán por medio de jsf, contendrá la lógica de negocios de SEVETREQ y se corresponden con las vistas de la capa de vista, es decir, cada vista tendrá una clase de controlador. El nombre de las clases de esta capa deberá tener como prefijo la palabra "controlador".

La capa de vista se desarrollará utilizando el framework Primefaces. Se considerarán los siguientes tipos de vistas:

4. Vista para agregar. Servirá para el código de los formularios de alta. Deberá nombrarse de la siguiente manera: agregarElemento, donde "Elemento" se refiere al elemento que agrega.
5. Vista para modificar. Servirá para el código de los formularios de modificación. Deberá nombrarse de la siguiente manera: modificarElemento, donde "Elemento" se refiere al elemento que agrega.
6. Vista para listar. Servirá para mostrar listas de elementos. Deberá nombrarse de la siguiente manera: "principalElemento", donde "Elemento" se refiere al elemento que lista.

1.1 Descripción de la arquitectura

Para la capa del modelo, se utilizará la herramienta Hibernate que sirve para realizar un mapeo del objeto-relacional de la base de datos al código en java. Se eligió por su capacidad para mapear atributos entre una base de datos relacional tradicional y el modelo de objetos. Además de ser software libre.

Las clases de la capa del controlador se alojarán en el paquete "controladores" y se realizarán por medio de jsf.

La capa de vista se desarrollará utilizando el framework Primefaces y html.

2. Vista lógica

2.1 Descripción

Se hará una pequeña modificación al patrón MVC creando paquetes para las clases del Modelo, para las clases del Controlador, para las clases del núcleo del sistema basado en conocimiento y para los archivos de las Vistas. En la figura Dis01 se muestra el diagrama general de paquetes.

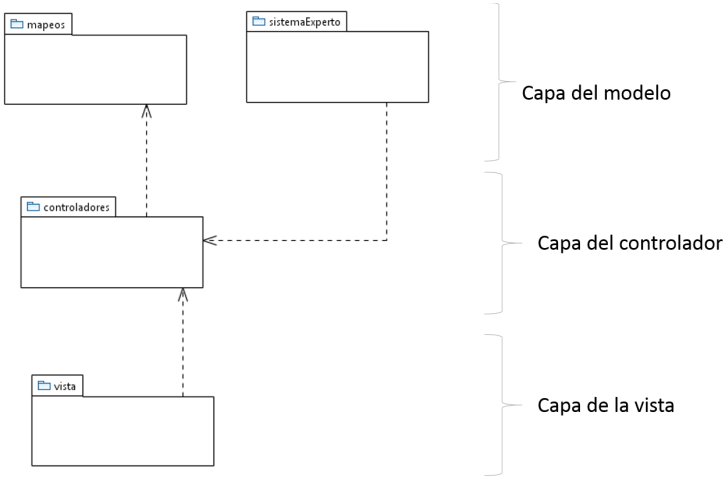


Figura Dis01. Diagrama general de paquetes del sistema SEVETREQ.

2.2 Paquetes de la arquitectura

A continuación, se hará una breve descripción de los paquetes de SEVETREQ y las clases más importantes que contiene.

2.2.1 Paquete mapeos

Contendrá los mapeos de la base de datos realizados por medio de Hibernate, cada clase se corresponderá con una tabla de la base, las propiedades privadas de cada clase serán los campos de su respectiva tabla, se accederá a ellas por medio los métodos *get* y *set*.

También incluirá las clases donde se harán las consultas a la base de datos, el nombre de estas llevará el prefijo "repo", y solo tendrán métodos. La imagen Dis02 contiene las clases de este paquete.

Figura Dis02. Clases del paquete mapeos

2.2.2 Paquete controladores

Se compondrá de las clases que implementan las reglas de negocio de SEVETREQ, y se creará uno para cada vista.

La siguiente figura ilustra las clases que lo compondrán.

Figura Dis03. Clases del paquete "controladores".

Las clases que empiezan con las palabras "Alta" o "Editar" contendrán las acciones para agregar o modificar la entidad a la que se refieren, la operación de eliminación se realizará en las clases de tipo Editar ya que los registros no se borrarán físicamente, solamente cambiarán de estado en el campo llamado "esActivo".

En las clases que no empiezan con "Alta" o "Editar" se llevarán a cabo las acciones de consulta.

2.2.3 Paquete sistemaExperto

En este paquete se almacenarán las clases que implementen el sistema basado en conocimiento para generar la matriz de trazabilidad. Aquí se programarán las reglas de inferencia y hechos por medio de la biblioteca "Jess" para "CLIPS".

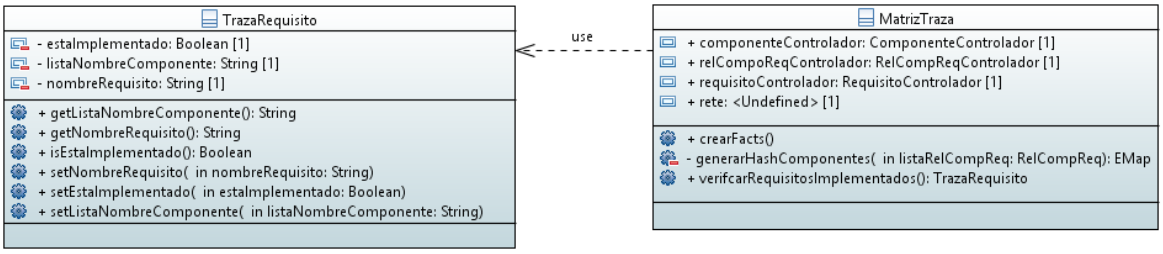


Figura Dis04.Clases del paquete "sistemaExperto".

La clase TrazaRequisito será una clase de tipo entidad que será utilizada por una vista para generar la matriz.

La clase MatrizTraza contendrá los métodos que permitirán generar los hechos y las reglas de inferencia para obtener la matriz.

2.2.4 Paquete vista

Las clases guardadas en este paquete se clasificarán por carpetas que corresponderán a la mayoría de los casos de uso más una, para las actividades de ABCC de solicitud de cambios.

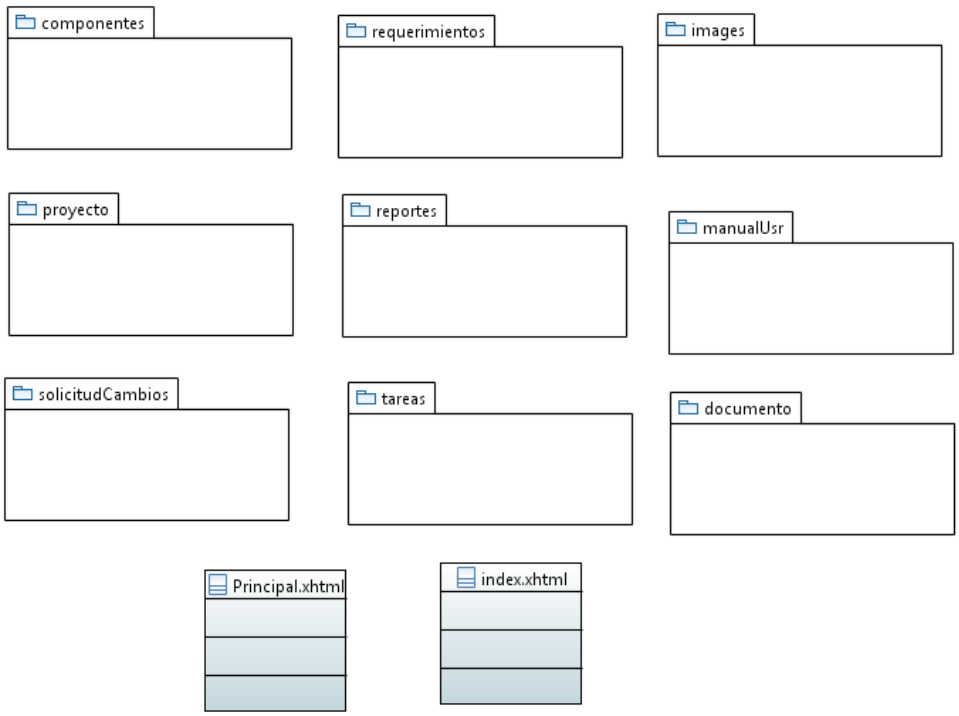


Figura Dis05. Carpetas de la capa vista.

Cada una de las carpetas tendrá vistas para las actividades ABCC de cada entidad, es decir, en la carpeta “componentes” se encontrarán las vistas para agregar, eliminar, modificar y consultar los componentes.

La carpeta “images” almacenará imágenes del sistema.

La carpeta “reportes” solo contendrá la vista que muestra la matriz de trazabilidad.

La clase “Principal.xhtml” mostrará la lista de proyectos dados de alta y permitirá crear nuevos. En la clase “index.xhtml” tendrá el formulario para iniciar sesión.

2.3 Vista de despliegue

El sistema se compondrá de dos nodos interconectados por medio de una red WLAN, usando el protocolo HTTP:

1. Servidor. Este nodo tendrá:
    a. Servidor de páginas: Tomcat.
    b. JSP's.
    c. Contenedor de repositorios.
    d. Base de datos.
2. Integrante del equipo de desarrollo. Puede ser el analista, líder técnico o diseñador.

2.4 Vista de datos

La información generada por SEVETREQ será almacenada en el gestor de base de datos MySQL. Su diagrama se encuentra en la siguiente figura.

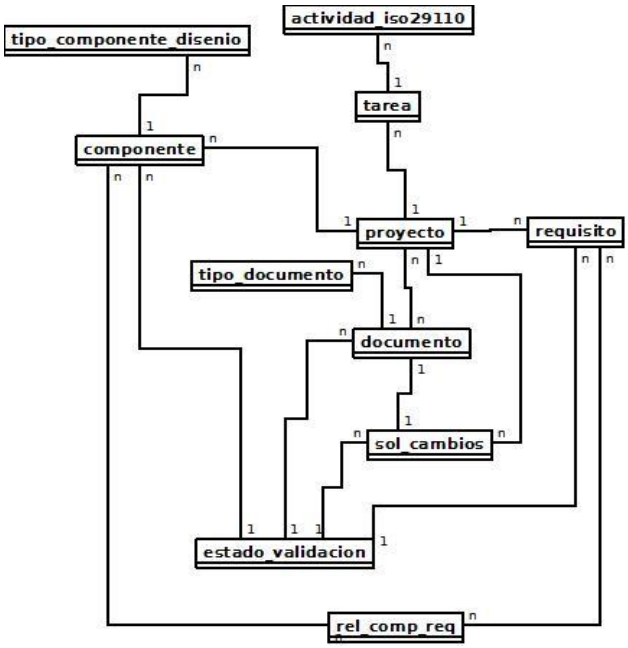


Figura Dis06. Diagrama de la base de datos.

La descripción de cada tabla se lista a continuación:

- actividades_iso29110. Catálogo para almacenar las actividades que la norma ISO 29110 propone. Se comenzará con las actividades de diseño y análisis.
    - id
    - nombre
- componente. Tabla donde se almacenarán los componentes del sistema de software en el que un equipo de desarrollo se encuentre trabajando.
    - id
    - nombreClase
    - descripción
    - tipoDisenio
    - idProyecto
    - esActivo
    - fechaCreacion
    - fechaModificacion
    - usuarioCreacion
    - usuarioModificacion
    - edoValidacion
    - idSolCambios
    - compSucesor
    - esCompNuevo
- documento. Almacenará la ubicación en el repositorio de los documentos que el equipo de desarrollo genere para documentar el sistema en el que se encuentre trabajando.
    - id
    - idTipoDocumento
    - rutaUbicacion

    - esActivo
    - fechaCreacion
    - fechaModificacion
    - usuarioCreacion
    - usuarioModificacion
    - edoValidacion
- estado_validacion. Catálogo de los tipos de validación para un componente, requisito, documento de análisis, documento de diseño y manual de usuario.
    - id
    - nombre
- proyecto. Almacenará la información de los proyectos de software en los que un equipo de desarrollo se encuentre trabajando.
    - id
    - nombre
    - esActivo
    - fechaCreacion
    - fechaModificacion
    - usuarioCreacion
    - usuarioModificacion
    - tieneManualUsr
    - idDocCasosPrueba
    - idDocuRequisitos
    - idDocDisenio
    - idDocManual
- rel_comp_req. Tabla donde se guardarán las relaciones existentes entre componentes y requisitos de un proyecto de software.
    - id
    - idRequisito
    - idComponente
- requisito. Tabla donde se almacenarán los requisitos del proyecto de software en el que un equipo de desarrollo se encuentre trabajando.
    - id
    - nombre
    - descripción
    - idProyecto
    - esActivo
    - fechaCreacion
    - fechaModificacion
    - usuarioCreacion
    - usuarioModificacion
    - edoValidacion
    - idSolCambio
    - reqSucesor
    - esReqNvo

- sol_cambios. Guardará la información de las solicitudes de cambio para los diferentes tipos de documentos que la norma sugiere, a saber, documento de requisitos, diseño y manual de usuario.
    - id
    - idProyecto
    - descripción
    - edoValidacion
    - esActivo
    - fechaCreacion
    - fechaModificacion
    - usuarioCreacion
    - usuarioModificacion
    - alias
    - idSolPadre
- tarea. En esta tabla se depositarán las tareas de análisis y diseño asignadas al equipo de desarrollo.
    - id
    - nombreAsignado
    - nombreTarea
    - idProyecto
    - esActivo
    - fechaCreacion
    - fechaModificacion
    - usuarioCreacion
    - usuarioModificacion
    - idActividadIso29110
    - descripción
- tipo_componente_disenio. Catálogo de los tipos de componente.
    - id
    - nombre
    - descripción
- tipo_documento. Catálogo de los tipos de documento. Tendrá en un inicio los tipos de análisis, diseño y manual de usuario.
    - Id
    - Nombre
- tipo_usuario. Catálogo de los tipos de usuario que se deberán tomar en cuenta para la administración de permisos en SEVETREQ.
    - id
    - nombre
- usuario. En esta tabla estará la información de los usuarios que podrán tener acceso a SEVETREQ.
    - id
    - nombre

D. DOCUMENTO DE CASOS DE PRUEBA

A continuación, se describen los casos de prueba para el sistema SEVETREQ.

Los casos de prueba se componen de un nombre que se corresponde con el del caso de uso que prueba, una lista de pruebas en las que se describen los valores de entrada y los valores esperados.

Caso de prueba 01AbccAsignarTareas

Prueba	Recibe	Esperado	Resultado
Crear una tarea de análisis	Nombre de la tarea, descripción y nombre de la persona a la que se asigna.	Mensaje de confirmación y tarea agregada a la tabla de tareas de análisis.	Mensaje de confirmación, pero no muestra la lista de tareas. Tamaño de la etiqueta "descripción" muy pequeña.
Crear una tarea de análisis sin información	No recibe valores.	Mensaje de error solicitando el ingreso del nombre de la tarea, descripción y nombre de la persona a la que se asigna.	Mensaje de error.
Crear una tarea de diseño	Nombre de la tarea, descripción y nombre de la persona a la que se asigna.	Mensaje de confirmación y tarea agregada a la tabla de tareas de diseño.	Mensaje de confirmación, pero no muestra la lista de tareas.
Crear una tarea de diseño sin información	No recibe valores.	Mensaje de error solicitando el ingreso del nombre de la tarea, descripción y nombre de la persona a la que se asigna.	
Consultar tareas de análisis	Elegir la opción para consultar tareas de análisis.	Muestra la lista de tareas de análisis	Lista de tareas de análisis en pantalla.
Consultar tareas de diseño	Elegir la opción para consultar tareas de diseño.	Muestra la lista de tareas de diseño	Lista de tareas de diseño en pantalla.
Modificar una tarea de análisis	Cambiar el nombre de una tarea de análisis.	Mensaje de confirmación y el nombre actualizado.	Mensaje de confirmación y el nombre actualizado.
Modificar una tarea de diseño	Cambiar el nombre de una tarea de diseño.	Mensaje de confirmación y el nombre actualizado.	Mensaje de confirmación y el nombre actualizado.
Eliminar una tarea de análisis 1	Elegir la opción para borrar una tarea de análisis.	Preguntar al usuario para confirmar acción.	No muestra mensaje de confirmación.
Eliminar una tarea de análisis 2	Aceptar la confirmación para eliminar tarea de análisis.	Eliminar tarea y regresar a la lista de tareas.	No muestra mensaje de confirmación.
Eliminar una tarea de diseño 1	Elegir la opción para borrar una tarea de diseño.	Preguntar al usuario para confirmar acción.	No muestra mensaje de confirmación.
Eliminar una tarea de diseño 2	Aceptar la confirmación para eliminar tarea de diseño.	Eliminar tarea y regresar a la lista de tareas.	No muestra mensaje de confirmación.

Caso de prueba 02Seguridad

Prueba	Recibe	Esperado	Resultado
Iniciar sesión con un usuario registrado y contraseña correcta	Nombre de usuario registrado y contraseña correcta	Acceso a la aplicación	Acceso a la aplicación
Iniciar sesión con un usuario registrado y contraseña incorrecta	Nombre de usuario registrado y contraseña incorrecta	Denegar acceso a la aplicación	Acceso denegado a la aplicación
Iniciar sesión con un usuario no registrado	Nombre de usuario no registrado	Denegar acceso a la aplicación	Acceso denegado a la aplicación

Caso de prueba 03AbccEspeqReq

Prueba	Recibe	Esperado	Resultado
Agregar un requisito	Nombre del requisito y descripción.	Mensaje de éxito, requisito creado y redirección a la lista de requisitos.	Requisito creado y mensaje de éxito, pero no muestra la lista de requisitos.
Agregar un requisito sin nombre	Descripción del requisito.	No crea requisito y muestra mensaje de error solicitando el nombre.	Requisito no creado y muestra mensaje de error solicitando el nombre.
Modificar nombre de un requisito	Nombre modificado del requisito.	Mensaje de éxito y requisito modificado.	Mensaje de éxito y requisito modificado
Borrar la descripción de un requisito	Quitar descripción al requisito.	No modifica el requisito y muestra mensaje de error solicitando una descripción.	No modifica requisito y muestra mensaje de error solicitando agregar una descripción.
Validar un requisito	Cambia el estado de validación a “validado” para algún requisito en estado “no validado”.	Modifica el estado del requisito.	Modifica el estado del requisito.
Ver la lista de requisitos creado	Seleccionar la opción para ver la lista de requisitos.	Se despliega lista de requisitos.	Se despliega lista de requisitos.
Eliminar requisito	Elegir un requisito y eliminarlo.	Muestra mensaje de confirmación antes de eliminar, al aceptar lo elimina; de lo contrario, lo conserva.	Elimina el requisito, pero no muestra mensaje de confirmación.
Agregar solicitud de cambios con solicitudes anteriores validadas	Ruta del documento de solicitud de cambios, alias y descripción.	Crea solicitud de cambios, muestra mensaje de confirmación y despliega lista de solicitudes de cambio.	Solicitud de cambios creada, muestra mensaje de confirmación y despliega lista de solicitudes de cambio.

Agregar solicitud de cambios con solicitudes anteriores no validadas		No muestra la opción para crear solicitudes de cambio.	No muestra la opción para crear solicitudes de cambio.
Agregar un requisito por solicitud de cambios	Nombre del requisito y descripción.	Mensaje de éxito, requisito creado y redirección a la lista de requisitos.	Error al crear el requisito.
Agregar un requisito sin nombre por solicitud de cambios	Descripción del requisito.	No crea requisito y muestra mensaje de error solicitando el nombre.	Requisito no creado y muestra mensaje de error solicitando el nombre.
Modificar nombre de un requisito por solicitud de cambios	Nombre modificado del requisito.	Mensaje de éxito y requisito modificado.	Mensaje de éxito y requisito modificado
Borrar la descripción de un requisito por solicitud de cambios	Quitar descripción al requisito.	No modifica el requisito y muestra mensaje de error solicitando una descripción.	No modifica requisito y muestra mensaje de error solicitando agregar una descripción.
Validar un requisito por solicitud de cambios	Cambia el estado de validación a "validado" para algún requisito en estado "no validado".	Modifica el estado del requisito.	Modifica el estado del requisito.
Ver la lista de requisitos creado por solicitud de cambios	Seleccionar la opción para ver la lista de requisitos.	Se despliega lista de requisitos.	Se despliega lista de requisitos.
Eliminar requisito por solicitud de cambios	Elegir un requisito y eliminarlo.	Muestra mensaje de confirmación antes de eliminar, al aceptar lo elimina; de lo contrario, lo conserva.	Elimina el requisito, pero no muestra mensaje de confirmación.

Caso de prueba 04AbccManualUsr

Prueba	**Recibe**	**Esperado**	**Resultado**
Agregar un manual de usuario al proyecto	Ruta del repositorio donde se aloja el manual de usuario.	Se añade una ruta de manual de usuario al proyecto.	Se añade una ruta de manual de usuario al proyecto.
Modificar un manual de usuario al proyecto	Ruta del repositorio donde se aloja el manual de usuario modificada.	Se modifica la ruta actual.	Ruta modificada.

Se elimina el manual de usuario del proyecto.	Borrar la ruta del manual en el proyecto.	Ruta borrada.	Ruta borrada.
Consultar ruta del manual de usuario del proyecto	Elegir un proyecto para ver la ruta de su manual de usuario.	Muestra ruta del manual de usuario.	Muestra ruta del manual de usuario.

Caso de prueba 05AbccDiseñoSw

Prueba	**Recibe**	**Esperado**	**Resultado**
Agregar un componente	Alias del componente, tipo (componente o interfaz) y descripción.	Mensaje de éxito, componente creado y redirección a la lista de componentes.	Error al mostrar el formulario. Error al asignar el componente al proyecto en cuestión.
Asignar requisitos al componente	Lista de requisitos que el componente implementa.	Requisitos enlazados y mensaje de éxito.	Error al enlazar los requisitos.
Agregar un componente sin nombre	Descripción y tipo del componente.	No crea componente y muestra mensaje de error solicitando el nombre.	Componente no creado y muestra mensaje de error solicitando el nombre.
Modificar nombre de un componente	Nombre modificado del componente.	Mensaje de éxito y componente modificado.	Mensaje de éxito y componente modificado.
Borrar la descripción de un componente	Quitar descripción al componente.	No modifica el componente y muestra mensaje de error solicitando una descripción.	No modifica componente y muestra mensaje de error solicitando agregar una descripción.
Eliminar un requisito de la lista de requisitos enlazados.	Quitar un requisito de la lista.	Requisito eliminado y muestra mensaje de éxito.	Requisito eliminado y muestra mensaje de éxito.
Validar un componente	Cambia el estado de validación a “validado” para algún componente en estado “no validado”.	Modifica el estado del componente.	Modifica el estado del componente.
Ver la lista de componentes creados	Seleccionar la opción para ver la lista de componentes.	Se despliega lista de componentes.	Se despliega lista de componentes.
Eliminar componente	Elegir un componente y eliminarlo.	Muestra mensaje de confirmación antes de eliminar, al aceptar lo elimina; de lo contrario, lo conserva.	Elimina el componente, pero no muestra mensaje de confirmación.
Agregar solicitud de cambios con solicitudes anteriores validadas	Ruta del documento de solicitud de cambios, alias y descripción.	Crea solicitud de cambios, muestra mensaje de confirmación y despliega lista de solicitudes de cambio.	Solicitud de cambios creada, muestra mensaje de confirmación y despliega lista de solicitudes de cambio.

Agregar solicitud de cambios con solicitudes anteriores no validadas		No muestra la opción para crear solicitudes de cambio.	No muestra la opción para crear solicitudes de cambio.
Agregar un componente por solicitud de cambios	Alias del componente, tipo (componente o interfaz) y descripción.	Mensaje de éxito, componente creado y redirección a la lista de componentes.	Error al mostrar el formulario. Error al asignar el componente al proyecto en cuestión.
Asignar requisitos al componente por solicitud de cambios	Lista de requisitos que el componente implementa.	Requisitos enlazados y mensaje de éxito.	Error al enlazar los requisitos.
Agregar un componente sin nombre por solicitud de cambios	Descripción y tipo del componente.	No crea componente y muestra mensaje de error solicitando el nombre.	Componente no creado y muestra mensaje de error solicitando el nombre.
Modificar nombre de un componente por solicitud de cambios	Nombre modificado del componente.	Mensaje de éxito y componente modificado.	Mensaje de éxito y componente modificado.
Borrar la descripción de un componente por solicitud de cambios	Quitar descripción al componente.	No modifica el componente y muestra mensaje de error solicitando una descripción.	No modifica componente y muestra mensaje de error solicitando agregar una descripción.
Eliminar un requisito de la lista de requisitos enlazados. por solicitud de cambios	Quitar un requisito de la lista.	Requisito eliminado y muestra mensaje de éxito.	Requisito eliminado y muestra mensaje de éxito.
Validar un componente por solicitud de cambios	Cambia el estado de validación a “validado” para algún componente en estado “no validado”.	Modifica el estado del componente.	Modifica el estado del componente.
Ver la lista de componentes creados por solicitud de cambios	Seleccionar la opción para ver la lista de componentes.	Se despliega lista de componentes.	Se despliega lista de componentes.
Eliminar componente por solicitud de cambios	Elegir un componente y eliminarlo.	Muestra mensaje de confirmación antes de eliminar, al aceptar lo elimina; de lo contrario, lo conserva.	Elimina el componente, pero no muestra mensaje de confirmación.

Caso de prueba 06AbccCasosPrueba

Prueba	Recibe	Esperado	Resultado
Agregar un documento de casos de prueba al proyecto	Ruta del repositorio donde se aloja el documento de casos de prueba.	Se añade una ruta del documento de casos de prueba al proyecto.	Se añade una ruta del documento de casos de prueba al proyecto.
Modificar un documento de casos de prueba al proyecto	Ruta del repositorio donde se aloja el documento de casos de prueba modificada.	Se modifica la ruta actual.	Ruta modificada.
Se elimina el documento de casos de prueba del proyecto.	Borrar la ruta del documento en el proyecto.	Ruta borrada.	Ruta borrada.
Consultar ruta del documento de casos de prueba del proyecto	Elegir un proyecto para ver la ruta de su documento de casos de prueba.	Muestra ruta del documento de casos de prueba.	Muestra documento de casos de prueba.

Caso de prueba 07AccProyectos

Prueba	Recibe	Esperado	Resultado
Agregar un proyecto	Nombre del proyecto, ruta de los documentos de análisis, diseño e indica si tendrá manual de usuario.	Se crea el proyecto, muestra mensaje de éxito y despliega la lista de proyectos existentes.	Proyecto creado, mensaje de éxito y lista de proyectos desplegada.
Agregar un proyecto sin nombre	Ruta de los documentos de análisis, diseño e indica si tendrá manual de usuario.	Mensaje de error solicitando nombre del proyecto.	Mensaje de error solicitando nombre del proyecto.
Agregar proyecto sin algún documento	Nombre del proyecto.	Se crea el proyecto, muestra mensaje de éxito y despliega la lista de proyectos existentes.	Proyecto creado, mensaje de éxito y lista de proyectos desplegada.
Modificar el nombre de un proyecto	Nombre del proyecto modificado.	Se modifica su nombre, muestra mensaje de éxito y despliega la lista de proyectos existentes.	Nombre modificado, muestra mensaje de éxito y despliega la lista de proyectos existentes.
Modificar la ruta del documento de análisis de un proyecto	Ruta del documento de análisis modificada.	Se modifica la ruta, muestra mensaje de éxito y despliega la lista de proyectos existentes.	Ruta modificada, muestra mensaje de éxito y despliega la lista de proyectos existentes.
Modificar la ruta del documento de diseño de un proyecto	Ruta del documento de diseño modificada.	Se modifica la ruta, muestra mensaje de éxito y despliega la lista de proyectos existentes.	Ruta modificada, muestra mensaje de éxito y despliega la lista de proyectos existentes.

Modificar la ruta del documento de casos de prueba de un proyecto	Ruta del documento de diseño modificada.	Se modifica la ruta, muestra mensaje de éxito y despliega la lista de proyectos existentes.	Ruta modificada, muestra mensaje de éxito y despliega la lista de proyectos existentes.
Consultar lista de proyectos	Elegir la opción para ver la lista de proyectos.	Se despliegan los proyectos creados hasta el momento.	Aparece la lista de proyectos.

Caso de prueba 08ConsultaMatriz

Prueba	**Recibe**	**Esperado**	**Resultado**
Consultar matriz	Elegir opción para ver matriz de trazabilidad.	Desplegar matriz con componentes y requisitos relacionados correctamente. Los componentes se encuentran en la primera columna y los requisitos en la primera fila.	Los componentes se encuentran relacionados.