REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Universidad Católica Santa María

Facultad de Ciencias e Ingenierías Físicas y Formales

Escuela Profesional de Ingeniería de Sistemas

MODELO DE ASOCIACIÓN E IDENTIFICACIÓN DE CONFLICTOS EN REQUISITOS NO FUNCIONALES EMPLEANDO ESCENARIOS Y CASOS DE USO

Tesis presentada por el Bachiller:

García Martínez, Gonzalo Enrique

Para optar el Título Profesional de:

Ingeniero de Sistemas

Especialidad en Ingeniería de Software

Asesor: Dr. Fernández del Carpio Álvaro Rodolfo

AREQUIPA-PERÚ

2018

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

FACULTAD DE CIENCIAS E INGENIERIAS FISICAS Y FORMALES

ESCUELA PROFESIONAL DE INGENIERIA DE SISTEMAS

INFORME DICTAMEN DE BORRADOR DE TESIS

VISTO

El Borrador de Tesis titulado:
Modelo de Asociación e Identificación de Conflictos en Requisitos no funcionales Empleando Escenarios y Casos de Uso

Presentado por (el) (la) (los) Bachiller (es):
García Martínez Gonzalo Enrique

Nuestro dictamen es:
PROCEDENTE

OBSERVACIONES:

Arequipa, 27 de MARZO de 2018

Montoya
1031

1803

(5154) 382036 (5154) 252642 ricamc.ucsm.edu.pe http://ricamc.ucsm.edu.pe

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ACTA TITULO PROFESIONAL

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

En Arequipa, a los ...19.. días del mes de ..ABRIL..... del ..2018.. siendo las ..19:15.. horas, en el local de la Universidad Católica de Santa María, se reunió el Jurado

Presidido por: ...ING. MANUEL MARIANO ZUÑIGA CARNETO.....

Integrantes : ...ING. ANGEL FELIPE MONTESINOS MURILLO.....

...ING. ALVARO RODOLFO FERNANDEZ DEL CARPIO.....

Actuando este último como Secretario con la finalidad de recibir las previas orales del(os) (la) (s) Señor(ita) Bachiller(s)

.....GONZALO ENRIQUE GARCIA MARTINEZ.....

.....

Quien(es) pretende(n) optar el Título Profesional de INGENIERO DE SISTEMAS: ESPECIALIDAD EN INGENIERIA DE SOFTWARE

Sustentando: ...LA TESIS.....

Titulado"...MODELO DE ASOCIACION E IDENTIFICACIÓN DE CONFLICTOS EN.....

.....REQUERIMIENTOS NO FUNCIONALES EMPLEANDO ESCENARIOS.....

.....Y CASOS DE USO.....

.....

....."

El Presidente del Jurado invitó al (los) Titulando(s) a hacer una exposición de su trabajo, conclusiones y recomendaciones, para luego proceder a realizar las preguntas que los Miembros del Jurado consideraron pertinente plantear. Posteriormente, se pasó a deliberar y emitir su voto en la forma establecida por el Reglamento de Grados y Títulos de la Facultad de Ciencias e Ingenierías Físicas y Formales siendo el resultado el siguiente:

.....APROBADO POR UNANIMIDAD.....

Con lo que se dio por terminado el Acto siendo las 20:45 horas, para dar fe firmamos a continuación los Miembros del Jurado y el (los) Titulado(s).

.....
PRESIDENTE

.....
SECRETARIO

.....
INTEGRANTE

.....
TITULANDO

.....
TITULANDO

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

PRESENTACIÓN

Sr. Director del Programa Profesional de Ingeniería de Sistemas.

Sres. Miembros del Jurado.

De conformidad con las disposiciones del Reglamento de Grados y Títulos del Programa Profesional de ingeniería de Sistemas, pongo a vuestra consideración el presente trabajo de investigación titulado: "MODELO DE ASOCIACIÓN E IDENTIFICACIÓN DE CONFLICTOS EN REQUISITOS NO FUNCIONALES EMPLEANDO ESCENARIOS Y CASOS DE USO", el mismo que de ser aprobado me permitirá optar el Título Profesional de Ingeniería de Sistemas.

Gonzalo García Martínez

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

DEDICATORIA

Agradezco a mi padre y a mi madre que me brindaron el apoyo para poder estudiar la carrera que tanto me apasiona, por las enseñanzas y valores que me inculcaron desde pequeño y por enseñarme a no desistir de mis metas y objetivos.

Agradezco a mi hermano y a mi hermana porque siempre me impulsaron a salir adelante, cuidándome y dándome consejos de vida.

Agradezco a mis amigos y amigas con los que compartí buenos momentos durante estos 5 años de universidad.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

EPÍGRAFE

"El fracaso es una gran oportunidad

para empezar otra vez con inteligencia"

Henry Ford

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

RESÚMEN

El presente trabajo de investigación consiste en proponer un modelo de asociación e identificación de conflictos en requisitos no funcionales por medio de escenarios y casos de uso. La idea de lograr estas identificaciones permitirá minimizar errores en la gestión de requisitos funcionales y no funcionales.

El propósito es obtener un conjunto de conflictos en los requisitos no funcionales, conflictos que ayudarán a una mejor gestión de los requisitos en la construcción del producto de software, permitiendo de esta forma mejores descripciones de los requisitos. Gran parte está relacionado con la educción y elicitación de requerimientos, las cuales son las primeras etapas en el proceso de requerimientos, pero presenta una mayor orientación hacia los requisitos no funcionales y su mal entendimiento como por ejemplo malas interpretaciones de requisitos funcionales.

Para ello se utilizaron los escenarios y los casos de uso, artefactos que permiten plasmar, en un lenguaje sencillo, los requisitos y de los cuales se deducen los conflictos y ambigüedades. Asimismo, el modelo de asociación incluye el conjunto de actividades que conducen a comprender cuál será el impacto de los conflictos sobre el producto en construcción.

Palabras clave: requerimientos no funcionales, modelo, conflictos.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

ABSTRACT

The present research work consists of proposing a model of association and identification of conflicts in non-functional requirements through scenarios and use cases. The idea of achieving these identifications will make it possible to minimize errors in the management of functional and non-functional requirements.

The purpose is to obtain a set of conflicts in the non-functional requirements, conflicts that will help to better manage the requirements in the construction of the software product, thus allowing better descriptions of the requirements. Much is related to the eduction and elicitation of requirements, which are the first stages in the requirements process but presents a greater orientation towards the non-functional requirements and conflicts that originate, such as, for example, misinterpretations of functional requirements.

To this end, the scenarios and use cases were used, devices that allow the requirements to be expressed in simple language and from which conflicts and ambiguities can be deduced. The association model also includes the set of activities that lead to an understanding of the impact of conflicts on the product under construction.

Keywords: nonfunctional requirements, model, conflicts.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

INTRODUCCION

Es importante destacar que muchos años atrás la Ingeniería de Requisitos no cubría con las expectativas de hoy en día. Los requisitos eran tomados de manera informal y se desconocía sobre los requisitos funcionales y no funcionales. Actualmente se conoce una división bastante clara sobre aquellos requisitos que ven las funcionalidades del sistema y aquellos que afectan, como factores externos, en la concepción de los requisitos.

La ingeniería de requerimientos, el dominio del problema y la información sostenida parte de la educción de requerimientos y tendrá una buena solución si el ingeniero tiene una comprensión bien fundada del problema, en donde los errores informáticos ya no se originarían más por requisitos funcionales y no funcionales inadecuados o incomprendidos.

Los requisitos no funcionales y la consecuente obtención de medidas, para adquirir conocimiento del dominio de un problema, pueden ser deducidas aplicando diferentes técnicas. La elección del modelo de asociación depende del tiempo y recursos disponibles por el analista y por supuesto, de la clase de información que necesita ser capturada.

El modelo de asociación propuesto constituye una secuencia de pasos para detectar los conflictos surgidos entre requisitos no funcionales. Esto permite verificar y obtener medidas que permitan establecer correctivos en la funcionalidad del sistema.

El presente trabajo de investigación se organiza de la siguiente manera: El capítulo 1 trata sobre el planteamiento teórico relacionado con el problema de investigación. En el capítulo 2 se presenta el marco teórico correspondiente, en el capítulo 3 se presenta el modelo de asociación propuesto para que en el capítulo 4 se aplique la misma mediante un caso de estudio. Finalmente se presentan las conclusiones y recomendaciones del caso.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

CONTENIDO

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

INDICE DE FIGURAS

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

INDICE DE TABLAS

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CAPÍTULO 1: PLANTEAMIENTO TEÓRICO

1.1. Título del proyecto

MODELO DE ASOCIACIÓN E IDENTIFICACIÓN DE CONFLICTOS EN REQUISITOS NO FUNCIONALES EMPLEANDO ESCENARIOS Y CASOS DE USO.

1.2. Descripción del problema

Cuando se realizan productos de desarrollo de software, estos se encuentran basados en un proyecto de desarrollo de software que contienen procesos que deben ser respetados para lograr productos de calidad. Cuando estos procesos son obedecidos, el desarrollo se torna fácil en todo su ciclo de desarrollo y en donde la verificación y control de la calidad hacen tener confianza en el mencionado producto.

Uno de los mayores problemas que se presenta cuando se construye todo sistema es la toma de requerimientos, para validar los mismos existen técnicas poco difundidas. Juntamente con la toma de requerimientos, la generación de interfaces de usuario también es otro problema del cual se escribe mucho. Si los requerimientos presentan una buena concepción entonces se generará una codificación mucho más cercana a la realidad. La forma como se adquieren los requerimientos o la adquisición del conocimiento de la lógica del negocio depende de cómo se logre entender la lógica del negocio y de la habilidad del Ingeniero de requerimientos para entender el problema.

La Ingeniería de Requerimientos, se encarga de obtener una explicación detallada de un problema, con el fin de elaborar un Sistema de Software, que pueda satisfacer las "necesidades y objetivos" de la organización donde se implantara este sistema. En la comunidad de Ingeniería de Requerimientos, estos objetivos constituyen el fundamento del sistema, y son usualmente definidos como las metas a ser cumplidas por el sistema y su entorno, aunque algunos autores distinguen los objetivos del sistema de los objetivos de la organización. Debido a que los documentos de especificación de requerimientos cumplen también una función contractual entre analistas y stakeholders, es importante para los stakeholders que el contenido de la especificación de requerimientos esté en un lenguaje entendible con el cual ellos puedan interactuar activamente.

Los stakeholders son aquellas personas afectadas de un modo u otro por el sistema

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

a construir, es decir, tienen alguna clase de interés en el proyecto. Mediante la concentración sobre Metas/Objetivos más que en requerimientos específicos, los analistas se comunican con los stakeholders usando un lenguaje basado en conceptos (ej. Objetivos) con los que se sienten familiarizados y confortables. A los stakeholders puede resultarle más fácil comprender los Objetivos a cumplir que la funcionalidad que se exhibiría en el sistema deseado. Los requerimientos a menudo son difíciles de entender, pero ellos pueden ser justificados y explicados a través de la discusión de Objetivos. Debe notarse que los Objetivos de la empresa y los Objetivos del sistema son más estables (a través del tiempo de vida de la empresa) que los requerimientos definidos en un momento dado.

Es por eso que con el presente trabajo de investigación se pretende proponer un modelo de asociación en la identificación de conflictos en los requerimientos no funcionales en base a escenarios y casos de uso que valide los requerimientos no funcionales de software tomando como punto de partida la lógica de negocio, y que luego permita ser aplicado a la etapa de la ingeniería de requerimientos.

1.3. Delimitaciones y definición del problema

1.3.1. Delimitaciones

a. Delimitación espacial

El presente trabajo de investigación se lleva a cabo en la ciudad de Arequipa.

b. Delimitación temporal

El trabajo se inicia en noviembre del 2016 y culmina en noviembre del 2017.

c. Delimitación social

Está orientado a resolver problemas sobre identificación de requisitos no funcionales en aplicaciones informáticas durante el proceso del diseño de software, por lo que su orientación social implica resolver una problemática a las personas dedicadas al desarrollo de productos de software.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

d. Delimitación conceptual

Requisitos, requisitos funcionales, requisitos no funcionales, proceso de desarrollo de software, modelo de asociación.

1.3.2. Definición del problema

El problema por tratar es la poca importancia que se le brinda a los requisitos no funcionales con el propósito de obtener productos de software con un alto grado de calidad; esto sumado a la poca experiencia que se tiene en el tratamiento de estos temas. Ante esta situación se considera la construcción de un modelo de asociación, que contemple los objetivos desde el diseño de estos, y otros aspectos como necesidades de los usuarios finales.

La proliferación de técnicas y modelos para la elaboración del catálogo de requisitos están dando resultados adecuados que permiten la construcción de productos de software de manera adecuada. Estos elementos hacen parte del diseño centrado en el usuario, un instrumento para lograr un mejor entendimiento e incrementar la satisfacción del cliente. A pesar de esto, la importancia de este tipo de requisitos dentro del proceso de desarrollo de software solo se considera como un atributo final del producto y no como un elemento inherente e incluyente durante el transcurso del desarrollo (Alarcón, 2008).

La construcción del catálogo de requisitos es un elemento intangible del software, por ello, es difícil de visualizar, medir y reconocer como un factor determinante de su calidad. La Ingeniería de Requisitos promueve la evaluación temprana del software en el proceso de desarrollo de software y la participación del usuario en todas las fases del ciclo de vida, las empresas no desconocen la importancia de este documento, pero las prácticas promovidas por la conceptualización de requisitos no se encuentran incorporadas.

Obtenidos los requisitos no funcionales, se espera obtener efectos que promuevan un buen catálogo de requisitos. Los inconvenientes que se pueden encontrar es que los escenarios no se hallen correctamente definidos no permitiendo una buena especificación de requisitos. Ante este inconveniente, es necesario elaborar una solución en base a un modelo de asociación que permita identificar conflictos entre los requisitos no funcionales.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

1.4. Formulación del problema

1.4.1. Problema principal

Proponer un modelo de asociación, basado en escenarios y casos de uso, que permita identificar conflictos en los requisitos no funcionales.

1.5. Objetivos de la investigación

1.5.1. Objetivo general

Proponer un modelo de asociación e identificación de conflictos en los requerimientos no funcionales mediante el uso de escenarios y casos de uso.

1.5.2. Objetivos específicos

- Conocer las características de los requerimientos no funcionales y su efecto en la construcción de productos de software.
- Identificar un conjunto de factores que conducen a la obtención de conflictos en los requerimientos no funcionales.
- Definir un conjunto de ítems que permitan bosquejar el modelo de asociación basado en escenarios y casos de uso.

1.6. Viabilidad de la investigación

1.6.1. Económica

Se cuenta con los recursos económicos necesarios para solventar el presente trabajo de investigación en todas sus fases de construcción, así como en el de analizar el caso de estudio correspondiente.

1.6.2. Técnica

Se cuenta con la capacidad académica y los conceptos adecuados para resolver el problema.

1.6.3. Operativa

Medios bibliográficos, Internet, bibliotecas y otras universidades con infraestructura, laboratorios, entre otros.

1.7. Justificación e importancia de la investigación

1.7.1. Justificación

La Ingeniería de Software, se encarga de obtener una descripción detallada de un problema, con el fin de elaborar un Sistema de Software, que

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

pueda satisfacer las “necesidades y objetivos” de la organización donde se implantara este sistema. En la comunidad de Ingeniería de Software, los objetivos vienen a ser el fundamento del sistema, y son generalmente dados a conocer como las metas a ser realizadas por el sistema y su entorno, aunque algunos autores distinguen los objetivos del sistema de los objetivos de la organización.

Dado que los documentos de construcción cumplen también una función contractual entre analistas y stakeholders, es importante para los stakeholders que el contenido de la construcción esté en un lenguaje entendible con el cual ellos puedan interactuar activamente. Los stakeholders son aquellas personas afectadas de un modo u otro por el sistema a construir, es decir, tienen alguna clase de interés en el proyecto. Mediante casos de uso los desarrolladores se comunican con los stakeholders usando un lenguaje basado en conceptos con los que se sienten familiarizados y confortables de tal manera que los requerimientos sean entendibles.

A los desarrolladores les es mucho más fácil entender los objetivos a cumplir que la funcionalidad que tendrá el sistema a desear. Los factores generalmente son difíciles de comprender, pero estos pueden ser justificados y mejor entendidos a través de la discusión de calidad, la cual comienza en la ingeniería de requerimientos. Debe percibirse que la construcción de productos de software, desde el análisis de procesos, es más estable pudiendo obtener costos más reales.

Entender los conflictos de los requerimientos no funcionales permitirá acentuar la calidad en la construcción de los productos de software. Existen muchos medios, pero uno de los más importantes es el proceso de desarrollo de software en función de las necesidades del cliente; en él se encuentra definida una lógica de negocio casi precisa, siempre y cuando este artefacto se encuentre bien diseñado y elaborado. Por otro lado, los casos de uso son un artefacto que presenta una relación muy directa con el cliente, por lo que elaborarlo de una manera adecuada y sólida permitirá obtener las primeras de medidas de calidad.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

1.7.2. Importancia

Permitir a los desarrolladores de software contar con un modelo de asociación que permita encontrar los conflictos en los requisitos no funcionales.

1.8. Limitaciones de la investigación

El trabajo de investigación no empleará métodos formales ni otro procedimiento para la concepción y definición de los requisitos no funcionales, el modelo de asociación fue comprobada en un caso de estudio práctico.

1.9. Área, línea, tipo y nivel de la investigación

1.9.1. Área de investigación

El área de investigación es la de Ingeniería de Software.

1.9.2. Línea de investigación

La línea de investigación es la de Ingeniería de Requisitos.

1.9.3. Tipo de investigación

Aplicada porque se desea obtener un conjunto de factores que permitan identificar los conflictos en los requerimientos no funcionales

1.9.4. Nivel de investigación

Exploratorio porque la concepción de hallar factores que identifican conflictos en los requerimientos no funcionales se encuentra en una fase insípida, descriptiva porque se pretende realizar un análisis del estado del objeto de estudio y experimental porque se intenta ensayar una nueva solución con la finalidad de ayudar a resolver situaciones concretas en la calidad de los productos de software.

1.10. Método y diseño de la investigación

1.10.1. Método de la investigación

Investigación aplicada, empleando el método lógico deductivo.

1.10.2. Diseño de la investigación

Para la investigación se utilizará las siguientes técnicas, instrumentos y materiales de verificación, como se señala en el siguiente cuadro:

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Tabla 1. Variables y técnicas para la investigación

VARIABLES	TECNICAS	INTRUMENTOS DOCUMENTALES
• Necesidades del cliente	Observación Revisión documental	Ficha de observación documental de las aplicaciones informáticas.
• Modelo de asociación	Observación Revisión documental	Ficha de observación documental de las aplicaciones informáticas

Fuente: Elaboración propia

1.10.3. Forma de tratamiento de los datos

Para el procesamiento de los datos provenientes de la aplicación de los instrumentos, se utilizará la sistematización, tabulación, cuadros estadísticos y análisis correspondientes. Para la solución, en función de las herramientas, en general los datos de entrada serán los Requerimientos No Funcionales y los Escenarios de Casos de Uso, se agregará un campo a los Escenarios: RNF asociado para relacionar ambos documentos

Luego se mapea la información a la aplicación y con la librería Elasticsearch se agregarán dos índices. Con la librería se buscarán las incidencias de palabras, concurrencia, entre otros que tengan relación con los RNF y así obtener las dependencias y conflictos mediante el análisis de texto. Finalmente se indexarán estos resultados en un nuevo índice y con ayuda de Kibana se visualizarán los datos.

a) Matriz de sistematización de datos

Para consolidar los datos de la aplicación de la técnica, estos se sistematizarán de acuerdo con los rangos conceptuales previstos en la definición. Esta matriz permite obtener los datos iniciales para formar esquemas que permitan representar las necesidades de los desarrolladores.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

b) Matriz de tabulación

Con el fin de contabilizar las respuestas a las observaciones hechas al caso de estudio que se aplicará.

c) Cuadros estadísticos

Elaboración de cuadros estadísticos descriptivos que permitan visualizar las respuestas correspondientes en términos de indicadores.

d) Análisis

Se llevará a cabo un análisis estadístico descriptivo aplicado a los resultados obtenidos. La descripción de los cuadros de distribución de frecuencias buscara obtener resúmenes adecuados de la información referente a la simulación de datos mediante la aplicación de redes neuronales.

1.11. Cobertura del estudio

1.11.1. Universo

Uno de los mayores problemas que presenta nuestra ciudad es la confección de estadísticas sobre el área de tecnología, a pesar de los esfuerzos que viene realizando el gobierno peruano por medio del Instituto de Estadística e Informática, no se ha podido consolidar esta información por lo que nuestro universo se encuentra conformado por personas dedicadas al desarrollo de software y, más explícitamente, a aquellas personas que se dedican a la Ingeniería de Requisitos.

1.11.2. Muestra

Al no existir estadísticas sobre empresas que utilicen este tipo de técnicas en la construcción de aplicaciones informáticas; el estudio se llevará a cabo únicamente bajo la modalidad de muestreo por cuotas.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

CAPÍTULO 2: MARCO TEÓRICO

2.1. Estado del Arte

- En (Ming-Xun et al, 2012) se propone un conjunto de gráficos interdependientes con el objetivo de presentar metas cualitativas y cuantitativas difusas en lo referente a los requerimientos no funcionales según el modelo FQQSIG (Fuzzy Qualitative and Quantitative Softgoal Interdependency Graphs). Indica que el modelo FQQSIG construye un gráfico jerárquico de criterios que sirven para descomponer los requerimientos no funcionales. Posteriormente, introduce un algoritmo de simulación llamado para transformar el aspecto cualitativo en evaluaciones que sirven para determinar los requerimientos no funcionales. Usa un nuevo algoritmo llamado matriz de relaciones, el cual le permite calcular todos los valores de contribución a los requerimientos no funcionales por el cual los desarrolladores pueden hacer un trueque entre las decisiones que se deben de tomar en cuenta entre dichos requerimientos. En el artículo se discuten y se proporcionan ejemplos para ilustrar el modelo FQQSIG.
- En (Bo et al, 2011) se indica que los requisitos de software, y en especial los requisitos no funcionales, son considerados como una condición imprescindible para la producción de software de alta calidad. Como es ampliamente aceptado, el modelamiento de objetivos no funcionales como son los marcos de trabajo NFR (Non Funtional Requirements), por lo general emplea el modelo de árbol, y presenta un proceso interactivo para el análisis de requisitos no funcionales. Sin embargo, aún existen algunos problemas durante la identificación de estado. El artículo se basa en el conocido modelo de modales objetivo de razonamiento NFR, claramente que distingue a la suposición de mundo cerrado y el mundo abierto, y propone un mecanismo automático para el razonamiento de metamodelos NFR con el fin de identificar los estados que satisfacen las raíces de los árboles objetivo según la contribución que hacen las hojas. Son transformados en estados que satisfacen los objetivos de sus padres. Entonces los padres satisfacen los estados que se desprenderán de acuerdo con el razonamiento de reglas derivadas de diferentes relaciones de descomposición.
- En (Supakkul el al, 2010) se menciona que los requisitos no funcionales (NFR), como la seguridad y el costo, son generalmente subjetivos y muchas veces

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

sinérgico o conflictivas entre sí. Tratar adecuadamente tales NFR requiere una gran cantidad de conocimientos. Sin embargo, existen algunos patrones para hacer frente a este tipo de conocimiento de NFR. En este trabajo, presentan cuatro tipos de patrones para capturar y reutilizar los NFR - patrón objetivo, patrón de problemas, patrón de alternativas y el patrón de selección. Los patrones NFR pueden ser representados visualmente, y organizados bajo una normativa de especialización para así construir patrones más grandes.

- En (Mairinza et al, 2010) se indica que estudios previos revelan que los conflictos entre los requisitos no funcionales (NFR) no siempre son absolutos. También pueden ser relativos en función del contexto del producto a ser desarrollado o implementado. Dado que las técnicas para gestionar los conflictos NFR se centran principalmente en la catalogación de las interrelaciones entre los diversos tipos de NFR, por lo tanto, es necesario implementar una técnica para gestionar los conflictos NFRS. En el artículo se presenta un novedoso marco para gestionar los conflictos entre NFR con respecto a características relativas de NFR. Mediante la aplicación de un enfoque experimental, es obtenida la evidencia cuantitativa de conflictos NFRS. Las métricas y medidas NFRS son utilizadas en los experimentos como parámetros para generar la evidencia cuantitativa. Esta evidencia puede entonces permitir a los desarrolladores identificar y razonar sobre los conflictos NFRS.
- En (Chin-Lun, 2016) se menciona que el analizar los conflictos en los requisitos no funcionales es una tarea importante en grandes proyectos de desarrollo de sistemas de software. Muchos de los requisitos no funcionales que se acumulan con el tiempo pueden variar. Los analistas de sistemas suelen mantener los requisitos no funcionales de forma incremental. Los problemas de sobrecarga de información y la rotación de los empleados pueden complicar la detección de conflicto en el proceso de evolución del requisito no funcional. Este trabajo propone un detector de conflictos en la evolución del requisito no funcional y utiliza ontologías como fundamento teórico para la detección automática de los conflictos.
- En (Tabassum et al, 2014) se indica que la dependencia entre los requisitos no funcionales (NFR) es uno de los principales problemas para manejar la entrega de software de calidad. Estas relaciones de dependencia son las razones de conflicto, aunque también hay relaciones donde un NFR ayuda a otra NFR. Las

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

relaciones de interdependencia, y que no son tratadas desde el principio de un desarrollo de software, pueden causar fallas en las fases posteriores del desarrollo. Para resolver estos problemas, se han determinado las interdependencias entre NFR. Se propone un marco de trabajo para hacer frente a las interdependencias entre NFR desde las primeras etapas del proyecto de desarrollo. El marco de interdependencia NFR propuesto mantiene los requisitos no funcionales con los requisitos funcionales que están asociados; esto ayuda a diseñar el sistema sin salir de los NFR en las etapas posteriores del desarrollo.

- En (Rao, 2011) se indica que la identificación de los requisitos no funcionales es importante para el éxito del desarrollo y despliegue del producto de software. La aceptación del producto de software por parte del cliente depende de los requisitos no funcionales que se incorporan en el software. Para ello, es necesario identificar todos los requisitos no funcionales necesarios por todas las partes interesadas. En la literatura muchos enfoques no están disponibles para este propósito. Por lo tanto, propone un enfoque de cuatro capas de análisis para la identificación de los requisitos no funcionales. El enfoque por capas propuesta tiene muchas ventajas sobre el enfoque que no está en capas. Como parte de este enfoque proponen algunas reglas para ser usado en cada capa. El enfoque se aplica con éxito en dos estudios. Los requisitos no funcionales identificados se validaron utilizando una lista de verificación y, además, la conformidad de los requerimientos no funcionales identificados se calcula utilizando una métrica.

2.2. Marco conceptual

2.2.1. Ingeniería de requerimientos

La Ingeniería de Requisitos cumple un papel primordial en el proceso de producción de software, ya que enfoca un área fundamental: la definición de lo que se desea producir. Esta disciplina establece el proceso de definición de requisitos como una sucesión de actividades mediante la cual lo que debe hacerse se "elicita", se modela y analiza aún a merced de sus complicaciones (figura 1). En este proceso se deben conciliar diferentes puntos de vista y utilizar una combinación de métodos, personas y herramientas. El resultado final constituye la documentación de los requisitos. Éstos deben expresarse de forma clara y estructurada de manera que puedan ser entendidos tanto por

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

expertos como por el usuario, quien deberá participar en la validación. (Wiegers, 2013)(Gómez, 2011)(Chun et al, 1999)

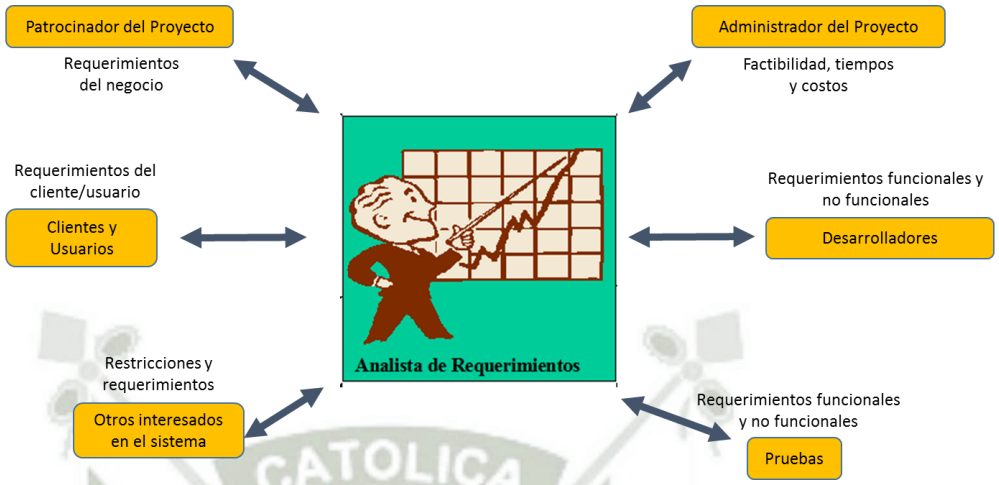


Figura 1. Dificultades de la Ingeniería de Requerimientos
Fuente: Elaboración propia

Para una clara separación de los diferentes niveles de descripción se hace utilizando el término requisitos de usuario, para designar los requisitos abstractos de alto nivel, y requisitos del sistema, para designar la descripción detallada de lo el sistema debe hacer. Los requisitos del usuario, los del sistema y la especificación del diseño del software se definen como se muestra a continuación:

Los requerimientos del usuario son declaraciones, en lenguaje natural y en diagramas de los servicios que se espera que el sistema provea y de las restricciones bajo las cuales debe operar.

Los requerimientos del sistema establecen con detalle los servicios y restricciones del sistema. El documento de requerimientos del sistema, algunas veces denominado especificación funcional, debe ser preciso. (Wiegers, 2013)(Gómez, 2011)

Una especificación del diseño del software es una descripción abstracta del software que es una base para un diseño e implementación detallados. Esta especificación añade detalle a la especificación de requerimientos del sistema. Los requerimientos no funcionales establecen con precisión los elementos que indirectamente afectan a la construcción del producto de software, como, por

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ejemplo: sistema operativo, lenguaje de programación, plataforma de uso, tipo de memoria a emplear, cantidad de memoria RAM, cantidad de memoria caché, entre otros.

Definición de requerimientos del usuario

1. El software debe proveer un medio para representar y acceder a archivos externos creados por otras herramientas.

Especificación de los requerimientos del sistema

1.1 Al usuario se le proveerá con los recursos para definir el tipo de archivos externos.
1.2 Cada tipo de archivo externo tendrá una herramienta asociada que será aplicada al archivo.
1.3 Cada tipo de archivo externo se representará como un icono específico sobre la pantalla del usuario.
1.4 Se proveerán recursos para que el usuario defina el icono que representa un tipo de archivo externo.
1.5 Cuando un usuario selecciona un icono que representa un archivo externo, el efecto de esa selección es aplicar la herramienta asociada con este tipo de archivo al archivo representado por el icono seleccionado.

Figura 2. Requerimientos del usuario y del sistema
Fuente: (Pressman, 2007)

Diferentes niveles de especificación del sistema son de utilidad debido a que comunican la información del sistema a los diferentes tipos de lectores. La figura 2 ilustra que la diferencia entre los requerimientos del usuario y del sistema. Muestra la manera en que un requerimiento del usuario se expande en varios requerimientos del sistema (Gómez, 2011).

2.2.1.1. Requerimientos funcionales y no funcionales

A menudo, los requerimientos de sistemas de software se clasifican en funcionales y no funcionales, o como requerimientos del dominio:

1. Requerimientos funcionales Son declaraciones de los servicios que proveerá el sistema, de la manera en que este reaccionará a entradas particulares y de cómo se comportará en situaciones particulares.

2. Requerimientos no funcionales Son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo, estándares, etcétera. La tabla 2 muestra

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

varias métricas posibles utilizadas para especificar las propiedades no funcionales del sistema con respecto a su construcción.

3. Requerimientos del dominio Son requerimientos que provienen del dominio de aplicación del sistema y que reflejan las características de ese dominio. Estos pueden ser funcionales o no funcionales. (Wiegers, 2013)(Gómez, 2011).

Tabla 2. Métricas para especificar requerimientos no funcionales en la construcción de software

Propiedad	Medida
Rapidez	Transacciones procesadas por segundo Tiempo de respuesta al usuario y a eventos Tiempo de actualización de la pantalla
Tamaño	KB Número de chips de RAM
Facilidad de uso	Tiempo de capacitación Número de cuadros de ayuda
Fiabilidad	Tiempo promedio entre fallas Probabilidad de no disponibilidad Tasa de ocurrencia de las fallas Disponibilidad
Robustez	Tiempo de reinicio después de fallas Porcentaje de eventos que provocan las fallas Probabilidad de corrupción de los datos después de las fallas
Portabilidad	Porcentaje de declaraciones dependientes del objetivo Número de sistemas objetivo.
Reusabilidad	Número de requerimientos reutilizados

Fuente: (Sommerville, 2005)

2.2.1.2. Requerimientos del usuario

Los requerimientos del usuario para un sistema detallan los requerimientos funcionales y no funcionales de tal forma que sean comprensibles por los usuarios del sistema que no posean un conocimiento técnico. Únicamente dan a conocer el comportamiento externo del sistema y evitan, tanto como sea posible, las características de diseño del sistema. Por consiguiente, los requerimientos del usuario no se deben definir utilizando un modelo de implementación. Deben

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

redactarse utilizando el lenguaje natural, representaciones y diagramas intuitivos sencillos. Sin embargo, pueden surgir diversos problemas cuando se redactan el lenguaje natural: (Wiegers, 2013)(Gómez, 2011)

1. Falta de claridad. Algunas veces es difícil utilizar el lenguaje de forma precisa y no ambigua sin detallar el documento y hacerlo difícil de leer.
2. Confusión de requerimientos. No se distinguen claramente los requerimientos funcionales y no funcionales, las metas del sistema y la información para el diseño.
3. Conjunción de requerimientos. Diversos requerimientos diferentes se expresan de forma conjunta como un único requerimiento.

Asimismo, un requerimiento debe cumplir ciertos criterios y características, los cuáles son:

1. Único: El requerimiento debe poder ser interpretado inequívocamente de una sola manera.
2. Verificable: Su implementación debe poder ser comprobada. El test debe dar como resultado CORRECTO o INCORRECTO.
3. Claro: Los requerimientos no deben contener terminología innecesaria. Deben ser establecidos de forma clara y simple.
4. Viable (realista y posible): El requerimiento debe ser factible según las restricciones actuales de tiempo, dinero y recursos disponibles.
5. Necesario: Un requerimiento no es necesario si ninguno de los interesados necesita el requerimiento o bien si la retirada de dicho requerimiento no tiene ningún efecto

Cuando los requerimientos del usuario incluyen demasiada información, restringen la libertad del desarrollador del sistema para proveer soluciones innovadoras a los problemas del usuario y hace que los requerimientos sean difíciles de comprender. Los requerimientos del usuario deben simplemente enfocarse a los recursos principales a proveer. (Sommerville, 2005).

Para minimizar las malas interpretaciones al redactar los requerimientos del usuario, se recomienda seguir unas pautas sencillas para redactar requerimientos:

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

1. Inventar un formato estándar y asegurar que todos los requerimientos se adhieren al formato.
2. Utilizar el lenguaje de forma consistente. En particular distinguir entre los requerimientos deseables y obligatorios.
3. Resaltar el texto (con negritas o itálicas) para ver las partes claves del requerimiento.
4. Evitar, hasta donde sea posible, utilizar el lenguaje "técnico" de computación.

2.2.1.3. Requerimientos del sistema

Éstos son descripciones más detalladas de los requerimientos del usuario. Sirven como base para definir el contrato de la especificación del sistema y, por lo tanto, debe ser una especificación completa y consistente del sistema. Son utilizados por los ingenieros de software como el punto de partida para el diseño del sistema (Wiegers, 2013)(Gómez, 2011).

La especificación de requerimientos del sistema incluye diferentes modelos del sistema como el de objetos o el de flujo de datos. En principio, los requerimientos del sistema deberán establecer lo que éste hará y no la manera en que se implementará. Sin embargo, en el nivel de detalle requerido para especificar el sistema completamente, es casi imposible excluir toda la información de diseño, esto es debido a que

1. Una arquitectura inicial del sistema se define para ayudar a estructurar la especificación de requerimientos.
2. En muchos casos, los sistemas deben operar con otros sistemas ya existentes.
3. El uso de un diseño específico es un requerimiento externo del sistema.

A menudo se utiliza el lenguaje natural para redactar las especificaciones de requerimientos del sistema. Sin embargo, pueden surgir problemas adicionales con el lenguaje natural si se utiliza para detallar la especificación. (Wiegers, 2013):

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

2.2.1.4. El documento de requerimientos del software

Éste es la declaración oficial de qué es lo que requieren los desarrolladores del sistema. Incluyen tanto los requerimientos del usuario par el sistema como una especificación detallada de los requerimientos del sistema. En algunos casos, los dos tipos de requerimientos se integran de una única descripción. En otros, los del usuario se definen en una introducción de la especificación de los del sistema. Si existen un gran número de requerimientos, los detalles de requerimientos del sistema se pueden presentar como documentos separados: (Pohl, 2013)(Gómez, 2011)

- Especificará únicamente el comportamiento externo del sistema.
- Especificará las restricciones de la implementación.
- Será fácil de cambiar.
- Servirá como herramienta de referencia para los mantenedores del sistema.
- Registrará las previsiones del ciclo de vida del sistema.
- Caracterizará las respuestas aceptables para los eventos no deseados.

Algunas veces es difícil especificar los sistemas en términos de lo que hacen (su comportamiento externo). Inevitablemente, debido a las restricciones existentes en los sistemas, el diseño de éste está restringido y esto se debe reflejar en el documento de requerimientos. El requisito de registrar las previsiones del ciclo de vida del sistema se acepta ampliamente, pero no se sigue al redactar los documentos de requerimientos. (Pohl, 2013)(Gómez, 2011)

2.2.1.5. Gestión y procesos de la ingeniería de requerimientos

Primero se define el papel que juegan la gestión y la tecnología, para poder aplicar la ingeniería de requerimientos, por tal motivo se analizan los conceptos a partir de las aportaciones de Siqueira (2015).

En términos generales, los conceptos de administración, gerencia y gestión son sinónimos, a pesar de los grandes esfuerzos y discusiones que se han hecho para diferenciarlos. En la práctica se observa que el

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

término management ha sido traducido como administración, pero también como gerencia.

En esa concepción, le corresponde al gerente echar una mirada al entorno de modo que la organización pueda generar desarrollo: tomar recursos y producir más recursos. Al administrador le corresponde más el mantenimiento y conservación, mientras que a la administración se le concibe funcional o vertical (Siqueira, 2015).

La gestión es lineal o tradicional donde es sinónimo de administración. Por gestión se entiende el conjunto de diligencias que se realizan para desarrollar un proceso o para lograr un producto determinado. Se asume como la dirección y el gobierno de actividades para hacer que las cosas funcionen, con capacidad para generar procesos de transformación de la realidad. (Mora, 1999)

La Gestión de Requerimientos del área de TI, tiene como finalidad dar apoyo a las aéreas internas de la empresa. Cualquier requerimiento de las aéreas internas de la empresa puede ser solicitado por cualquier empleado, pero debe ser aprobado por su Jefe inmediato superior, luego el área de TI se encarga de evaluar el impacto que tendrá dentro de la organización, si la repercusión es muy grande el responsable del área de TI convocará a una reunión extraordinaria a las partes involucradas para obtener un consenso y aprobación por escrito sobre los nuevos cambios que tendrán en la empresa. (Bartolo, 2012)

Los requisitos son la base de todos los proyectos que involucren el desarrollo de software. La definición de los usuarios, clientes, proveedores, desarrolladores y empresas involucradas en la creación de un nuevo sistema, dará como resultado lo que el sistema debe de hacer para satisfacer las necesidades de dicho proyecto.

Es conocido que muchos proyectos de software fracasan porque los analistas no conceptualizan una debida educción, elicitación y especificación de requisitos de software. Comúnmente no se hace participar al usuario final por lo que se llega a identificar requerimientos incompletos e insistentes.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Al acordar los requisitos para el sistema, se proporcionan las bases para la planificación del desarrollo de un proyecto de software. Estos deben ser entendidos por todos los involucrados en el proyecto, expresados en un lenguaje común, captando la necesidad del problema por completo, sin ambigüedades.

Los requisitos permiten la gestión de los riesgos, inclusive desde que se inicia el desarrollo del sistema. Los riesgos planteados pueden ser rastreados, su impacto evaluado y los efectos del plan mitigados mediante grandes ahorros en el proyecto.

La ingeniería de requerimientos sirve para determinar las necesidades que un cliente tiene en la implementación o desarrollo de software, mediante el proceso de recopilar, analizar y verificar la especificación de los requisitos del software de una manera completa y correcta (Chaves, 2011). Además, contribuye a construir un producto de software de alta calidad, bajo las restricciones de tiempo y presupuesto, donde se requiere de rigor, creatividad, documentación y gestión en todas sus actividades (Pohl, 2013).

Los requerimientos son importantes para la construcción de software, en ella se debe tomar en cuenta la planificación en donde se incluye tiempo y costo; por otro lado, se toman en cuenta todos los recursos y el correspondiente cronograma.

La ingeniería de requerimientos es un proceso que comprende todas las actividades requeridas para crear y mantener un documento de requerimientos del sistema. Existen cuatro actividades genéricas de alto nivel en el proceso de ingeniería de requerimientos. Estas son un estudio de factibilidad del sistema, la obtención y el análisis de requerimientos, la especificación de éstos y su documentación y, finalmente, la validación. La Figura 3 ilustra la relación entre estas actividades. También muestra el documento que se produce en cada etapa del proceso de ingeniería de requerimientos (IR) (Wiegers, 2013)(Gómez, 2011).

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

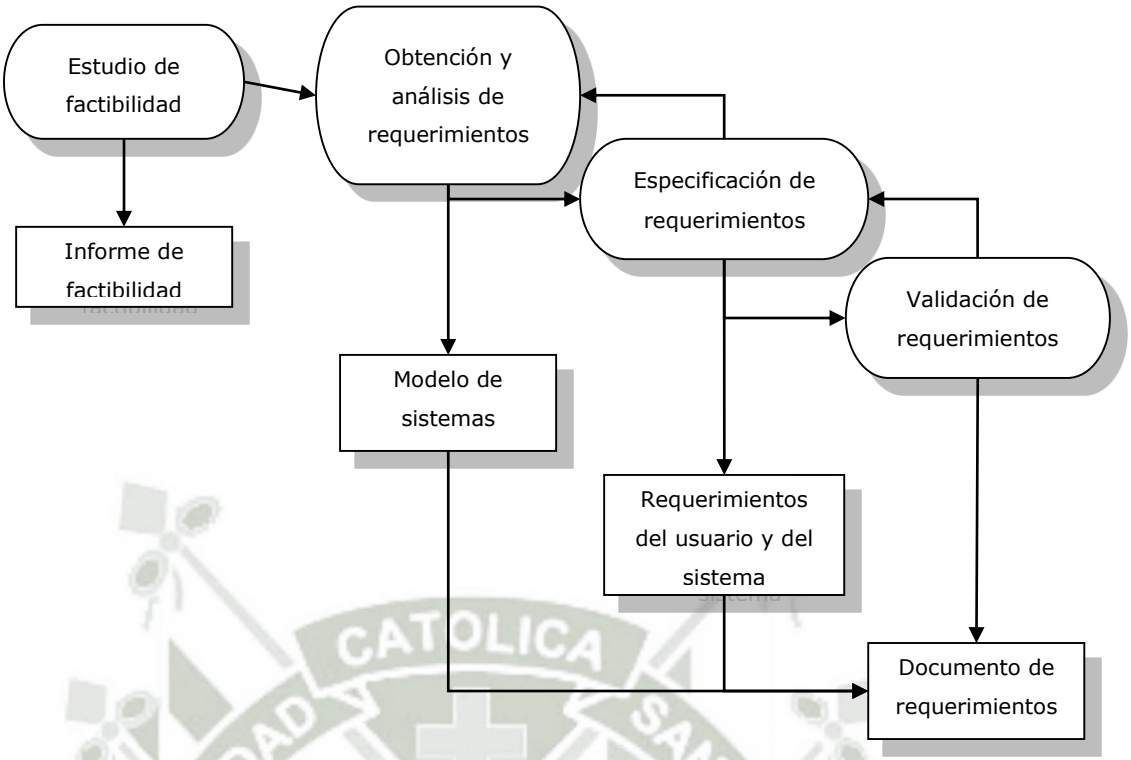


Figura 3. El proceso de ingeniería de requerimientos
Fuente: (Sommerville, 2005)

a) Estudios de factibilidad

Cuando se construyen nuevos sistemas, todo comienza con un estudio de factibilidad. La información inicial es un concepto genérico del sistema teniendo como visión una descripción resumida del sistema y de cómo se utilizará dentro de una organización. El resultado de este estudio es la recomendación sobre si conviene hacer la ingeniería de requerimientos.

El estudio de factibilidad intenta resolver las siguientes interrogantes:

1. ¿El nuevo sistema contribuirá a los objetivos de la organización?
2. ¿Se podrá construir el sistema empleando tecnología actual y con un conjunto de restricciones?
3. ¿Podrá el sistema coexistir con otros sistemas elaborados para la organización?

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Aquí la cuestión crítica es si el sistema contribuye a los objetivos del negocio. Si no contribuye a estos, entonces no tiene valor real en el negocio.

b) Obtención y análisis de requerimientos

En la presente actividad, los ingenieros de requisitos trabajan con los clientes y los usuarios finales del sistema para determinar el modelo de negocio de la futura aplicación, así como servicios y desempeños del sistema, las limitaciones de hardware, entre otros (Gómez, 2011).

La obtención y análisis de requerimientos incluyen diferentes tipos de personas de la organización. El termino stakeholder se utiliza para referirse a cualquier persona que tiene influencia directa o indirecta sobre los requerimientos del sistema. Entre los stakeholders también son los ingenieros que desarrollan o dan mantenimiento a otros sistemas relacionados, los administradores de negocio, los expertos en el dominio del sistema, los representantes de los trabajadores, y otros. En la Figura 4 se muestra un modelo genérico del proceso de obtención y análisis.

Cada organización tendrá su propia versión o instancia de este modelo general dependiendo de los factores locales, como la habilidad del personal, el tipo de sistema a desarrollar, los estándares utilizados, etcétera. (Sommerville, 2005)

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

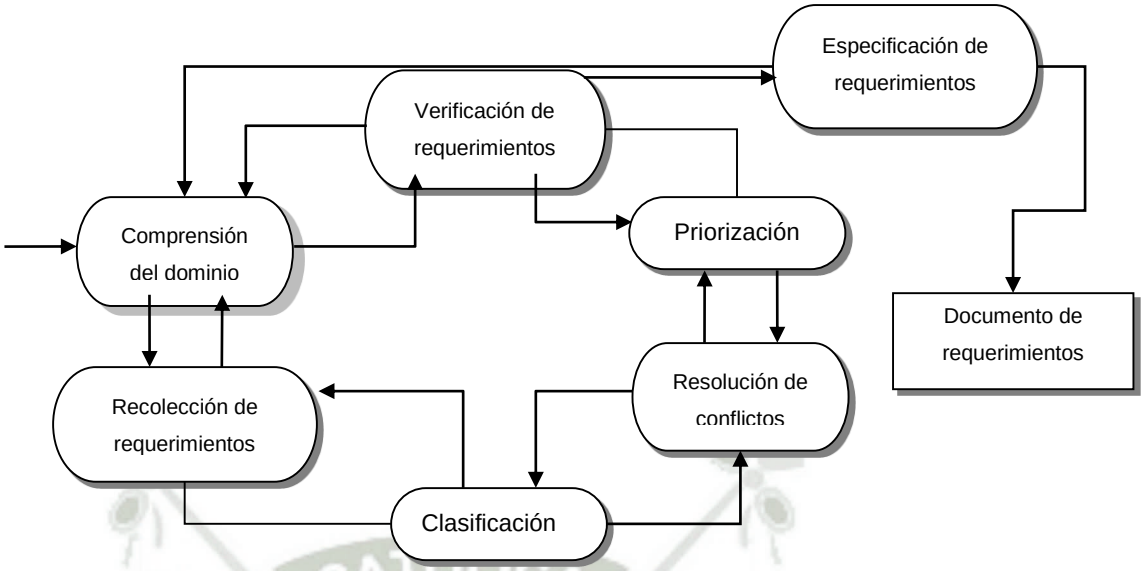


Figura 4. Proceso de obtención y análisis de requerimientos
Fuente: (Sommerville, 2005)

Las actividades del proceso son:

1. Comprensión del dominio. El analista debe desarrollar su propia comprensión del dominio de la aplicación.
2. Recolección de requerimientos. Éste es el proceso de interactuar con los stakeholders del sistema para descubrir sus requerimientos. Obviamente, la comprensión del dominio se desarrollará más durante esta actividad.
3. Clasificación. Esta actividad considera la recolección no estructurada de requerimientos y los organiza en grupos coherentes.
4. Resolución de conflictos. De forma inevitable, cuando existen muchos stakeholders involucrados, los requerimientos entrarán en conflicto. Esta actividad se refiere a encontrar y resolver estos conflictos.
5. Priorización. En cualquier conjunto de requerimientos, alguno de ellos será más importante que los otros.
6. Verificación de requerimientos. Los requerimientos se verifican para descubrir si están completos, son consistentes y acordes con lo que realmente quieren los stakeholders del sistema.

La figura 4 muestra que la obtención y análisis de requerimientos es un proceso iterativo con retroalimentación continua de cada

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

actividad a las otras actividades. El ciclo del proceso inicia con la comprensión del dominio y termina con la verificación de requerimientos. La comprensión de los requerimientos por parte del analista mejora en cada vuelta de ciclo.

c) Gestión de requerimientos

En la práctica, en casi todos los sistemas los requerimientos cambian. Las personas involucradas desarrollan una mejor comprensión de lo que quieren que haga el software; la organización que compra el sistema cambia; se hacen modificaciones a los sistemas de hardware, software y al entorno organizacional. El proceso de organizar y llevar a cabo los cambios en los requerimientos se llama gestión de requerimientos. (Gómez, 2011)

El objetivo del analista es reconocer los elementos básicos de un sistema tal como lo percibe el usuario/cliente. El analista debe establecer contacto con el equipo técnico y de gestión del usuario/cliente y con la empresa que vaya a desarrollar el software. El gestor del programa puede servir como coordinador para facilitar el establecimiento de los caminos de comunicación.

Una vez que un sistema se ha instalado, inevitablemente surgen nuevos requerimientos. Es difícil para los usuarios y clientes del sistema anticipar qué efectos tendrá el sistema nuevo en la organización. Cuando los usuarios finales tienen experiencia con un sistema, descubren nuevas necesidades y prioridades. (Gómez, 2011)

Los clientes y los usuarios, rara vez son las mismas. Los clientes imponen y/o cambian requerimientos debido a las limitaciones de la organización y obviamente del presupuesto.

Otro motivo por el que cambian los requerimientos es que en ocasiones el entorno de negocios y técnico del sistema cambian después de la instalación. Puede ser que se introduzca un nuevo hardware, o puede ser que surja la necesidad de que el sistema interactúe con otros sistemas. También cambian las prioridades del

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

negocio, las legislaciones y las regulaciones, y esto debe estar reflejado en sistema. (Gómez, 2011)

La gestión de requerimientos es el proceso de comprender y controlar los cambios en los requerimientos del sistema bajo cierta cultura (figura 5). Es necesario mantenerse al tanto de los requerimientos particulares y mantener vínculos entre los requerimientos dependientes de forma que se pueda evaluar el impacto de los cambios en los requerimientos. El proceso de gestión de requerimientos debe empezar cuando esté disponible una versión preliminar del documento de requerimientos. Hay que establecer un proceso formal para implantar las propuestas de cambios y planear como se van a gestionar los requerimientos que cambian durante el proceso de obtención de requerimientos. (Sommerville, 2005)

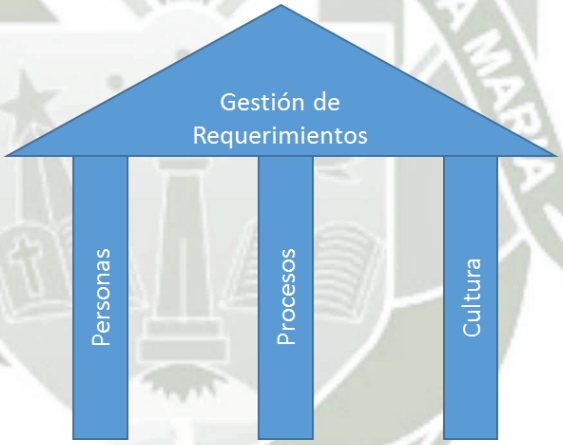


Figura 5. Pilares de la Gestión de Requerimientos
Fuente: (Pressman, 2007)

d) Técnicas para la Obtención y análisis de requerimientos

No existe un enfoque perfecto y universal aplicable a la obtención y el análisis de requerimientos. Normalmente, es necesario utilizar varios de estos enfoques para desarrollar un entendimiento completo y un análisis de requerimientos (Wiegers, 2013)(Gómez, 2011).

1. Obtención orientada a puntos de vista

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Para cualquier sistema grande o de mediano tamaño, existen diferentes tipos de usuarios finales. Muchos stakeholders tienen un cierto interés en los requerimientos del sistema.

Los diferentes puntos de vista de un problema consideran a este de diferentes formas. Sin embargo, sus perspectivas no son completamente independientes, sino que se traslapan, por lo que tienen requerimientos comunes.

Un punto clave del análisis orientado a puntos de vista es que toma en cuenta la existencia de varias perspectivas y provee un marco de trabajo para descubrir conflictos en los requerimientos propuestos por diferentes stakeholders.

Métodos diferentes tienen ideas diferentes de lo que significa un "punto de vista". Este se puede considerar como:

- Una fuente o consumidor de datos. En este caso, los puntos de vista son responsables de producir o consumir datos.
- Un marco de trabajo de la representación. En este caso, un punto de vista se considera un tipo particular del modelo del sistema.
- Un receptor de servicios. En este caso, los puntos de vista son externos al sistema y reciben servicios de él.

2. Escenarios

Normalmente, las personas encuentran más fácil dar ejemplos de la vida real que descripciones abstractas. Pueden comprender y criticar un escenario de cómo podría interactuar con un sistema de software. Los ingenieros de requerimientos pueden utilizar la información obtenida de esta discusión para formular los requerimientos reales del sistema.

Los escenarios pueden ser especialmente útiles para agregar detalle a un bosquejo de descripción de requerimientos. Son descripciones de ejemplos de las sesiones de interacción. Cada escenario cubre uno o un número reducido de posibles interacciones. Diferentes formas de escenarios se han desarrollado y proveen diferentes tipos de información en diferentes niveles de detalles del sistema. (Sommerville, 2005)

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

El escenario inicia con un bosquejo de la interacción y, durante la obtención, se agregan detalles para crear una descripción completa de esa interacción. De forma general, un escenario incluye:

1. Una descripción del estado del sistema al inicio del escenario.
2. Una descripción del flujo normal de eventos en el escenario.
3. Una descripción de lo que puede ir mal y cómo manejarlo.
4. Información de otras actividades que se podrían llevar a cabo al mismo tiempo.
5. Una descripción del estado del sistema después de completar el escenario.

Es posible llevar a cabo de manera informal la obtención de requerimientos basada en escenarios cuando los ingenieros de requerimientos trabajan con los stakeholders en la identificación de escenarios y en la captura de detalles de dichos escenarios. De forma alternativa, se puede utilizar un enfoque más estructurado como los escenarios de eventos o los casos de uso. (Sommerville, 2005)

3. Escenarios de eventos

Son utilizados para especificar el comportamiento del sistema cuando se le presenta eventos particulares. Estos eventos de interacción distinto se documentan como un escenario de eventos diferente. Los escenarios de eventos incorporan un detalle del flujo de datos y las acciones del sistema y especifican de igual manera las excepciones que pueden ocurrir. La Figura 6 muestra el escenario del evento.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

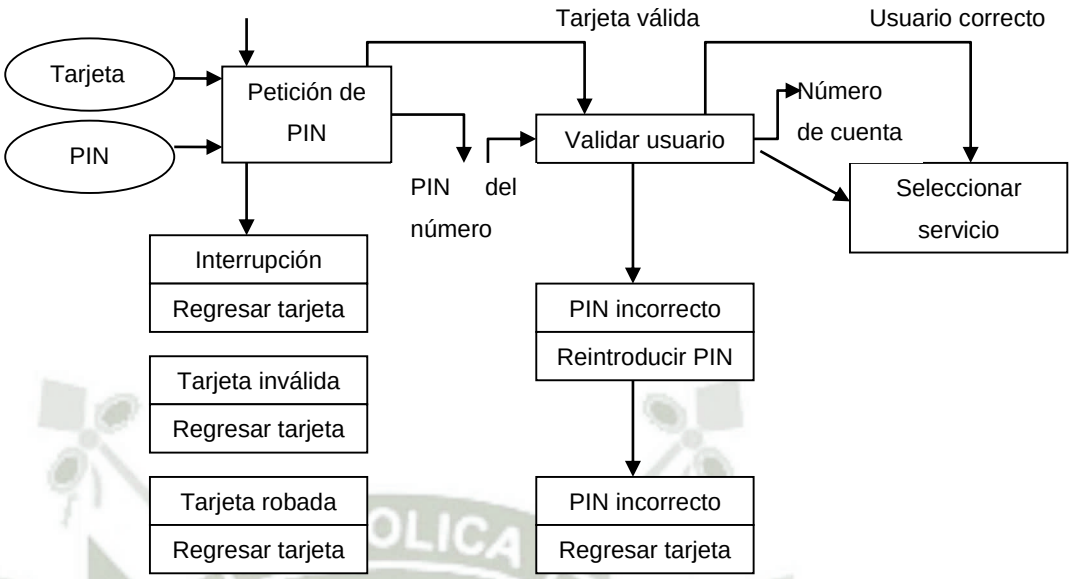


Figura 6. Escenario de eventos - Transacción de inicio
Fuente: Elaboración propia

4. Casos de uso

Estos son una técnica que se basa en escenarios para la obtención de requerimientos que se introdujeron por primera vez en el método Objetory (Jacobson et al., 2000).

Los actores en el proceso se representan como figuras delineadas y cada clase de interacción se representa como una elipse con su nombre. El conjunto de casos de uso representa todas las posibles interacciones que ocurrirán en los requerimientos del sistema. Esto se ilustra en la figura 7. (Sommerville, 2005)

Los escenarios se prestan a confusión cuando se trata de escribir casos de uso; el criterio general es que un caso de uso encapsula un conjunto de escenarios. En este caso habrá un escenario para la interacción normal y escenarios adicionales para las posibles excepciones.

Dentro de UML, los diagramas de secuencia se utilizan para agregar información a un caso de uso. Estos diagramas

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

muestran a los actores involucrados en la interacción, los objetos del sistema como los que pueden interactuar y las operaciones asociadas con estos objetos.

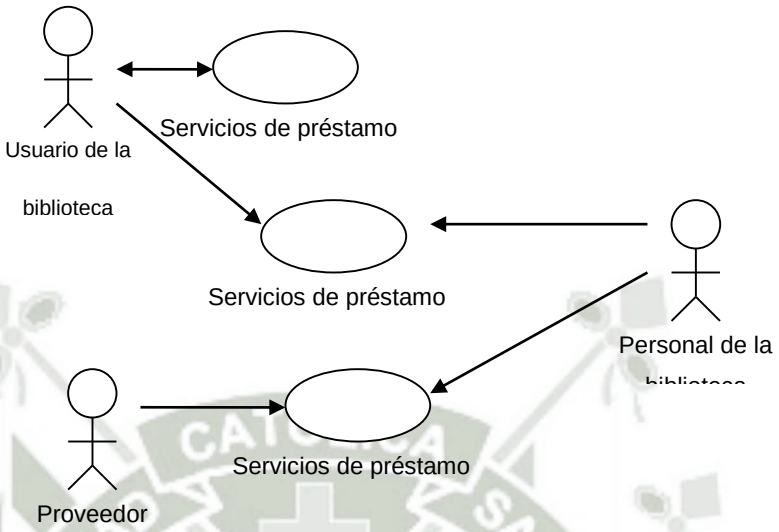


Figura 7. Casos de uso para una biblioteca
Fuente: Elaboración propia

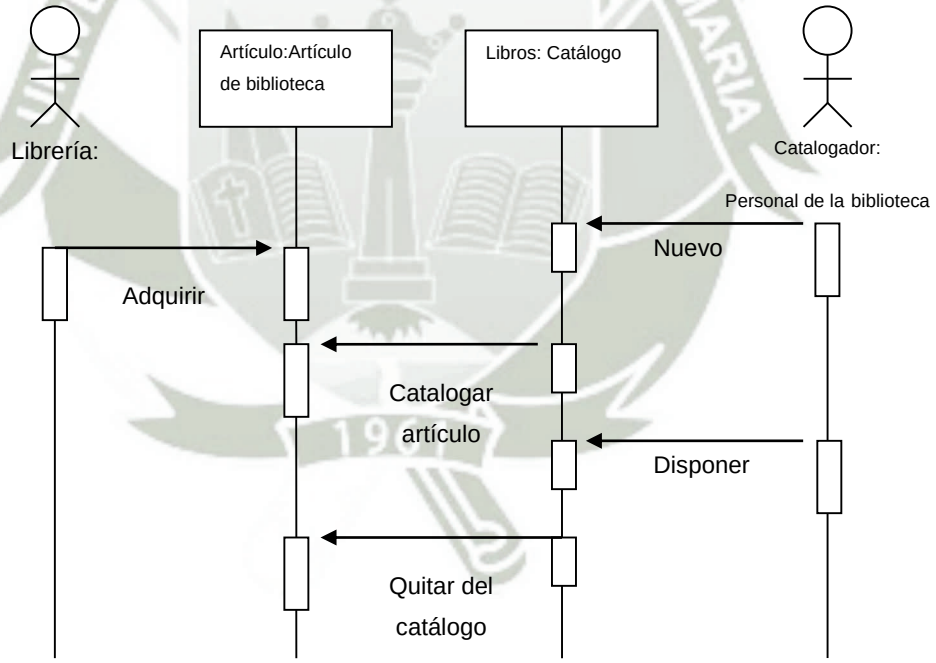


Figura 8. Diagrama de secuencia
Fuente: Elaboración propia

5. Etnografía

La etnografía es una técnica de observación que se puede utilizar para entender los requerimientos sociales y

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

organizacionales. Un analista se sumerge por sí solo en el entorno laboral donde el sistema se utilizará. El trabajo diario se observa y se hacen notas de las tareas reales en las que los participantes están involucrados. El valor de la etnografía en que ayuda a descubrir los requerimientos implícitos que reflejan los procesos reales más que los formales en los que la gente está involucrada (Velasco, 2009).

La etnografía es especialmente efectiva para descubrir dos tipos de requerimientos:

1. Los requerimientos que se derivan de la forma en la que la gente trabaja realmente más que de la forma en la que las definiciones de los procesos establecen que debería trabajar.
2. Los requerimientos que se derivan de la cooperación y conocimiento de las actividades de la gente.

La etnografía se puede combinar con la construcción de prototipo (ver figura 9). La etnografía suministra información al desarrollo del prototipo de forma que se requieran menos ciclos de refinación (Velasco, 2009).

Los estudios etnográficos pueden revelar los detalles de los procesos críticos que otras técnicas de obtención de requerimientos a menudo olvidan. Sin embargo, puesto que se centran en el usuario final, este enfoque no es apropiado para descubrir los requerimientos organizacionales o del dominio (Velasco, 2009). No siempre puede identificar nuevas propiedades a agregar al sistema. Por lo tanto, la etnografía no es un enfoque completo para la obtención de requerimientos y debe utilizarse en conjunto con otros enfoques, como el análisis de casos de uso.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

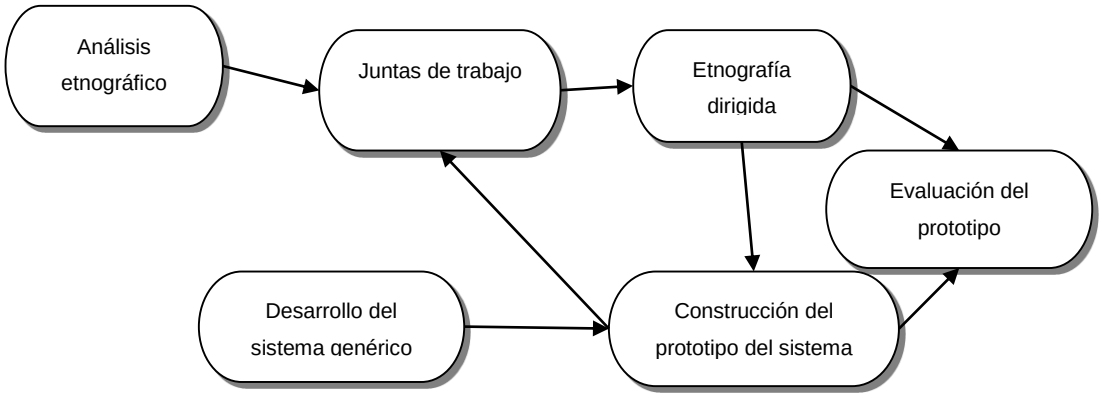


Figura 9. Etnografía y construcción de prototipos
Fuente: (Sommerville, 2005)

e) Validación de requerimientos

Esta validación implica satisfacción del cliente, sin embargo, son procesos distintos puesto que la validación comprende un bosquejo completo del documento de requerimientos mientras que el análisis implica trabajar con requerimientos incompletos (Wiegers, 2013).

Las verificaciones que se suelen realizar incluyen:

1. Verificación de validez. Existe una diferencia entre lo que se necesita y lo que se debe de hacer.
2. Verificaciones de consistencia. No existen requerimientos contradictorios ni ambiguos.
3. Verificaciones de integridad. Se comprueba que los requerimientos deben adoptar funciones y restricciones del usuario.
4. Verificaciones de realismo. Requerimientos que pueden implementarse.
5. Verificabilidad. Escritura de requerimientos que puedan verificarse.

Existen varias técnicas de validación de requerimientos que pueden utilizarse en conjunto o de forma individual:

1. Revisiones de requerimientos. Los requerimientos son analizados sistemáticamente por un equipo de revisores.
2. Construcción de prototipos. En este enfoque de validación, se muestra un modelo ejecutable del sistema a los usuarios finales y los clientes. Estos pueden hacer experimentos con este

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

modelo para ver si cumple sus necesidades reales. (Sommerville, 2005)

3. Generación de casos de prueba. De forma ideal, los requerimientos deben poder probarse. Si las pruebas para éstos se consideran como parte del proceso de validación, esto a menudo revela los problemas en los requerimientos. Si una prueba es difícil o imposible de diseñar, por lo regular esto significa que los requerimientos serán difíciles de implementar y deberían ser considerados nuevamente. (Sommerville, 2005)
4. Análisis de consistencia automático. Si los requerimientos se expresan como un modelo del sistema en una notación estructurada o formal, entonces las herramientas CASE deben verificar la consistencia del modelo. Para verificar la consistencia, la herramienta CASE debe construir una base de datos de requerimientos y después, utilizando las reglas del método o notación, verificar todos los requerimientos en dicha base de datos. Un analizador de requerimientos produce un informe de las inconsistencias recién descubiertas. (Sommerville, 2005)

f) Especificación de requerimientos

La Especificación es un documento que define, de forma completa, precisa y verificable, los requisitos, el diseño y el comportamiento u otras características, de un sistema o componente de un sistema. Para realizar bien el desarrollo de software es esencial tener una especificación completa de los requerimientos. Independientemente de lo bien diseñado o codificado que esté, un sistema pobremente especificado decepcionará al usuario y hará fracasar el desarrollo.

La forma de especificar tiene mucho que ver con la calidad de la solución. Los ingenieros de software que se han esforzado en trabajar con especificaciones incompletas, inconsistentes o mal establecidas han experimentado la frustración y confusión que invariablemente se produce. (Sommerville, 2005)

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Los requerimientos de software pueden ser analizados de varias formas diferentes. Las técnicas de análisis pueden conducir a una especificación en papel que contenga las descripciones gráficas y el lenguaje natural de los requerimientos del software. La construcción de prototipos conduce a una especificación ejecutable, esto es, el prototipo sirve como una representación de los requerimientos. Los lenguajes de especificación formal conducen a representaciones formales de los requerimientos que pueden ser verificados o analizados. (Sommerville, 2005)

La especificación, independientemente del modo en que se realice, puede ser vista como un proceso de representación. Los requerimientos se representan de forma que conduzcan finalmente a una correcta implementación del software. Sommerville plantea que una buena especificación debe procurar: (Sommerville, 2005)

- Separar funcionalidad de implementación. Una especificación es una descripción de lo que se desea, en vez de como se realiza. Esto en la práctica puede llegar a no suceder del todo, sin embargo, es un buen lineamiento a seguir.
- Una especificación debe abarcar el entorno en el que el sistema opera. Similarmente, el entorno en el que opera el sistema y con el que interactúa debe ser especificado.
- Debe ser modificable. Ninguna especificación puede ser siempre totalmente completa. El entorno en el que existe es demasiado complejo para ello. Una especificación es un modelo, una abstracción, de alguna situación real (o imaginada). Por tanto, será incompleta. Además, al ser formulada existirán muchos niveles de detalle.

g) Administración de requerimientos

La planeación comienza al mismo tiempo que la obtención de requerimientos inicial y la administración activa de requerimientos debe iniciar tan pronto como esté, lista la primera versión del documento de requerimientos (Pohl, 2013)(Gómez, 2011).

1. Requerimientos duraderos y volátiles

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

El desarrollo de requerimientos de software centra su atención en las capacidades de éste, los objetivos del negocio y otros sistemas de la organización. Especificar el desarrollo de un sistema grande puede llevar varios años. Con el tiempo, el entorno del sistema y los objetivos del negocio seguramente cambiará. Por lo tanto, los requerimientos deben evolucionar para reflejar esto. Desde una perspectiva evolutiva, los requerimientos son de dos clases:

a) Requerimientos duraderos. Éstos son relativamente estables que se derivan de la actividad principal de la organización y que están relacionados directamente con el dominio del sistema.
b) Requerimientos volátiles. Éstos cambiarán probablemente durante el desarrollo del sistema o después de que éste se haya puesto en operación.

Los requerimientos volátiles pueden ser de las siguientes clases, como se muestra en la Tabla 3:

Tabla 3. Clasificación de requerimientos volátiles

Tipos de requerimientos	Descripción
Requerimientos mutantes	Requerimientos que cambian debido a los cambios en el ambiente en el que opera la organización.
Requerimientos emergentes	Requerimientos que emergen al incrementarse la comprensión del cliente en el desarrollo del sistema. El proceso de diseño puede revelar requerimientos emergentes nuevos.
Requerimientos consecutivos	Requerimientos que son resultado de la introducción del sistema de cómputo
Requerimientos de compatibilidad	Requerimientos que dependen de sistemas particulares o procesos de negocios dentro de la organización. Cuando estos últimos cambian, los requerimientos de compatibilidad del sistema contratado o a entregar también pueden cambiar.

Fuente: (Pressman, 2007)

2. Planeación de la administración de requerimientos

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Ésta es una primera etapa esencial del proceso de administración de requerimientos. La administración de requerimientos es muy cara y, para cada proyecto, la etapa de planeación establece el nivel de detalle necesario en la administración de requerimientos. Durante la etapa de administración de requerimientos se tiene que decidir sobre:

a) La identificación de requerimientos. Cada requerimiento se debe identificar de forma única de tal forma que puedan entrar en referencia cruzada con otros requerimientos de manera que pueda utilizarse en las evaluaciones de rastreo.

b) Un proceso de administración del cambio. Éste es el conjunto de actividades que evalúa el impacto y costo de los cambios.

c) Políticas de rastreo. Éstas definen la relación entre requerimientos y la de éstos en el diseño del sistema que se debe registrar y la manera en que estos registros se deben mantener.

d) Ayuda de herramientas CASE. La administración de requerimientos comprende el procesamiento de grandes cantidades de información de los requerimientos.

3. Administración del cambio de los requerimientos

Esta administración se aplica a todos los cambios propuestos en los requerimientos. La ventaja de utilizar un proceso formal para administrar el cambio es que todos los cambios propuestos son tratados de forma consistente y que los cambios en el documento de requerimientos se hacen de forma controlada. Existen tres etapas principales en un proceso de administración de cambio:

1. Análisis de problema y especificación del cambio.
2. Análisis del cambio y costeo.
3. Implementación del cambio.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

2.2.2. Escenarios

La construcción de aplicaciones informáticas es difícil. La Ingeniería de Requisitos es clave cuando se trata de construcción de calidad ya que se debe de representar el modelo del negocio. La metodología basada en el uso de LEL (Language Extended Lexicon) permite concebir un vocabulario claro y contundente del macrosistema, así como de las escenas para modelar el comportamiento por medio de tarjetas CRC (Bertolami, 2001).

Es común pensar que la confiabilidad del sistema permita deducir la confiabilidad de la información resultante. Los errores que se dejan en la etapa de construcción de software pueden desencadenar una serie de perjuicios. Por mucho de que se tenga que producir con sumo cuidado, el desarrollo todavía puede ser informal. La toma de requerimientos cumple un papel fundamental en la construcción ya que de su definición depende la buena producción de software. En este proceso se deben conciliar diferentes puntos de vista y utilizar una combinación de métodos, personas y herramientas (Wiegers, 2013).

El resultado final constituye la documentación de los requisitos. Éstos deben expresarse de forma clara y estructurada de manera que puedan ser entendidos tanto por expertos como por el usuario, quien deberá participar en la validación. Existen varios enfoques para analizar los requisitos, cuyas especificaciones son en general producidas teniendo en cuenta al usuario. El diseño centrado en el punto de vista del usuario para definir los requisitos es el enfoque al que adhieren recientes metodologías, por ejemplo, OBA y Objectory que propone la especificación de "casos de uso" que complementen satisfactoriamente las características que propone el mencionado enfoque. Durante esta etapa la interacción con el usuario es indispensable, por lo cual estas herramientas deben ser relativas al lenguaje que ellos manejan para facilitar la comunicación. En Leite se propone el uso de LEL (Language Extended Lexicon) que permite definir todo macrosistema. Las herramientas fundamentan la línea base (Ruidias, 2014).

Como consecuencia de esta actividad se logró un ámbito de trabajo razonablemente independiente pero bien predispuesto para aplicarla, lo que de alguna manera se constituyó en una suerte de beta-test de la misma. La metodología fue seguida etapa por etapa, respetando las heurísticas

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

establecidas en forma estricta. Esta forma de trabajo permitió detectar algunas imprecisiones que no resultan visibles en un estudio teórico del método.

Los escenarios se derivan del léxico del macrosistema. Estos son empleados para describir el comportamiento del sistema basado en la visión del usuario. El modelo de escenarios forma parte de la extensión propuesta a la original de "Requirements Baseline" con el objetivo de modelar aspectos de comportamiento. Esta propuesta es una combinación de ideas presentadas en la literatura. Incorpora los siguientes conceptos respecto de los escenarios: i) evolucionan durante el proceso de desarrollo de software, ii) están naturalmente conectados con el LEL, iii) describen situaciones utilizando lenguaje natural (Kaplan, 2002).

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

CAPÍTULO 3: MODELO DE ASOCIACIÓN

3.1. Introducción

El presente modelo de asociación intenta que los requisitos no funcionales queden de manera clara y precisa ya que los mismos influyen en los requisitos funcionales. El hecho de asumir las características de los escenarios y casos de uso hacen que este modelo sea flexible y presente resultados orientados a otorgar soluciones adecuadas cuando de construcción de software se trata.

Aunque una de sus desventajas es el tiempo de construcción, se piensa que la poca cantidad de requisitos no funcionales, que es genérico en la construcción de sistemas, no conllevan a un gran tiempo de elaboración como sí es necesario en los requisitos funcionales. La figura 11 muestra el ciclo de vida del modelo de asociación propuesto.

La flexibilidad del modelo de asociación permite que por medio de sus iteraciones sucesivas se logre encontrar requisitos no funcionales que presenten una mayor solidez en su constitución, permitiendo que el efecto a los requisitos funcionales quede claro. Los documentos o artefactos con los que se trabajará en el modelo son:

- Documento de educción de requisitos no funcionales
  - Atributo: requisitos no funcionales educcionados
  - Atributo: requisitos no funcionales educcionados con criterios de ambiguedad
- Documento de elicitación de requisitos no funcionales
  - Atributo: requisitos no funcionales elicitados
  - Atributo: requisitos no funcionales elicitados con criterios de ambigüedad
- Documento de especificación de requisitos de software no funcionales
  - Atributo: requisitos de software no funcionales especificados
  - Atributo: requisitos de software no funcionales especificados con criterios de ambigüedad
- Casos de uso en donde se encontrarán los escenarios
  - Atributo: casos de uso no ambiguos
  - Atributo: casos de uso con insuficiente claridad
- Documento de dependencias

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

- Atributo: dependencias observadas

Un requisito no funcional es un requisito que da a conocer criterios que pueden utilizarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos. Por tanto, se refieren a todos los requisitos que no describen información a guardar, ni funciones a realizar, sino características de funcionamiento, por eso en el presente trabajo de investigación no se presenta una tipificación de los mismos.

Para determinar los conflictos entre los requerimientos no funcionales se utilizará el análisis de textos por medio de la herramienta Elasticsearch, el mismo que es un servidor de búsqueda basado en Lucene y provee un motor de búsqueda de texto completo, distribuido y con capacidad de multi-tenencia con una interfaz web RESTful y con documentos JSON. Herramienta desarrollada en Java y está publicado como código abierto bajo las condiciones de la licencia Apache. Para la visualización de datos se empleará la herramienta Kibana, la misma que es un complemento de visualización de datos de código abierto para Elasticsearch. Proporciona capacidades de visualización sobre el contenido indexado en un clúster de Elasticsearch. Los usuarios pueden crear gráficos de barra, línea y dispersión, o gráficos circulares y mapas. La figura 10 muestra cómo se engranan estas herramientas para obtener los conflictos de los requerimientos no funcionales.

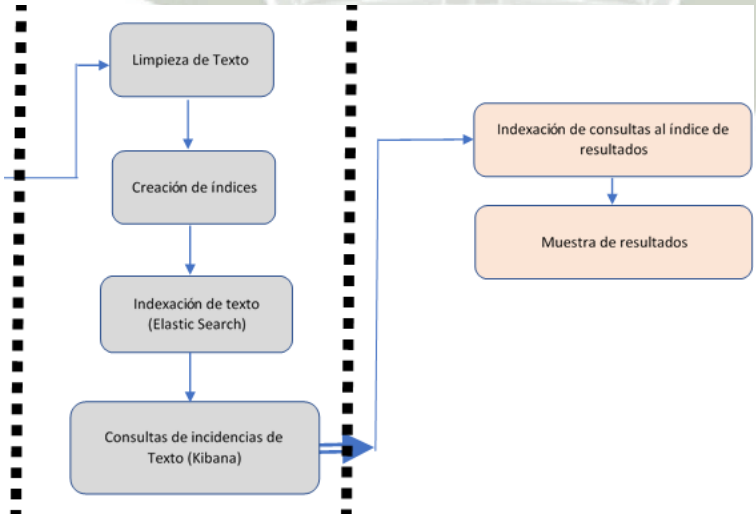


Figura 10. Interacción con las herramientas Elasticsearch y kibana
Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

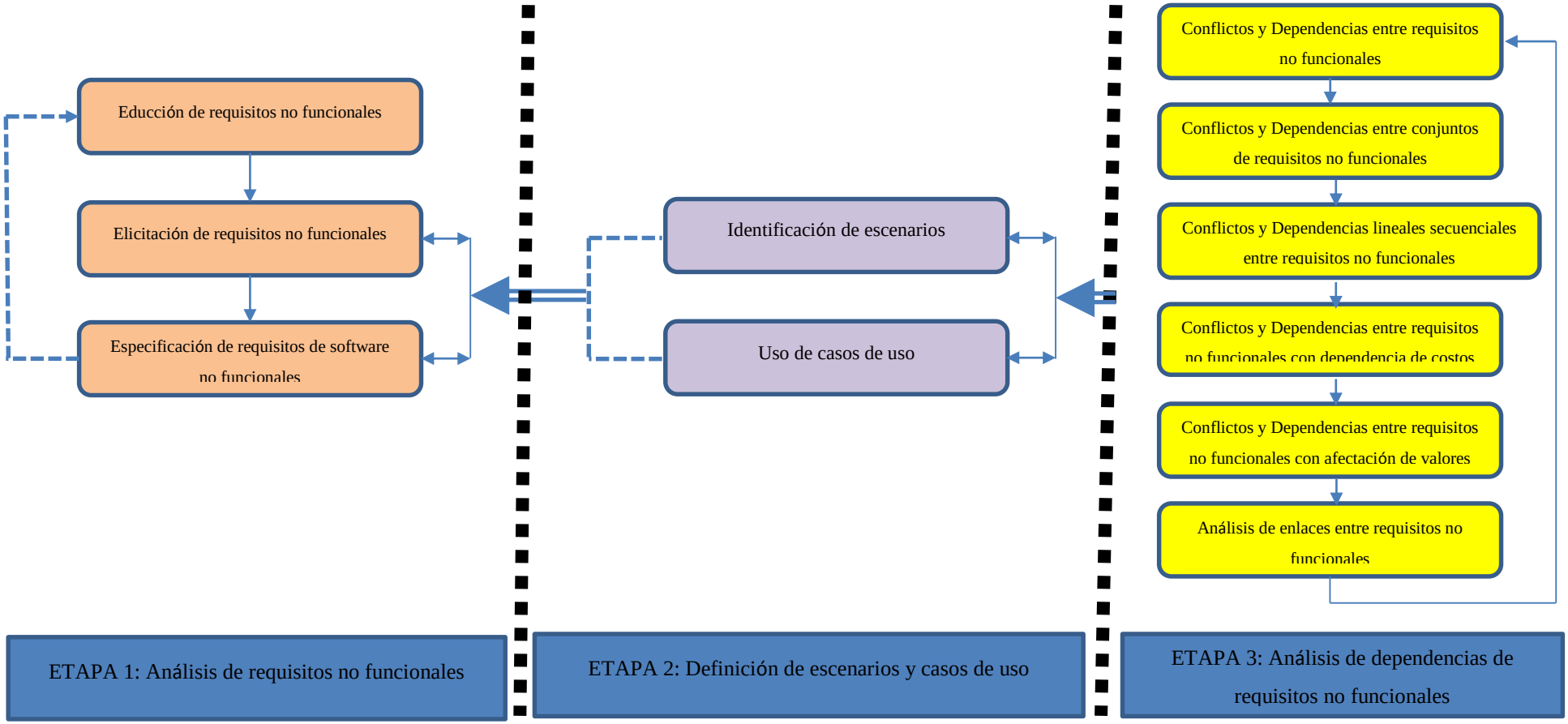


Figura 11. Ciclo de vida del modelo de asociación propuesto
Fuente: Elaboración propia

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

3.2. Propuesta

3.2.1. Etapa de análisis de requisitos no funcionales

Esta etapa de análisis de requisitos no funcionales comprende las fases de educción de requisitos no funcionales, elicitación de requisitos no funcionales y la fase de especificación de requisitos de software no funcionales.

3.2.1.1. Fase de educción de requisitos no funcionales

La obtención de los requisitos no funcionales para esta fase es la siguiente:

- Identificar los orígenes de los requisitos, para lo cual se debe primeramente identificar a aquellas personas que son consideradas importantes en el manejo de este. Esta identificación permitirá entablar una primera asociación de las necesidades del usuario final y su relación con el producto final.
- Elaborar un conjunto de interrogantes, orientadas a la solución que permita obtener y comprender necesidades reales de los usuarios finales.
- Analizar la información reunida para detectar ambigüedades e inconsistencias en los requisitos no funcionales, así como requisitos poco claros.
- Comprobar que el entendimiento del dominio concuerda con los requisitos entregados por los usuarios principales, así como la del modelo del negocio.
- Redactar los requisitos no funcionales de manera correcta. Estos no deben presentar inconsistencias sintácticas ni semánticas.

3.2.1.2. Fase de elicitación de requisitos no funcionales

Los problemas por resolver son los siguientes:

- Problemas de alcance. Cuando los límites del problema se encuentran mal definidas producto de información innecesaria entregada por los clientes o por las personas relacionadas con información importante.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

- Problemas de comprensión. Cuando los clientes no saben expresar sus necesidades provocando inconsistencias en el entendimiento de los límites del problema.
- Problemas de volatilidad. Cuando los requisitos cambian con el tiempo producto de una mala explicación del problema o producto del giro de la organización.

El análisis de asociación debe resolver los siguientes inconvenientes:

- Ejecutar de forma organizada la captura de requisitos no funcionales dentro del dominio.
- Resolver el impacto de los requisitos no funcionales producto de la asociación entre los mismos.

3.2.1.3. Fase de especificación de requisitos de software no funcionales

Los criterios a tomar son los siguientes:

- Buscar la asociación entre el requisito no funcional y la correspondiente codificación del producto.
- Si los niveles de entendimiento y asociación son claros entonces los niveles de codificación no sufrirán consecuencias.
- Si la especificación es inconsistente entonces se logrará un producto mal codificado por lo que se deberá de ingresar a una etapa de refinamiento hasta lograr una definición clara del requisito no funcional.

3.2.2. Etapa de definición de escenarios y casos de uso

3.2.2.1. Fase de identificación de escenarios

Un escenario es una descripción parcial del comportamiento de la aplicación en un momento específico. La utilización de escenarios implica identificar distintas situaciones y describir la acción a llevar a cabo. Los mismos son de gran ayuda en el momento de especificar requisitos; y su rol principal es el de lograr la comunicación entre expertos de software y del dominio, y analizar aspectos específicos de un sistema, describiéndolo en forma concreta. La ventaja de los escenarios sobre cualquier otro método de elicitación de requisitos es

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

que los escenarios guardan una gran similitud a la forma en que los seres humanos entienden y describen los problemas (Gil, 2002).

Los escenarios describen actores, objetivos y episodios. Un actor no necesariamente es una persona o agente físico, un actor representa un rol dentro del sistema, por lo tanto, los actores son las entidades que hacen uso del sistema para satisfacer cierta necesidad, estas necesidades son los objetivos, que representan las condiciones a ser alcanzadas. Los objetivos están representados por episodios. Un episodio es un conjunto de acciones asignadas a determinados actores. Están formados por un conjunto de oraciones en concordancia a un lenguaje natural simple, que hace posible la descripción operacional del comportamiento, las cuales involucran la actividad de alguna función del sistema (Gil, 2002).

El escenario debe constar de componentes que se describen a continuación:

- **Título:** Identifica a un escenario. En el caso de un sub_escenario el título es el mismo que la sentencia que contiene al episodio. Se especifica utilizando la siguiente sintaxis:

Frase | ([Actor|Recurso}+Verbo+Predicado).

- **Objetivo:** Establece la finalidad del escenario describiendo cómo se alcanza ese objetivo. Sintaxis:

[Sujeto] +Verbo + Predicado

- **Contexto:** describe la ubicación física/geográfica y temporal del escenario, así como un estado inicial o precondición del escenario. Sintaxis: Ubicación + Estado, siendo el primero un sustantivo, y el segundo [Actor|Recurso] +Verbo+Predicado+ {Restricción}
- **Recursos:** Identifica las entidades pasivas con las cuales los actores trabajan. Sintaxis: Sustantivo + {Restricción}
- **Actores:** Detalla las entidades que se involucran activamente en el escenario o tienen un rol en este, generalmente puede ser una persona o una organización. Sintaxis: Sustantivo.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

- **Secuencia de Episodios:** Los episodios se presentan como una serie ordenada de sentencias escritas en lenguaje natural. Cada uno de ellos representa una acción realizada por un actor, con la participación de otros actores y la utilización de recursos. Un episodio puede ser expresado como un sub-escenario. Una excepción causa una interrupción en la evolución de un escenario, presentando un conjunto de acciones diferentes que se describe como un caso alternativo o como otro escenario. La posibilidad de registrar excepciones en escenarios ha sido cuestionada, en algunos casos argumentando que las discusiones sobre excepciones podrían distraer a los expertos del dominio, alejándolos de tratar los puntos centrales del sistema, en otros aduciendo que podría verse afectada la "vendibilidad" del sistema. Sin embargo, negar o descuidar importantes excepciones en ingeniería de requisitos, podría resultar en la disconformidad del cliente y altísimos costos a largo plazo. (Gil, 2002).
- **Excepciones**: Los actores y recursos son enumeraciones. El título, objetivo, contexto y excepciones son sentencias declarativas, mientras que los episodios son un conjunto de sentencias expresadas en un lenguaje simple que hace posible una descripción operacional de comportamientos. Un episodio puede concebirse como un escenario en sí mismo, esto posibilita la descomposición de un escenario en subescenarios. (Gil, 2002).

La construcción de escenarios se lleva a cabo de la siguiente manera:

- **Derivar**

  Esta tarea apunta a generar los escenarios candidatos a partir de la información de los requisitos no funcionales. La etapa de *Derivación de escenarios* consiste en tres pasos:

  a. **Identificar actores.** Se identifican las personas relacionadas y asociadas con el requisito no funcional. Deben pertenecer al tipo sujeto. Los actores son clasificados en actores principales y secundarios. Los primeros son aquellos que ejecutan acciones directamente en el dominio de la aplicación, y los segundos son

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

los que reciben y/o dan información, pero no comparten responsabilidades en la acción.

b. **Identificar escenarios.** Se extraen del dominio del problema y de los impactos de los símbolos elegidos como actores principales y secundarios. Cada impacto representa un posible escenario, y es incorporado a la lista de escenarios candidatos. El título del escenario se construye con la acción (verbo) incluido en el impacto, pero expresado en infinitivo más un predicado también tomado del impacto

c. **Crear escenarios.** La intención de esta etapa es construir el escenario aprovechando la información proporcionada por el usuario final, mediante la aplicación de las heurísticas de creación. El producto de esta etapa lo constituyen los escenarios candidatos derivados. El contenido de cada impacto del símbolo de tipo Sujeto que llevó a un escenario candidato es analizado para encontrar símbolos del léxico del tipo Verbo.

- Si el impacto contiene un Verbo:
  - El objetivo se define según el título y la noción del símbolo Verbo, y el punto de vista de la aplicación.
  - Los actores y recursos del escenario son identificados a partir de la información contenida en el símbolo Verbo y deberían ser símbolos de tipo Sujeto y Objeto respectivamente.
  - Los episodios se derivan a partir de cada uno de los impactos del símbolo Verbo.
- Si el impacto no contiene un símbolo Verbo:
  - Los símbolos del léxico contenidos en el impacto son identificados y considerados como posibles fuentes de información.
  - El objetivo se define de acuerdo al título del escenario y el punto de vista de la aplicación.
  - Leyendo la definición completa de los símbolos mencionados, se seleccionan posibles actores y recursos.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Los primeros se derivan de los símbolos de tipo Sujeto y los segundos de los de tipo Objeto.

- Los episodios no se derivan del LEL. Su definición se posterga hasta una etapa posterior.

- **Describir**

  En esta actividad se completan los escenarios candidatos agregando información del requisito no funcional utilizando las heurísticas de descripción y tomando como base el modelo de escenario. El resultado es un conjunto de escenarios candidatos completamente descriptos.

  a. **Completar componentes.** Esta actividad generalmente se basa en entrevistas estructuradas, observaciones y lectura de documentos. La información obtenida permite confirmar y mejorar el curso normal de eventos del escenario. Puesto que algunos escenarios pueden estar parcialmente descriptos en este punto, en esta etapa deberían revisarse las descripciones iniciales. Después de terminar de describir el conjunto de escenarios, los episodios deben ser revisados cuidadosamente para confirmar el orden secuencial o detectar paralelismos, y para encontrar episodios opcionales.

  b. **Crear subescenarios.** Los subescenarios son una posible solución al problema de la explosión de escenarios señalada en [Cockburn95]. Se utilizan cuando:

     - Se detecta comportamiento común en varios escenarios,
     - Aparecen cursos de acción condicionales o alternativos complejos en un escenario, o
     - Se detecta en un escenario la necesidad de mejorar una situación con un objetivo concreto y preciso.

  c. **Completar restricciones.** Las restricciones se utilizan para caracterizar requisitos no funcionales aplicados a Contexto, Recursos y Episodios. Algunas pueden ser elicitadas desde la unidad del discurso y otras pueden surgir examinando los episodios.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

d. Completar excepciones. Finalmente, se deben detectar los casos alternativos y excepciones. Algunas causas de excepción son elicitadas desde las fuentes de información mientras que otras pueden deducirse analizando los episodios y la no disponibilidad o malfuncionamiento de los recursos. Cuando se descubren las causas de una excepción, los ingenieros de requisitos deberían investigar cómo es tratada la excepción en el UdeD, surgiendo así una nueva situación que podría necesitar ser descripta a través de un escenario separado.

- **Organizar**

  Esta actividad es la encargada de detectar las siguientes debilidades en los requisitos no funcionales:

  – Falta de homogeneidad,
  – Problemas semánticos menores, y
  – Falta de perspectiva global

  Estos problemas se resuelven realizando las siguientes tareas:

  a. **Reorganizar.** Los escenarios pueden tener diferente grado de detalle o solapamiento de información, especialmente cuando son construidos por más de un ingeniero de requisitos. Esta situación se complica más a medida que se manejan más escenarios y el equipo de trabajo es más numeroso. Entonces, las heurísticas de reorganización, aplicadas a escenarios, tienen por objeto obtener escenarios consistentes y homogéneos. Básicamente la reorganización consiste en juntar dos o más escenarios en uno, o dividir un escenario en uno o más. La primera situación se da cuando un único escenario ha sido artificialmente dividido durante las etapas Derivar o Describir. La necesidad de partir el escenario surge cuando éste contiene más de una situación. Para establecer la necesidad de operaciones de composición/descomposición, algunas propiedades y relaciones entre escenarios deberían ser previamente identificadas.

  b. Definir relaciones entre escenarios. En este caso, el objetivo de las relaciones a definir es la construcción de escenarios

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

integradores, así que se necesita un nuevo conjunto de relaciones. En esta etapa se identifican diferentes relaciones entre escenarios con el fin de integrarlos.

c. **Integrar.** Esta actividad implementa el comportamiento "middle out" del proceso de construcción de escenarios [Hadad99], siendo los tres primeros pasos bottom-up y los últimos dos top-down. La integración de escenarios como tal podría ser vista como un procedimiento de cinco pasos:
   - Construir jerarquías de escenarios
   - Detectar orden parcial entre jerarquías
   - Construir secuencia de jerarquías
   - Construir el esqueleto de la integración
   - Proponer Título, Objetivo y Contexto para los escenarios integradores

- **Verificar**

  Esta actividad es llevada a cabo al menos dos veces durante el proceso de construcción de escenarios; la primera sobre el conjunto de escenarios completamente descriptos y la segunda luego de la actividad *Organizar*. Esta etapa se realiza siguiendo un checklist con heurísticas de verificación.

- **Validar**

  Los escenarios son validados con los clientes/usuarios efectuando entrevistas estructuradas o reuniones. Durante la validación, debe prestarse especial atención al componente Dudas para aquellos escenarios que lo incluyen.

  Es importante destacar que, aunque se use una descripción estructurada, los escenarios son escritos en lenguaje natural, empleando el propio vocabulario de los clientes/usuarios y describiendo una situación específica bien delimitada.

3.2.2.2. Fase de uso de casos de uso

Para emplear los casos de uso, estos deben ser tratados de tal forma que se deban de eliminar los siguientes defectos:

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

- Ambigüedades surgidas por la aplicación de la descomposición

  Cuando se pone mucho esfuerzo en la estructuración de los casos de uso, los proyectos de desarrollo de software empiezan a tener muchas dificultades. El Lenguaje de Modelado Unificado utiliza un par de relaciones que son empleados entre los casos de uso: *include* y *extend*. Estos pueden ser muy útiles, pero causan problemas con mucha frecuencia, sobre todo la relación *include*. En tales casos, los analistas toman un caso de uso bastante general y lo dividen en sub-casos de uso. Cada sub-caso de uso se puede desglosar, por lo general, hasta llegar a algún tipo de caso de uso elemental, que parece atómica en algún grado.

  Esta descomposición funcional es, un estilo de diseño que es la antítesis del desarrollo orientado a objetos. Esto conduce a dos tipos de problemas:

  1. Cuando la estructura de los casos de uso se ve reflejada directamente en el código, de modo que el diseño de los casos de uso del sistema se parece a los casos de uso del usuario. Un síntoma común es encontrar al comportamiento de los objetos manipulando datos: esto es poco más que una estructura de datos encapsulada.

     Este tipo de diseño ayuda a perder la mayor parte de los beneficios y características de los objetos. El sistema duplica el comportamiento a través de los distintos controladores, y el conocimiento de esta estructura de datos también es conocido por los demás controladores. La esencia del problema es que la descomposición funcional nos anima a pensar en el contexto de un comportamiento de alto nivel. Hecho de esa manera, es difícil utilizar ese mismo comportamiento en otro contexto, aunque sea casi idéntica. La descomposición funcional no promueve ese enfoque. Se puede evitar este problema, recordando que la estructura interna del sistema no tiene que parecerse a la estructura externa del mismo (los casos de uso). Sin embargo, este es un concepto difícil para los desarrolladores sin

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

experiencia ya que son los más propensos a crear una descomposición funcional.

2. Si no se encuentra que la descomposición funcional afecta a su diseño interior, un enorme conjunto estructurado de casos de uso tiene problema debido, fundamentalmente, a que la gente termina invirtiendo mucho tiempo en ellas. Llevar a todos los casos de uso hasta los pasos elementales implica que los argumentos sobre los casos de uso que caben en los casos de uso de mayor nivel, consumen tiempo que podría ser empleado en otras cosas. La captura de todos los detalles de los casos de uso no es necesaria en las primeras fases del desarrollo. Se necesita primero buscar en las zonas de alto riesgo, pero muchos de los detalles se pueden dejar para las últimas etapas del desarrollo iterativo. Esa es la filosofía del desarrollo iterativo. Esto causa desaliento al emplear la relación include. En la práctica se advierte similitudes entre el diseño de los casos de uso y los casos de uso del usuario, y estos son señales de advertencia. La relación *extend* causa menos problemas, aunque los investigadores poco dicen de sus deficiencias.

- Ambigüedades surgidas por el abuso de la abstracción

  Los objetos nacen de la abstracción. Un buen diseño es aquel que se basa en las abstracciones simples, haciendo un problema complejo manejable. Así como los diseñadores utilizan la abstracción, otros integrantes del equipo de desarrollo de software también la pueden usar.

  Pero la abstracción puede llevar a problemas en los casos de uso. Comúnmente los usuarios entienden los casos de uso, al principio, y posteriormente se sienten perdidos con los casos de uso más abstractos. Uno de los principales propósitos de los casos de uso consiste en comunicarse con los usuarios -los clientes- del sistema. La abstracción de los casos de uso, más allá del nivel de comprensión, no ayuda a nadie. La falta de abstracción en los casos

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

de uso no es perjudicial, ya que la estructura interna no tiene que ser la misma que la estructura externa.

La abstracción de los casos de uso puede conducirnos a casos de uso más grandes, que, en el desarrollo iterativo, son más difíciles de planificar. Se puede pasar mucho tiempo discutiendo acerca de la abstracción en vez de usar el tiempo en otras cosas.

- Ambigüedades surgidas por el uso de las interfaces gráficas del usuario

  Con todas las herramientas para diseñar interfaces gráficas de usuario que existen en estos días, muchos desarrolladores las están usando para ayudar a determinar los casos de uso. Esta lógica es atractiva. Una interfaz gráfica de usuario es concreta a un usuario, nos ayuda a no olvidar detalles; da al usuario una sensación de cómo se verá el sistema. Las interfaces gráficas de usuario son fáciles de construir su prototipo ya que hacen demostraciones razonables para explicar la capacidad del futuro sistema.

  Existe un problema fundamental. Al mostrar un prototipo de interfaz gráfica de usuario a un usuario, parece que casi todo lo hace, y que lo único que queda son pocos detalles detrás de las escenas. Sabemos que lo que se esconde detrás de las escenas es la parte más complicada del ejercicio, pero esto es muy difícil de entender para los clientes. Las interfaces gráficas de usuario conducen a una falsa indicación de los avances y dificultades. Hay una enorme diferencia entre el esfuerzo real y el esfuerzo percibido, por lo que es difícil hacer la negociación que es tan importante en el control del alcance.

  A pesar del hecho de que las herramientas para interfaces gráficas de usuario parecen tan fáciles de usar, siempre existen ajustes para que se vea perfecto. Este ajuste fino establece expectativas de la gente en una determinada dirección, haciendo que la gente se vuelva reacia a realizar cambios.

- Ambigüedades surgidas por el uso de plantillas de otros proyectos de desarrollo de software

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

En muchas ocasiones insertamos, en el proyecto actual, plantillas de casos de uso de otros proyectos llevados a cabo. El problema con estos casos de uso es que no abordan directamente las necesidades de un usuario. Las necesidades reales del usuario son algo así como garantizar un formato consistente dentro de un documento.

El problema con ir directamente al caso de uso del sistema es que se le niega la oportunidad de llegar a los casos de uso de otro sistema que se ocupe de lo mismo. No se puede utilizar los casos de uso del usuario, debido a que no encajan bien en el proceso de programación iterativo.

En un principio se pone atención a los casos de uso del sistema, ya que son los más útiles para la planificación de la iteración y pruebas del sistema. Sin embargo, con todos los casos el uso del sistema existe otro caso de uso de usuario que lo respalda.

- Ambigüedades surgidas por el mal diseño de los casos de uso

  Cuando se planifica la construcción de un sistema automatizado se prevee que los artefactos generados en las etapas iniciales sean adecuados para la etapa de diseño. El defecto de los casos de uso, para esta etapa, implica que no existen los mecanismos adecuados para lograr determinar los comportamientos generales o especializados haciendo que el actor, relacionado con el caso de uso, viaje a través de la generalización y la especialización del caso de uso correspondiente. Esto conlleva a que no se logre visualizar con claridad con que tipo de caso de uso se está trabajando.

- Ambigüedades surgidas por la tecnología de objetos

  Los casos de uso no son elementos que están orientados a objetos. Cada caso de uso captura una abstracción funcional importante que puede causar numerosos problemas con la descomposición funcional que la tecnología de objetos espera evitar. Esto implica que el modelo de casos de uso y el modelo de objetos pertenecen a diferentes paradigmas (es decir, funcional y orientada a objetos) y por lo tanto utilizan diferentes conceptos como en la terminología, las técnicas y notaciones. La estructura simple del modelo de casos

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

de uso permite no definir claramente la estructura de la red del modelo de objetos con sus objetos y las clases de colaboración. Estas asignaciones son informales y un tanto arbitrarias, sirven de poco para lograr identificar objetos, clases, y sus interacciones.

- Ambigüedades surgidas por las iteraciones

  Otro problema potencial con el modelado de casos de uso es no saber cuándo parar. Cuando uno está construyendo una aplicación no trivial, a menudo hay un gran número de casos de uso que pueden producir un número esencialmente infinito de escenarios de uso, especialmente con interfaces de usuario gráficas.

- Ambigüedades surgidas por la arquitectura

  Otro problema importante corresponde a la arquitectura del sistema que puede dar lugar a una mala interpretación de los casos de uso, debido a la mala interpretación que se lleva a cabo de la arquitectura del software. Dichas arquitecturas, típicamente, exhiben encapsulación pobre y de acoplamiento excesivo, y una inadecuada distribución de la inteligencia de la demanda entre las clases.

- Ambigüedades surgidas por los escenarios

  Esta ambigüedad surge por la poca claridad que existe entre el concepto de caso de uso y escenario.

- Ambigüedades surgidas por la mala interpretación de los dominios del sistema

  Este problema ocurre cuando se desconocen los dominios del sistema real a construir o cuando no se tiene muy en claro las áreas o módulos que contiene el mencionado sistema. Generalmente los analistas tratan de representar, al mismo tiempo, a los usuarios del sistema y a los usuarios del negocio.

- Ambigüedades surgidas por la educción de requerimientos

  Esta es la primera etapa de la ingeniería de requerimientos. En ella se toma contacto con el cliente para empezar a entender el problema que se tiene que resolver. Generalmente los clientes no entienden de computación o informática y proporcionan sus ideas

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

de manera desordenada pensando que el ingeniero de sistemas puede entenderlas a cabalidad.

Esta percepción nos lleva a una mala interpretación de los casos de uso ya que, en la generalidad, los ingenieros de sistemas empiezan a esbozar los primeros casos de uso sobre problemas técnicos y no sobre la problemática de los procesos; y en muchos casos confundiendo los objetivos de la construcción del software con la definición de los casos de uso.

Al establecer las cualidades para un sistema, es importante identificar todas las categorías de usuarios (incluyendo de otros sistemas) que interactuaran con el sistema, y entender que atributos de calidad se debe tener en cuenta. Un atributo de calidad como el desempeño puede surgir para un usuario como un interés, y para otro como un valor, por eso es útil una educción directa de requisitos para ambos, es decir interés y valor para cualquier (grupo) de usuario(s). Es importante dirigir a crear "justo lo que quieren los usuarios", con las cualidades que ellos toman en cuenta; los rasgos de baja prioridad o cualidades solo incrementan la complejidad para la organización de desarrollo y /o el usuario.

El equipo de requerimientos debería, sin embargo, estar alerta a requerimientos que el usuario presupone o que no estén disponibles para articularse directamente. Entender el objetivo de los usuarios y forzar a que impacte su suceso y sentido de utilidad, ayudará a surgir y establecer las prioridades de cualidades del sistema, así como la funcionalidad. En adición a descubrir que cualidades son importantes al usuario en el nivel de sistema, las cualidades asociadas a una función particular u objetivo de usuario debe ser educido. Las cualidades pueden necesitar ser trasladadas por los desarrolladores desde los objetivos de nivel usuario, valores e intereses, en requerimientos específicos de calidad técnica. Por ejemplo, un requerimiento de usuario no puede ser degradado o impedido por el rendimiento lento del sistema, al realizar una tarea que puede ser trasladada en requerimientos sobre el tiempo de transacciones y la potencia de red.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

- Ambigüedades surgidas por la elicitación de requerimientos

  Esta es la segunda etapa de la ingeniería de requerimientos. El ingeniero de sistemas tiene en mente toda la problemática del problema a resolver; incluso se arriesga a representar el problema con los primeros casos de uso. Después de varias iteraciones concibe el documento de elicitación de requerimientos. A partir de este artefacto se diseñan los casos de uso posteriores.

  Las fallas de redacción, ortografía, sintaxis y semántica conllevan a una mala interpretación de los casos de uso. El principal resultado del proceso de elicitación de requerimientos es la serie de requerimientos que deberán ser utilizados por el equipo de desarrollo de software para crear un producto correcto y de manera apropiada. Además de este resultado principal, el proceso ofrece una serie de salidas intangibles.

  Cuando existe un buen proceso de elicitación de requerimientos, se ayuda a los usuarios a entender qué es lo que quieren, qué es lo que necesitan, cuáles son las restricciones y alternativas, ventajas y desventajas de cada una. Desde el punto de vista de los ingenieros y desarrolladores, la existencia de un buen proceso de elicitación de requerimientos ayuda a los mismos a resolver el problema correcto, es decir, lo que realmente desea y necesita el cliente, a resolver un problema factible, a ganar la confianza del cliente y su cooperación y a ganar conocimiento sobre el dominio del problema. (Tuffley, 2005)

  Cuando el proceso de elicitación es pobre es posible que se dé respuesta a un problema equivocado, los usuarios podrían expresar su descontento ya que los desarrolladores no los escuchan y se produciría un desarrollo caótico en el sentido que se pierde información, se toman decisiones incorrectas, los costos y el cronograma se salen del plan y se pierde dinero. La alternativa de producto de software que se desarrolla podría afectar el proceso de elicitación ya que el contacto con el usuario y el conocimiento del mismo surgen desde medios diferentes según se desarrolle software a medida o empaquetado. Los usuarios para una u otra alternativa

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

de software tienen diferentes expectativas y, muy probablemente, diferente percepción del proceso de elicitación de requerimientos. (Tuffley, 2005)

3.2.3. Etapa de análisis de conflictos y dependencias para requisitos no funcionales

Esta etapa se lleva a cabo por el uso de la herramienta Elasticsearch de los escenarios y los casos de uso. Por medio de esto se pueden deducir los conflictos y dependencias entre requisitos no funcionales. Pueden existir otras técnicas para resolver este problema como por ejemplo compilador de textos que implican hacer el análisis léxico, sintáctico y semántico de los escenarios y casos de uso y por medio de la automatización deducir las dependencias entre requisitos no funcionales, pero esto conlleva en construir una herramienta que implica otro trabajo de investigación.

3.2.3.1. Fase de conflictos y dependencias entre requisitos no funcionales

Esta fase implica buscar requisitos dependientes con respecto a su funcionalidad. Esto implica buscar requisitos que no puedan ejecutarse hasta no tener respuesta del requisito origen lo que significa que un requisito necesita de la funcionalidad o respuesta de otro y queda estático hasta no encontrar la solución del requisito origen.

3.2.3.2. Fase de conflictos y dependencias entre conjuntos de requisitos no funcionales

Esto se presenta cuando la funcionalidad de un requisito no funcional depende de un par de requisitos no funcionales. La solución de este problema se encuentra en hallar solución a los requisitos no funcionales independientes.

3.2.3.3. Fase de conflictos y dependencias lineales secuenciales entre requisitos no funcionales

Este esquema muestra cuando existen un par de requisitos no funcionales en la cual un requisito no funcional es secuencial con respecto al otro requisito no funcional otorgando la figura de secuencialidad en la ejecución de requisitos no funcionales. Esta

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

linealidad puede implicar la transmisión de efectos colaterales dentro de los requisitos.

3.2.3.4. Fase de conflictos y dependencias entre requisitos no funcionales con involucramientos de costos

Esta fase se encuentra relacionada con los costos que se encuentran involucrados los requisitos no funcionales. Esto significa que el costo en el que se encuentra involucrado un requisito no funcional afecta en el costo de otro requisito no funcional. El costo involucra tiempo, uso de recursos computacionales y eficiencia del producto a construir.

3.2.3.5. Fase de conflictos y dependencias entre requisitos no funcionales con afectación de valores

Esta fase se encuentra relacionada con los valores que se encuentran involucrados o asociados con los requisitos no funcionales. Esto significa que el valor que devuelve un requisito no funcional afecta el valor del otro requisito no funcional. El valor involucra datos.

3.2.3.6. Fase de análisis de enlaces entre requisitos no funcionales

Esta fase implica que, si existe un par de requisitos no funcionales, la funcionalidad de cualquiera de ellas resuelve la funcionalidad de cualquier otro. Esto significa que basta tener el resultado de la funcionalidad de uno de ellos para poder obtener la funcionalidad de algún otro requisito no funcional.

3.3. Comparación con otras técnicas o métodos

El modelo de asociación propuesto, comparado con otras técnicas resulta ser más versátil puesto que la ayuda de los escenarios y casos de uso proporcionan la flexibilidad del caso. Si bien es cierto que como presenta el tiempo de construcción o elaboración, el resultado proporciona respuestas adecuadas de tal forma que se obtengan resultados bastantes rápidos.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Las técnicas específicas para los requisitos no funcionales son las mismas que aquellas que permiten obtener requisitos funcionales. La Tabla 4 muestra la comparación con otras técnicas.

Tabla 4. Comparación del modelo de asociación con otras técnicas

TECNICA	VENTAJAS	DESVENTAJAS
Definición de diagramas	Emplea artefactos de UML (diagramas de secuencias, estados y casos de uso). Cuando varias personas necesiten conocer particularidades del diseño. Cuando se necesita exponer alguna porción de código en particular.	Influenciado por el proceso. El diagrama permite que otra persona realice la codificación. No se usan todos los artefactos propuestos.
Definición de casos de uso	Identificación clara de los actores. Propuesta clara de los intereses de los actores. Identificación de eventos. Identificación de escenarios.	Degenerados por la aplicación de la descomposición. Degenerados por el abuso de la abstracción. Degenerados por los interfaces gráficos de usuario. Degenerados por el uso de plantillas de otros proyectos. Degenerados por el mal diseño de los casos de uso. Degenerados por la Tecnología Orientada a Objetos. Degenerados por las iteraciones.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

		Degenerados por la arquitectura. Degenerado por los escenarios. Degenerados por la mala interpretación del dominio. Degenerados por la mala toma de requerimientos.
Especificación de casos de uso	Relatos secuenciales. Descripción clara de los eventos. Evita terminología vaga. Evita las ambigüedades textuales de los casos de uso. Permite definir las pre y post condiciones.	No permite describir las necesidades del usuario. No cubren los requisitos en su totalidad. Existe independencia en los requisitos del dominio y del sistema.
Prototipos	Se puede usar en las diferentes etapas del desarrollo de software. Son rápidos de construir. Evolucionan por medio de las iteraciones. Presentan bajo costo para el desarrollo.	No tiene equivalencias con el producto final. No lleva a cabo la totalidad de la funcionalidad del sistema. Es una aplicación que comúnmente no funciona.
Definición de criterios de aceptación	Satisfacen funcionalidades específicas. Proporcionan aceptaciones realistas. Necesita del documento de requisitos concluido. Los requisitos son medibles. Los requisitos no son contradecibles.	Las ambigüedades afectan la funcionalidad. Una mala escritura afecta al requisito. Si el requisito no es probado entonces no posee la calidad suficiente.
Modelo de asociación	Elimina las ambigüedades de los casos de uso. Emplea las propiedades de los casos de uso. Captura, en esencia, todas las características de los escenarios.	Es empleado en la primera etapa del desarrollo del sistema.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

	Emplea niveles de profundidad en los escenarios. Satisfacen funcionalidades específicas. Evita la terminología vaga. Identificación clara de los actores y sus roles. Presenta evolución iterativa. Los requisitos no son contradecibles. Las pre y post condiciones se encuentran relacionadas, en forma directa, con las escenas.	Los requisitos no son medibles. Se pierde tiempo en su elaboración.

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

CAPÍTULO 4: CASO DE ESTUDIO

4.1. Introducción

Existen razones justificadas para sugerir trabajar con un caso de estudios. Estas son las siguientes (Yin, 2013):

- El enfoque consiste en resolver el cómo se determinan las dependencias entre requisitos no funcionales, los mismos que pueden influir en los requisitos funcionales.
- Esto se presenta bajo un enfoque contemporáneo lo que significa que sucede en un periodo de tiempo en la construcción del producto de software.
- Se cumple con el siguiente esquema de investigación:
  a) Interrogantes

     ¿Cómo el modelo de asociación identifica los conflictos en los requisitos no funcionales?

     ¿Por qué los conflictos identificados en los requisitos no funcionales son más explícitos empleando escenarios y casos de uso?

  b) Proposiciones
     - El modelo de asociación identifica los conflictos entre requisitos no funcionales por medio de escenarios.
     - El modelo de asociación explica los conflictos entre requisitos no funcionales por medio de casos de uso.

  c) Unidades de análisis
     - Requisitos no funcionales
     - Requisitos expresados en escenarios
     - Requisitos explicados en casos de uso

  d) Encadenamiento lógico de la data a las proposiciones
     - El encadenamiento se lleva a cabo por medio del modelo de asociación e identificación propuesto.

  e) Criterios para interpretar lo hallado
     - Descripción de los escenarios
     - Descripción de los casos de uso

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Para los desarrolladores de software pocas son las herramientas que hablan sobre la especificación de los requerimientos de software, pero si hablan sobre un producto de trazabilidad, el usuario no entiende como se realiza este proceso pues al no contar con las etapas de educción y especificación de requerimientos, no pueden relacionar las actividades del cliente, analista y desarrollador. Lo que se busca es una herramienta completa con estos tres puntos para una primera etapa de desarrollo.

Para el proceso de educción de requerimientos se hizo una entrevista con el cliente, quien entregó los primeros alcances sobre el software que se desea desarrollar, esta entrevista despeja varias dudas y ayuda a tener una perspectiva más clara acerca del proyecto. A continuación, se presentan los puntos más relevantes de la entrevista:

- Dentro del desarrollo de algún requerimiento, el usuario deberá ser único para evitar los errores que puedan ocurrir a la hora de hacer la gestión de cambios de algún requerimiento.
- El sistema debe generar tres documentos indispensables, un documento para la educción, elicitación y especificación de requerimientos, es decir los artefactos para el análisis del proyecto desarrollado.
- El sistema debe poder generar la exportación de estos artefactos en archivos con extensiones .doc y .pdf.
- El sistema de gestión de requerimientos recibirá el nombre de REMAN, además está orientado a analistas o desarrolladores, es decir personas conocedoras de la gestión de requerimientos.
- El proyecto se desarrolla para una versión en escritorio, la herramienta tiene una estimación de dos años para su construcción total.
- El Stakeholder proporciona plantillas, las mismas que se deben completar con las faltantes (requerimientos no funcionales).
- Una educción tiene una o varias elicitaciones de requerimientos.
- Una elicitación tiene una o varias especificaciones de requerimientos de software.
- La especificación es el aspecto resumido o detallado de los requerimientos que van a ser entregados a los desarrolladores.
- En las plantillas de actores y organización se debe especificar ambos lados, tanto de la organización que requiere el desarrollo del sistema y de la empresa desarrolladora.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

- En la organización solicitante se deben de especificar los cargos para una buena identificación de actores.
- Un requisito puede ser propuesto por los siguientes actores: Arquitecto, analista, desarrollador, stakeholder. (en esencia los requerimientos no funcionales, son lo que se deben precisar).
- En los comentarios van aquellos detalles que servirán para la construcción o resolución de algún problema.
- El documento histórico registrará versión, fecha y la razón del cambio para luego en la trazabilidad detectar lo bueno y lo malo o lo perdido.
- La exportación se lleva a cabo de las últimas versiones generadas.
- Futuros trabajos serán: Cálculo de métricas, trazabilidad, desarrollo de diagramas, entre otros.
- Los requerimientos funcionales nos llevan a la construcción del producto.
- Los requerimientos no funcionales nos llevan a medir la calidad del producto.
- Los requerimientos funcionales salen de las plantillas proporcionadas por el stakeholder.
- La entrevista debe ser considerada en la parte administrativa.
- Para la parte ADMINISTRATIVA del proyecto se debe dejar estructurado un libro que lleve la administración del proyecto, pero se construirá en una segunda versión.
- Se define un libro de DATOS GENERALES donde se contienen a los actores, la organización solicitante, organización proveedora, entre otros datos.
- El producto contiene un botón de reportes y su relación con el historial, donde se podrá escoger la elicitación de requerimientos.
- Para el presente producto queda suspendida el artefacto de trazabilidad, pero debe quedar estructurado para una segunda versión.
- Para un futuro próximo, la herramienta debe permitir generar el documento completo de análisis.
- Para los Requerimientos No Funcionales la plantilla propuesta generalmente consistirá en una plantilla de trabajo salvo opinión del desarrollador.
- Las salidas del producto son artefactos, los mismos que son independientes.
- Visualmente se agrega tabla por tabla y en el reporte debe se deben de especificar las últimas versiones.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

- Lo que se supone que falta en el producto se definirá bajo los términos “POR CONSTRUIR”.
- La herramienta REM no tiene educción y especificación de requerimientos, en REM el proceso de elicitación lo desarrollan bajo el concepto de casos de uso encontrándose muy bien desarrollado.

El objetivo de este documento es explicar de forma detallada cada una de las funcionalidades de las que consta el Software REMAN para la gestión y/o manejo de requerimientos.

La gestión de requerimientos es un aspecto amplio y muy importante a la hora de crear software de calidad, y es uno de los puntos clave del éxito de un proyecto, es por eso que tenemos conocimiento y sabemos la importancia de gestionar requerimientos y usar software adecuados para mantener un orden y una armonía en el completo desarrollo de los proyectos.

Actualmente en el mercado existen múltiples softwares para la gestión de requerimientos, pero muchos de ellos no nos brindan un producto completo para dicha finalidad, y en algunos casos solo están disponibles por cierto tiempo de manera gratuita.

REMAN es un software gratuito y de código abierto, desarrollado para resolver el problema de la administración de requerimientos y pensando en la importancia al realizar software de calidad y que cumpla con todos los estándares para poder salir al mercado y ser un producto competitivo y exitoso.

REMAN consta de varios módulos. Estos se denominan: EDUCCIÓN, ELICITACIÓN, ESPECIFICACIÓN, ORGANIZACIÓN y REQUISITOS NO FUNCIONALES; cada uno de ellos tiene una característica vital a la hora de comenzar con un proyecto de software, y necesitan en algunos casos pre-requisitos para poder desarrollar un libro posterior y mantener la armonía para poder construir un proyecto en general.

Cada uno de los libros y funcionalidades que REMAN posee, son mencionadas detalladamente en el manual de usuario para brindar una ayuda general y en algunos casos detallados del uso adecuado de REMAN.

En el anexo 1, 2 y 3 se muestran los requerimientos funcionales en las fases de educción, elicitación y especificación.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

4.2. Especificación del producto

4.2.1. Etapa de análisis de los requisitos no funcionales

Esta etapa consiste en analizar la problemática del producto a construir y lograr asociar los requerimientos no funcionales con respecto a los requerimientos funcionales. Además de ello, también se asocia, en cascada la educción, elicitación y especificación de requerimientos de software permitiendo una trazabilidad para requerimientos no funcionales.

Las herramientas convencionales permiten la trazabilidad de requerimientos funcionales, pero no lo hace con respecto a los requerimientos no funcionales. Es por eso que los requerimientos no funcionales en la etapa de elicitación deben referenciar a uno o más requerimientos en la etapa de educción; lo mismo sucede con la especificación de requerimientos de software quien debe contener uno o más requerimientos en la etapa de elicitación. De esta manera se asocian los requerimientos en esta etapa obteniendo como resultado una primera trazabilidad de requerimientos no funcionales.

4.2.1.1. Educción de requisitos no funcionales

Para implementar los requisitos no funcionales se tuvo que llevar a cabo consultas al cliente y a los principales stakeholders del producto, y estuvieron orientados a los principales atributos de calidad. Para ello se emplearon los siguientes casos de uso esenciales:

Tabla 5. RNF: Facilidad de uso

RNF-0001	Facilidad de Uso
Versión	0.1
Requisitos asociados	Ninguno
Descripción	Un usuario del sistema totalmente capacitado no debe tener más de 2 errores. Se necesita un sistema amigable.
Importancia	Vital
Comentarios	Ninguno

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Tabla 6. RNF: Facilidad de mantenimiento

RNF-0002	Fácil mantenimiento
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá contar con manuales técnicos y de usuario los cuales permitan el mantenimiento, con respecto a los posibles errores que se puedan presentar durante la operación del sistema o con nuevas funcionalidades que se desee incorporar.
Importancia	Importante
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 7. RNF: Escalabilidad del sistema

RNF-003	Escalabilidad del sistema
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá estar en capacidad de permitir en el futuro el desarrollo de nuevas funcionalidades, modificar o eliminar funcionalidades después de su construcción y puesta en marcha inicial.
Importancia	Vital
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 8. RNF: Flexibilidad del sistema

RNF-004	Flexibilidad del sistema
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá estar en capacidad de permitir en el futuro la modificación de módulos de acuerdo a nuevos requerimientos o que se requieran eliminar, sin que sufra cambios drásticos en su funcionamiento.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Importancia	Vital
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 9. RNF: Número mínimo de usuarios concurrentes

RNF-005	Número mínimo de usuarios concurrentes
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá permitir el acceso simultáneo y concurrente de 100 usuarios
Importancia	Vital
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 10. RNF: Tiempo límite de espera para respuestas del sistema

RNF-006	Tiempo límite de espera para respuestas del sistema
Versión	0.1
Requisitos asociados	RNF-008 Tiempo límite de espera para casos excepcionales
Descripción	El tiempo límite de espera de respuesta del sistema es de 3 segundos para el 90% de servicios del sistema.
Importancia	Importante
Comentarios	Las excepciones se dan en caso de que la solicitud de datos al sistema sea mayor a la usual, por ejemplo la generación de reportes o consultas por los usuarios.

Fuente: Elaboración propia

Tabla 11. RNF: Tiempo límite de espera para casos excepcionales

RNF-007	Tiempo límite de espera para casos excepcionales
Versión	0.1
Requisitos asociados	Ninguno

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Descripción	Este tiempo será de 10 segundos, y se da en alta carga de tráfico en la red de comunicación usada.
Importancia	Vital
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 12. RNF: Autorización para el uso del sistema

RNF-008	Autenticación para el uso del sistema
Versión	0.1
Requisitos asociados	Todos los Requerimientos Funcionales
Descripción	Los usuarios deben identificarse antes de ingresar al sistema. Cada usuario contará con una entrada y una contraseña.
Importancia	Alta
Comentarios	El administrador del sistema es la persona encargada de asignar cuentas a los usuarios.

Fuente: Elaboración propia

Tabla 13. Número máximo de autentificaciones fallidas

RNF-009	Número máximo de autentificaciones fallidas
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema bloqueara al usuario con 4 intentos fallidos de autorizaciones consecutivas.
Importancia	Normal
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 14. RNF: Canal seguro de autorización

RNF-010	Canal seguro de autorización

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá contar con un nivel de seguridad alta (encriptación de contraseña con letras números y caracteres como “ |!$%&/”)
Importancia	Normal
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 15. RNF: Lenguaje

RNF-011	Lenguaje
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá ser implementado en el Lenguaje PHP.
Importancia	Importante
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 16. RNF: Base de datos

RNF-012	Base de Datos
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema utilizará la base de datos origen, la cual fue desarrollada en FoxPro.
Importancia	Importante
Comentarios	Ninguno

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Tabla 17. RNF: Interfaz

RNF-013	Interfaz
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema se implementará utilizando una interface en PHP.
Importancia	Importante
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 18. RNF: Arquitectura de computadores

RNF-014	Arquitectura del Computador
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá funcionar tanto en la arquitectura de X86 como X32
Importancia	Importante
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 19. RNF: Autorización para el uso del sistema

RNF-015	Autorización para el uso del sistema
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá funcionar previa autorización de la administración
Importancia	Importante
Comentarios	Ninguno

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

4.2.1.2. Elicitación de requisitos no funcionales

Tabla 20. RNF - Elicitación: Facilidad de uso

RNF-0001	Facilidad de Uso
Versión	0.1
Requisitos asociados	Ninguno
Descripción	Un usuario del sistema totalmente capacitado no debe tener más de 2 errores. Se necesita un sistema amigable. La solución se orienta al empleo de interfaces de usuario aprobados por el usuario final.
Importancia	Vital
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 21. RNF - Elicitación: Fácil mantenimiento

RNF-0002	Fácil mantenimiento
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá contar con manuales técnicos y de usuario los cuales permitan el mantenimiento, con respecto a los posibles errores que se puedan presentar durante la operación del sistema o con nuevas funcionalidades que se desee incorporar. En los manuales se debe de incluir índices temáticos, índices de figuras e índices de esquemas, así como figuras que apunten a un problema en específico. Asimismo, en los referidos artefactos se deben de incluir fechas, hora y contacto del ingeniero que llevó a cabo el mantenimiento y el futuro gestión del cambio. Se debe de contar con un listado de todos los artefactos que han sido elaborados para la construcción del producto de software final.
Importancia	Importante
Comentarios	Ninguno

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Tabla 22. RNF - Elicitación: Escalabilidad del sistema

RNF-003	Escalabilidad del sistema
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá estar en capacidad de permitir en el futuro el desarrollo de nuevas funcionalidades, modificar o eliminar funcionalidades después de su construcción y puesta en marcha inicial. Para lograr la escalabilidad es preciso que se incluya la arquitectura de software y hardware, así como las plataformas a donde se pretende escalar. Adecuar un documento de gestión de cambios. Agregar las ventajas y desventajas de las plataformas a donde se desea migrar en un futuro, así como las características de software o hardware asociados. Agregar características del lenguaje de programación a emplear en la futura escalabilidad así como una descripción de las bibliotecas asociadas.
Importancia	Vital
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 23. RNF - Elicitación: Flexibilidad del sistema

RNF-004	Flexibilidad del sistema
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá estar en capacidad de permitir en el futuro la modificación de módulos de acuerdo a nuevos requerimientos o que se requieran eliminar, sin que sufra cambios drásticos en su funcionamiento. Para lograr este objetivo se debe de agregar en el artefacto como se llevará a cabo la gestión del cambio, así como lo indicado en los RNF 002 y RNF 003. Asimismo deberá quedar especificado la estructura donde será posible modificar si es que el cliente pide una modificación total de interfaces producto de usuarios finales con discapacidad.
Importancia	Vital
Comentarios	Ninguno

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Tabla 24. RNF - Elicitación: Número mínimo de usuarios concurrentes

RNF-005	Número mínimo de usuarios concurrentes
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá permitir el acceso simultáneo y concurrente de 100 usuarios. Se debe dejar un artefacto donde se especifique las pruebas llevadas a cabo durante el POOLING de concurrencia, así como los problemas asociados a este tipo de pruebas. Si el producto pasa a un estado de SISTEMA DISTRIBUIDO se debe de especificar el modelo para resolver problemas de computación masiva utilizando un gran número de ordenadores organizados en clústeres incrustados en una infraestructura de telecomunicaciones distribuida. Especificar cómo el programador accede a los componentes de software (objetos) remotos, de la misma manera en que accedería a componentes locales, en un grupo de computadoras que usan un middleware entre los que destacan (RPC) y SOAP para conseguir un objetivo.
Importancia	Vital
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 25. RNF - Elicitación: Tiempo límite de espera para respuestas del sistema

RNF-006	Tiempo límite de espera para respuestas del sistema
Versión	0.1
Requisitos asociados	RNF-008 Tiempo límite de espera para casos excepcionales
Descripción	El tiempo límite de espera de respuesta del sistema es de 3 segundos para el 90% de servicios del sistema. Asimismo, se debe entregar un artefacto donde se especifique el tipo de pruebas que han llevado a cabo para lograr esta conclusión, así como modelo asociado, metodología empleada, técnica y metodología usada. Redactar en capítulo aparte el análisis algorítmico llevado a cabo.
Importancia	Importante

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Comentarios	Las excepciones se dan en caso de que la solicitud de datos al sistema sea mayor a la usual, por ejemplo la generación de reportes o consultas por los usuarios.

Fuente: Elaboración propia

Tabla 26. RNF - Elicitación: Tiempo límite de espera para casos excepcionales

RNF-007	Tiempo límite de espera para casos excepcionales
Versión	0.1
Requisitos asociados	Ninguno
Descripción	Este tiempo será de 10 segundos, y se da en alta carga de tráfico en la red de comunicación usada. El artefacto deberá especificar lo siguiente: • Cuáles son los casos excepcionales • Riesgo de cada caso excepcional • Como enfrentar a cada caso excepcional • Solución de cada caso excepcional • Especificar el tiempo límite de espera para cada caso excepcional, así como el de su riesgo asociado. • Especificar cuando un usuario final se enfrenta ante un caso excepcional
Importancia	Vital
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 27. RNF - Elicitación: Autorización para el uso del sistema

RNF-008	Autorización para el uso del sistema
Versión	0.1
Requisitos asociados	Todos los Requerimientos Funcionales
Descripción	Los usuarios deben identificarse antes de ingresar al sistema. Cada usuario contará con una entrada y una contraseña. El artefacto deberá especificar: • Cantidad de veces para realizar el punto de entrada • Tiempo permitido para hacer la entrada

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Importancia	Alta
Comentarios	El administrador del sistema es la persona encargada de asignar cuentas a los usuarios.

Fuente: Elaboración propia

Tabla 28. RNF - Elicitación: Número máximo de autentificaciones fallidas

RNF-009	Número máximo de autentificaciones fallidas
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema bloqueará al usuario con 4 intentos fallidos de autorizaciones consecutivas. En el artefacto se debe de especificar lo siguiente: • Lo que sucede si es que se ejecuta más de 4 intentos • Como se lleva a cabo la ejecución de un nuevo ciclo de intentos para acceder al sistema
Importancia	Normal
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 29. RNF - Elicitación: Canal seguro de autorización

RNF-010	Canal seguro de autorización
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá contar con un nivel de seguridad alta (encriptación de contraseña con letras números y caracteres como "|!$%&/"). Especificar el modelo de encriptación y la forma como se lleva a cabo el proceso inverso para la contraseña.
Importancia	Normal
Comentarios	Ninguno

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Tabla 30. RNF - Elicitación: Lenguaje

RNF-011	Lenguaje
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá ser implementado en el Lenguaje PHP. Agregar en el artefacto las consideraciones y comparaciones, con otros lenguajes de programación, por el cual se optó por el lenguaje definido. Asimismo el riesgo que se corre al asumir plataformas diferentes producto de la escalabilidad.
Importancia	Importante
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 31. RNF - Elicitación: Base de datos

RNF-012	Base de Datos
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema utilizará la base de datos origen, la cual fue desarrollada en MySQL. Especificar en el artefacto los riesgos asociados producto de la utilización de un número considerable de proyectos "ABIERTOS" al mismo tiempo y su efecto en la escalabilidad del producto. Considerar lo siguiente: • Cantidad de tablas dependientes • Cantidad de tablas independientes • Cantidad de tablas de migración • Cantidad de información (medidos en MB) que soporta cada tabla • Cantidad de transacciones que soporta en el presente y su proyección a futuro • Posibilidad de comercialización del producto de software construido
Importancia	Importante
Comentarios	Ninguno

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Tabla 32. RNF - Elicitación: Interfaces

RNF-013	Interfaz
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema se implementará utilizando una interface en PHP. En el artefacto se debe de especificar el riesgo asociado al cambio de interfaces producto de la utilización de lenguajes de programación más estables.
Importancia	Importante
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 33. RNF - Elicitación: Arquitectura del computador

RNF-014	Arquitectura del Computador
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema deberá funcionar tanto en la arquitectura de X86 como X32. Incluir en el artefacto la posibilidad de cambio y riesgo si se lleva a cabo la migración producto de la escalabilidad.
Importancia	Importante
Comentarios	Ninguno

Fuente: Elaboración propia

Tabla 34. RNF - Elicitación: Autorización para el uso del sistema

RNF-014	Autorización para el uso del sistema
Versión	0.1
Requisitos asociados	Ninguno
Descripción	El sistema, para su funcionamiento, deberá tener el correspondiente permiso del administrador. En caso de urgencia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	y no encontrarse el administrador, el segundo en el mando debe de otorgar la autorización.
Importancia	Importante
Comentarios	Ninguno

Fuente: Elaboración propia

4.2.2. Etapa de definición de escenarios y casos de uso

Esta etapa permite entender como la educción, elicitación y especificación de requerimientos de software pueden determinarse con precisión por medio de escenarios. Estos escenarios permiten entender las asociaciones de la problemática y los requerimientos no funcionales.

4.2.2.1. Identificación de escenarios

CUE - 00001: GESTION DE PROGRAMA PRINCIPAL

CU - 00001: INGRESAR A PROGRAMA PRINCIPAL

Tabla 35
Tabla 35. CU – Ingresar al programa principal

ESCENARIO DE CASO DE USO 00001			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.1	Estado de desarrollo	Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00001		
NOMBRE	Ingresa a programa principal		
OBJETIVO	acceder al programa para realizar la gestión requerimiento de proyectos		
DESCRIPCIÓN	El usuario se autenticará al sistema para hacer uso de todas las funciones de gestión de requerimientos.		
ACTORES	Usuario, Interfaz		
RNF RELACIONADOS	Autorización para el uso del sistema, Autorización para el uso del sistema , Número máximo de autorizaciones fallidas , Canal		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	seguro de autorización, Tiempo límite de espera para respuestas del sistema , Tiempo límite de espera para casos excepcionales
PRE-CONDICIONES	Instalación correcta del sistema y previa autenticación segura. Si hubiese alguna autenticación fallida, deshabilitar el login
ESCENARIO PRINCIPAL	1. El Usuario accede al Programa 2. La Interfaz carga la Pantalla de Inicio del Programa VPrincipal 3. Se muestra la pantalla al usuario
ESCENARIO ALTERNATIVO	-
ESCENARIO DE EXCEPCIÓN	-
POST-CONDICIONES	-

Fuente: Elaboración propia

CU - 00002: SALIR DE PROGRAMA PRINCIPAL

Tabla 36. CU – Salir del programa principal

ESCENARIO DE CASO DE USO 00002			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.1	Estado de desarrollo	Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00002		
NOMBRE	Salir de Programa Principal		
OBJETIVO	Finalizar el uso del sistema		
DESCRIPCIÓN	El usuario debe poder finalizar el uso del sistema		
ACTORES	Usuario, Interfaz		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

RNF RELACIONADOS	Facilidad de Uso , Escalabilidad del sistema , Fácil mantenimiento , Tiempo límite de espera para respuestas del sistema , Flexibilidad del sistema , Lenguaje , Interfaz
PRE-CONDICIONES	-
ESCENARIO PRINCIPAL	1. El Usuario Presiona el Botón Salir del Programa 2. La Interfaz cierra todas las pantallas de la aplicación
ESCENARIO ALTERNATIVO	1. El Usuario se ubica en la Pestaña Archivo y selecciona la Opción Salir del Programa
ESCENARIO DE EXCEPCIÓN	-
POST-CONDICIONES	-

Fuente: Elaboración propia

CUE - 00002: GESTION DE PROYECTO

CU - 00003: INGRESAR AL PROYECTO

Tabla 37. CU – Ingresar al proyecto

ESCENARIO DE CASO DE USO 00003			
PROYECTO	REMAN		
AUTOR	Quispe Huamán, Kevin Percy Montero Nifla Max		
VERSIÓN	0.3	Estado de desarrollo	Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00003		
NOMBRE	Ingresar a Proyecto Seleccionado		
OBJETIVO	Acceder aun un proyecto creado		
DESCRIPCIÓN	El usuario puede acceder a un proyecto en la cual realizara la gestión de los requerimientos que desarrollara.		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ACTORES	Usuario, Sistema, Interfaz
RNF RELACIONADOS	Facilidad de Uso , Escalabilidad del sistema , Fácil mantenimiento , Tiempo límite de espera para respuestas del sistema , Flexibilidad del sistema , Lenguaje , Interfaz
PRE-CONDICIONES	-
ESCENARIO PRINCIPAL	2. El Usuario presiona el botón abrir proyecto "btnVPAbrir" 3. La Interfaz muestra los proyectos con la extensión ".reman" 4. La Interfaz manda la dirección del proyecto al Sistema 5. El Sistema ubica el directorio del proyecto 6. El Sistema carga los datos del proyecto y los manda a la interfaz 7. El Sistema envía los datos a la interfaz 8. La Interfaz muestra los datos en los diferentes libros del proyecto.
ESCENARIO ALTERNATIVO	1.1 El Usuario se ubica en la pestaña archivo y presiona "Abrir Proyecto" 1.2 Continua el paso 2
ESCENARIO DE EXCEPCIÓN	2.1. La interfaz no encuentra archivos con la extensión ".reman" 2.2. La interfaz notifica al Usuario que no existe ningún proyecto creado 2.3 FIN
POST-CONDICIONES	-

Fuente: Elaboración propia

CU - 00004: SALIR DE PROYECTO ACTUAL

Tabla 38. CU – Salir del proyecto actual

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ESCENARIO DE CASO DE USO 00004			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.1	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00004		
NOMBRE	Salir de Proyecto Actual		
OBJETIVO	Cerrar un proyecto		
DESCRIPCIÓN	El usuario debe de poder salir del proyecto en la que está trabajando		
ACTORES	Usuario, Sistema.		
RNF RELACIONADOS	Facilidad de Uso , Escalabilidad del sistema		
PRE-CONDICIONES	-		
ESCENARIO PRINCIPAL	El usuario se ubica en la pestaña archivo y selecciona la opción. Cerrar Proyecto. La Interfaz envía un mensaje de confirmación. El Usuario responde que si La interfaz Cierra los libros del proyecto en cuestión. La interfaz carga la pantalla de Inicio del Sistema ”VPrincipal”.		
ESCENARIO ALTERNATIVO	-		
ESCENARIO DE EXCEPCIÓN	-		
POST-CONDICIONES	-		

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CU - 00005: CREAR PROYECTO

Tabla 39. CU – Crear proyecto

ESCENARIO DE CASO DE USO 00005			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.1	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00005		
NOMBRE	Crear un Proyecto		
OBJETIVO	Proporcionar el detalle de un proyecto		
DESCRIPCIÓN	El usuario registra cada proyecto. Agregando nombre, organizaciones participantes, fechas de inicio y final, entre otros datos de cada Proyecto establecida por el usuario.		
ACTORES	Usuario, Sistema, Interfaz		
RNF RELACIONADOS	Interfaz , Facilidad de Uso , Facil mantenimiento , Escalabilidad del sistema		
PRE-CONDICIONES	EL Usuario ejecuto el sistema Los actores (fuente y especialista) debe de estar creados Las organizaciones deben estar creadas		
ESCENARIO PRINCIPAL	1. El Usuario presiona el botón “nuevoProyecto” 2. La Interfaz muestra la panta “VNuevoProyecto” 3. El usuario llena los datos referidos al Proyecto Nuevo 4. La interfaz manda los datos al sistema con la función crearProyecto() 5. El Sistema genera el Directorio donde se almacenará el proyecto 6. El sistema genera un diálogo donde se guardarán las propiedades del proyecto 7. El Sistema crea el archivo histórico.		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	8. El Sistema asigna a un líder por defecto al Proyecto 9. El Sistema envía una notificación de creación a la Interfaz 10. La Interfaz Notifica al Usuario que el Proyecto ha sido creado
ESCENARIO ALTERNATIVO	-
ESCENARIO DE EXCEPCIÓN	3.1. Valores Nulos en la Entrada 3.2. se muestra un mensaje de Datos mal llenados 3.3. Se cierra la ventana 5.1 Falta de permisos en la ubicación de la carpeta seleccionada 5.2 Se cierra la ventana
POST-CONDICIONES	-

Fuente: Elaboración propia

CU - 00006: MODIFICAR PROYECTO

Tabla 40. CU – Modificar proyecto

ESCENARIO DE CASO DE USO 00006			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.5	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00046		
NOMBRE	Modificar Proyecto		
OBJETIVO	Cambiar los datos de un Proyecto.		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

DESCRIPCIÓN	El usuario puede modificar cualquier campo de la plantilla proyecto. Siguiendo las condiciones de crear un proyecto
ACTORES	Usuario, sistema, Interfaz
RNF RELACIONADOS	Base de Datos, Lenguaje
PRE-CONDICIONES	El proyecto debe de estar creado
ESCENARIO PRINCIPAL	1. El Usuario se ubica en la pestaña archivo "ModificarProyecto" 2. La Interfaz muestra la pantalla "ModificarProyecto" 3. El usuario llena los nuevos datos referidos al Proyecto 4. La interfaz manda los datos al sistema 5. El Sistema se ubica en el directorio del proyecto 6. El sistema modifica el diálogo donde se guardan las propiedades del proyecto 7. El sistema envía una notificación a la interfaz 8. La Interfaz Notifica al Usuario de la modificación del proyecto 9. La Interfaz carga el proyecto en pantalla
ESCENARIO ALTERNATIVO	
ESCENARIO DE EXCEPCIÓN	
POST-CONDICIONES	- Los datos se modifican en la base de datos codificando Regresar a la lista

Fuente: Elaboración propia

CU - 00007: ELIMINAR PROYECTO

Tabla 41. CU – Eliminar proyecto

ESCENARIO DE CASO DE USO 00007	
PROYECTO	REMAN

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

AUTOR	Quispe Huamán, Kevin Percy		
VERSIÓN	0.1	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00047		
NOMBRE	Eliminar Proyecto		
OBJETIVO	Desechar los proyectos que no sirvan para el usuario		
DESCRIPCIÓN	El usuario puede eliminar los proyectos que no necesite. Al seleccionar la eliminación de un Proyecto; el usuario debe de confirmar la realización de la eliminación.		
ACTORES	Usuario, Sistema		
RNF RELACIONADOS	Interfaz , Facilidad de Uso , Fácil mantenimiento , Escalabilidad del sistema		
PRE-CONDICIONES	El proyecto debe de existir o haber sido creado		
ESCENARIO PRINCIPAL	1. El Usuario se ubica en la pestaña archivo “eliminarProyecto” 2. La Interfaz muestra un mensaje de confirmación 3. El usuario responde que si 4. La interfaz llama a la función eliminarProyecto() 5. El Sistema Elimina el Proyecto 6. El sistema devuelve un estado verdadero a la interfaz 7. La Interfaz notifica al Usuario 8. La Interfaz carga la pantalla “VPrincipal”		
ESCENARIO ALTERNATIVO	-		
ESCENARIO DE EXCEPCIÓN	-		
POST-CONDICIONES	-		

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CUE - 00003: GESTION DE EDUCCIÓN DE REQUERIMIENTOS

CU - 00008: CREAR EDUCCIÓN DE REQUERIMIENTOS

Tabla 42. CU – Crear educción de requerimientos

ESCENARIO DE CASO DE USO 00008			
PROYECTO	REMAN		
AUTOR	Montero Nifla, Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.1	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00008		
NOMBRE	Crear Educción de Requerimientos		
OBJETIVO	Proporcionar el detalle de una educción de requerimiento.		
DESCRIPCIÓN	El usuario registra cada educción de requerimiento. Agregando código, nombre, versión, descripción, observaciones, entre otros datos de cada educción establecida por el usuario		
ACTORES	Usuario, Sistema, Interfaz, XML		
NFR RELACIONADOS	Interfaz , Facilidad de Uso , Fácil mantenimiento , Escalabilidad del sistema		
PRE-CONDICIONES	EL Usuario ejecutó el sistema El Usuario debe de estar dentro de un proyecto existente o ya creado El usuario debe de haber creado las organizaciones Deben de estar creados los actores que intervienen en la educción Deben estar creados las fuentes (stakeholder)que intervienen en la educción		
ESCENARIO PRINCIPAL	1. El usuario presiona el botón “btnVPEduccion” 2. La Interfaz muestra la pantalla “VAEduccion” 3. El usuario llena los datos 4. El usuario presiona el botón “btnAEAgregar”		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	5. La Interfaz llama a la función crearEduccion(). 6. El Sistema crea el XML de la educción 7. El Sistema aumenta el contador de educciones 8. El Sistema envía los datos al XML 9. XML guarda los datos de la nueva educción 10. XML envía notificación de confirmación al Sistema 11. El Sistema envía una notificación de creación a la Interfaz 12. La Interfaz Notifica al Usuario que la Educción ha sido creada
ESCENARIO ALTERNATIVO	3.1 El Usuario lleno los datos incorrectamente 3.2 La interfaz notifica al Usuario 3.3 Se vuelve al Paso 2
ESCENARIO DE EXCEPCIÓN	5.1 Valores nulos en los parámetros de entrada 5.2 Notificación de ERROR 9.1 falta de Permisos en la ubicación de la carpeta seleccionada 9.2 Notificación de ERROR
POST-CONDICIONES	Se crea el xml con nombre “eduX” donde X es el número de la educción Aumenta en 1 el contador e educciones del archivo propiedades

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CU - 00009: MODIFICAR LA EDUCCIÓN DE REQUERIMIENTO

Tabla 43. CU – Modificar la educción de requerimientos

ESCENARIO DE CASO DE USO 00009			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.4	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00009		
NOMBRE	Modificar la Educción de Requerimiento		
OBJETIVO	Cambiar los datos de una educción		
DESCRIPCIÓN	El usuario puede modificar cualquier campo de la educción de requisitos. Siguiendo las condiciones de crear una educción		
ACTORES	Usuario, sistema, Interfaz, XML		
RNF RELACIONADOS	Base de Datos, Lenguaje		
PRE-CONDICIONES	La educción de requisito debe de estar creado El usuario no puede cambiar el código de la educción		
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “modificarEducción” 2. Interfaz llama a la función recuperarEduccion(). 3. XML extrae los datos de la última versión actual de la educción 4. XML envía un objeto tipo Educción al Sistema 5. El Sistema envía los datos del objeto Educción a la interfaz 6. La interfaz Muestra la pantalla VMEduccion 7. El Usuario llena los datos que desee Actualizar 8. El usuario presiona el botón “btnVEDGuardar” 9. La Interfaz llama a la función modificarEduccion(..) 10. El Sistema modifica los datos en la versión actual 11. El Sistema envía los datos a XML		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	12. XML guarda los nuevos datos en la versión actual 13. XML envía un valor booleano True a la Interfaz 14. La interfaz notifica al Usuario de la modificación realizada
ESCENARIO ALTERNATIVO	7.1 El Usuario lleno los datos incorrectamente 7.2 La interfaz notifica al Usuario 7.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	13.1 El Sistema no puede actualizar la Educción 13.2 El Sistema envía notificación a la Interfaz 13.3 La Interfaz Notifica al Usuario 13.4 FIN
POST-CONDICIONES	- Se Modifica la educción actual del xml

Fuente: Elaboración propia

CU - 00010: ELIMINAR LA EDUCCIÓN DE REQUERIMIENTO

Tabla 44. CU – Eliminar la educción de requerimientos

ESCENARIO DE CASO DE USO 00010			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.1	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00010		
NOMBRE	Eliminar la Educción de Requerimiento		
OBJETIVO	Desechar las educciones que no sirvan para el usuario		
DESCRIPCIÓN	El usuario puede eliminar las educciones que no necesite. Al seleccionar la eliminación de una educción; el usuario debe de confirmarla realización de la eliminación.		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ACTORES	Usuario, Sistema, Interfaz
RNF RELACIONADOS	Interfaz , Facilidad de Uso , Facil mantenimiento , Escalabilidad del sistema
PRE-CONDICIONES	La educción debe de existir dentro del proyecto
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción "eliminarEducción" 2. La interfaz llama a la función eliminarEduccion() 3. El Sistema retorna un valor booleano True 4. La Interfaz Notifica al usuario sobre la Eliminación
ESCENARIO ALTERNATIVO	
ESCENARIO DE EXCEPCIÓN	2.1. Falta de permisos en la ubicación de la carpeta seleccionada 2.2. Fin del Programa
POST-CONDICIONES	- Se elimina el xml de la educción

Fuente: Elaboración propia

CU - 00011: MOSTRAR LA EDUCCIÓN DE REQUERIMIENTO

Tabla 45. CU – Mostrar la educción de requerimientos

ESCENARIO DE CASO DE USO 00011			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.4	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00011		
NOMBRE	Mostrar la Educción de Requerimiento		
OBJETIVO	Ver las plantillas de todas las educciones con sus datos ingresados		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

DESCRIPCIÓN	El usuario podrá visualizar cada educción creada y llenada con sus datos. La estructura de visualización de los datos es según las plantillas proporcionada en Diseño. El sistema debe de mostrar automáticamente las educciones creadas.
ACTORES	Usuario, sistema, Interfaz ,XML
RNF RELACIONADOS	Escalabilidad del sistema, Interfaz , Tiempo límite de espera para respuestas del sistema
PRE-CONDICIONES	Las educciones de requisitos deben de existir o haber sido creadas
ESCENARIO PRINCIPAL	1. El usuario se sitúa sobre la educción que desea ver 2. La interfaz llama a la función mostrarEduccion() 3. El Sistema pide los datos al XML 4. XML extrae los datos de la última versión actual de la educción 5. XML envía los datos al sistema 6. El Sistema envía los datos a la Interfaz 7. La Interfaz muestra los datos en pantalla en la ventana "VTEduccion"
ESCENARIO ALTERNATIVO	-
ESCENARIO DE EXCEPCIÓN	4.1. falta de permisos en la ubicación de la carpeta seleccionada
POST-CONDICIONES	Se elimina el xml de la educción

Fuente: Elaboración propia

CU - 00012: VERSIONAR EDUCCIÓN DE REQUERIMIENTO

Tabla 46. CU – versionar la educción de requerimientos

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ESCENARIO DE CASO DE USO 00012			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.7	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00012		
NOMBRE	Versionar Educción de Requerimiento		
OBJETIVO	Proporcionar la trazabilidad de cada educción a lo largo de la realización del proyecto		
DESCRIPCIÓN	El usuario ingresara la versión del requisito; el cual debe de ser incremental y mayor respecto a los anteriores valores de la versión. La versión inicialmente debe ser 1.0. La versión cambiara según el usuario haga sus modificaciones en la educción. El usuario es que hace el versiona miento.		
ACTORES	Usuario, Sistema, Interfaz , XML		
RNF RELACIONADOS	Lenguaje , Base de Datos , Facil mantenimiento , Escalabilidad del sistema , Flexibilidad del sistema , Arquitectura del Computador , Facilidad de Uso		
PRE-CONDICIONES	El usuario debe de realizar una modificación de una educción para realizar el versionamiento de dicha educción		
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “modificarEduccion” 2. La Interfaz llama a la función recuperarEduccion() 3. El Sistema pide los datos al XML 4. XML extrae los datos de la última versión actual de la educción 5. XML envía un objeto Educción al sistema 6. El Sistema envía los datos a la Interfaz 7. La Interfaz muestra los datos al Usuario por la ventana “VMEduccion” 8. El Usuario llena los datos nuevos		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	9. El usuario presiona el botón “btnVEDVersionar” 10. La Interfaz Muestra la pantalla “VVersionarElem” 11. El usuario llena los datos de la versión 12. La Interfaz llama a la función versionarEduccion() 13. El Sistema recupera la Educción a Versionar 14. El Sistema agrega la nueva versión 15. El Sistema envía a XML la nueva versión 16. XML guarda la nueva versión 17. XML retorna un valor booleano True 18. El Sistema retorna un valor booleano True a la Interfaz 19. La Interfaz Notifica al Usuario que la Educción ha sido Versionada
ESCENARIO ALTERNATIVO	9.1 El Usuario lleno los datos incorrectamente 9.2 La interfaz notifica al Usuario 9.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	17.1la versión generada debe ser mayor a la anterior 17.2valores nulos en los parámetros de entrada 17.3falta de permisos en la ubicación de la carpeta seleccionada 17.4 FIN
POST-CONDICIONES	Se versiona la educción en el archivo xml Se guardan los valores de la educción actual

Fuente: Elaboración propia

CU - 00013: RESTAURAR EDUCCIÓN DE REQUERIMIENTO

Tabla 47. CU – Restaurar la educción de requerimientos

ESCENARIO DE CASO DE USO 00013

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.4	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00013		
NOMBRE	Restaurar Educción de Requerimiento		
OBJETIVO	Poder restaurar el requerimiento a nivel elemento a cualquier versión registrada por el sistema.		
DESCRIPCIÓN	Restaurar es una de las opciones dentro de cada requerimiento y es accesible desde las opciones de modificar.		
ACTORES	Usuario, sistema. Interfaz, XML		
RNF RELACIONADOS	Fácil Mantenimiento , Lenguaje , Base de Datos , Flexibilidad del sistema		
PRE-CONDICIONES	El usuario debe de encontrarse dentro del requerimiento a restaurar.		
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “RestaurarEducción” 2. La Interfaz llama a la función getHistorico(). 3. El Sistema pide los datos al XML 4. XML extrae los datos de educción 5. XML envía los datos al sistema 6. El Sistema envía el Histórico de la educción seleccionada 7. La Interfaz muestra la pantalla “VRestaura” 8. El Usuario selecciona la versión que se quiere restaurar 9. El usuario presiona el botón “Restaurar” 10. La Interfaz llama a la fuincion restaurarVersion() 11. EL Sistema modifica el XML de la educción 12. El Sistema envía los datos a XML 13. XML guarda la nueva versión 14. XML envía un valor booleano True al Sistema 15. EL Sistema notifica a la interfaz		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	16. La interfaz notifica al Usuario que la Educción ha sido restaurada
ESCENARIO ALTERNATIVO	9.1 El Usuario lleno los datos incorrectamente 9.2 La interfaz notifica al Usuario 9.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	12.1 El Sistema no puede versionar la Educción 12.2 El Sistema envía notificación a la Interfaz 12.3 La Interfaz Notifica al Usuario 12.4 FIN 13.1 debe existir la versión a recuperar 13.2 valores nulos en los parámetros de entrada 13.3 falta de permisos en la ubicación de la carpeta seleccionad
POST-CONDICIONES	- Se modifica el estado actual del xml por el de la versión seleccionada

Fuente: Elaboración propia

CUE - 00004: GESTION DE ELICITACIÓN DE REQUERIMIENTOS

CU - 00014: CREAR ELICITACIÓN DE REQUERIMIENTOS

Tabla 48. CU – Crear la elicitación de requerimientos

ESCENARIO DE CASO DE USO 00014			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00014		
NOMBRE	Crear Elicitación de Requerimientos		
OBJETIVO	Proporcionar el detalle de una elicitación de requerimiento.		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

DESCRIPCIÓN	El usuario registra cada elicitación de requerimiento. Agregando código, nombre, versión, descripción, observaciones, entre otros datos de cada elicitación establecida por el usuario. En esta parte se registrarán los requerimientos del sistema orientados al analista, tomando en cuenta que una educción puede contener una o más elicitaciones.
ACTORES	Usuario, Sistema, Interfaz, XML
RNF RELACIONADOS	Interfaz , Facilidad de Uso , Fácil mantenimiento , Escalabilidad del sistema
PRE-CONDICIONES	EL Usuario ejecuto el sistema El Usuario debe de estar dentro de un proyecto existente o ya creado Debe de haber creado una educción de requerimiento
ESCENARIO PRINCIPAL	1. El usuario presiona el botón “btnVPElicitacion” 2. La Interfaz muestra la pantalla “VAElicitacion” 3. El usuario llena los datos 4. El usuario presiona el botón “btnAELAgregar” 5. La Interfaz llama a la función crearElicitacion (). 6. El Sistema crea el XML de la Elicitación 7. El Sistema aumenta el contador de Elicitación 8. El Sistema envía los datos al XML 9. XML guarda los datos de la nueva Elicitación 10. XML envía notificación de confirmación al Sistema 11. El Sistema envía una notificación de creación a la Interfaz 12. La Interfaz Notifica al Usuario que la Elicitación ha sido creada
ESCENARIO ALTERNATIVO	3.1 El Usuario lleno los datos incorrectamente 3.2 La interfaz notifica al Usuario 3.3 Se vuelve al Paso 2
ESCENARIO DE EXCEPCIÓN	5.1 Valores nulos en los parámetros de entrada 5.2 Notificación de ERROR 9.1 falta de Permisos en la ubicación de la carpeta seleccionada

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	9.2 Notificación de ERROR
POST-CONDICIONES	-

Fuente: Elaboración propia

CU - 00015: MODIFICAR LA ELICITACIÓN DE REQUERIMIENTO

Tabla 49. CU – Modificar la elicitación de requerimientos

ESCENARIO DE CASO DE USO 00015			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00015		
NOMBRE	Modificar la Elicitación de Requerimiento		
OBJETIVO	Cambiar los datos de una Elicitación		
DESCRIPCIÓN	El usuario puede modificar cualquier campo de la Elicitación de requisitos. Siguiendo las condiciones de crear una Elicitación		
ACTORES	Usuario, sistema		
RNF RELACIONADOS	Base de Datos , Lenguaje		
PRE-CONDICIONES	La Elicitación de requisito debe de estar creado El usuario no puede cambiar el código de la elicitación		
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “modificar Elicitación” 2. Interfaz llama a la función recuperar Elicitación (). 3. XML extrae los datos de la última versión actual de la Elicitación 4. XML envía un objeto tipo Elicitación al Sistema		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	5. El Sistema envía los datos del objeto Elicitación a la interfaz 6. La interfaz Muestra la pantalla VM Elicitación 7. El Usuario llena los datos que desee Actualizar 8. El usuario presiona el botón “btnVEDGuardar” 9. La Interfaz llama a la función modificar Elicitación (..) 10. El Sistema modifica los datos en la versión actual 11. El Sistema envía los datos a XML 12. XML guarda los nuevos datos en la versión actual 13. XML envía un valor booleano True a la Interfaz 14. La interfaz notifica al Usuario de la modificación realizada
ESCENARIO ALTERNATIVO	7.1 El Usuario lleno los datos incorrectamente 7.2 La interfaz notifica al Usuario 7.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	13.1 El Sistema no puede actualizar la Elicitación 13.2 El Sistema envía notificación a la Interfaz 13.3 La Interfaz Notifica al Usuario 13.4 FIN
POST-CONDICIONES	- Se Modifica la Elicitación actual del xml

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CU - 00016: ELIMINAR LA ELICITACIÓN DE REQUERIMIENTO

Tabla 50. CU – Eliminar la elicitación de requerimientos

ESCENARIO DE CASO DE USO 00016			
PROYECTO	REMAN		
AUTOR	Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00016		
NOMBRE	Eliminar la Elicitación de Requerimiento		
OBJETIVO	Desechar las elicitaciones que no sirvan para el usuario		
DESCRIPCIÓN	El usuario puede eliminar las elicitaciones que no necesite. Al seleccionar la eliminación de una Elicitación; el usuario debe de confirmar la realización de la eliminación.		
ACTORES	Usuario, Sistema		
RNF RELACIONADOS	Interfaz , Facilidad de Uso , Facil mantenimiento , Escalabilidad del sistema		
PRE-CONDICIONES	La Elicitación debe de existir dentro del proyecto La Elicitación no debe de tener especificaciones de requerimiento		
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “eliminar Elicitación” 2. La interfaz llama a la función eliminar Elicitación () 3. El Sistema retorna un valor booleano True 4. La Interfaz Notifica al usuario sobre la Eliminación		
ESCENARIO ALTERNATIVO			
ESCENARIO DE EXCEPCIÓN	2.1. Falta de permisos en la ubicación de la carpeta seleccionada 2.2. Fin del Programa		
POST-CONDICIONES	- Se elimina el xml de la Elicitación		

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CU - 00017: MOSTRAR LA ELICITACIÓN DE REQUERIMIENTO

Tabla 51. CU – Mostrar la elicitación de requerimientos

ESCENARIO DE CASO DE USO 00017			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00017		
NOMBRE	Mostrar la Elicitación de Requerimiento		
OBJETIVO	Ver las plantillas de todas las elicitaciones con sus datos ingresados		
DESCRIPCIÓN	El usuario podrá visualizar cada Elicitación creada y llenada con sus datos. La estructura de visualización de los datos es según las plantillas proporcionada en Diseño. El sistema debe de mostrar automáticamente las educciones creadas.		
ACTORES	Usuario, sistema		
RNF RELACIONADOS	Escalabilidad del sistema , Interfaz , Tiempo límite de espera para respuestas del sistema		
PRE-CONDICIONES	Las elicitaciones de requisitos deben de existir o haber sido creadas		
ESCENARIO PRINCIPAL	1. El usuario se sitúa sobre la Elicitación que desea ver 2. La interfaz llama a la función mostrarE Elicitación () 3. El Sistema pide los datos al XML 4. XML extrae los datos de la última versión actual de la Elicitación 5. XML envía los datos al sistema 6. El Sistema envía los datos a la Interfaz 7. La Interfaz muestra los datos en pantalla en la ventana “VT Elicitación”		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ESCENARIO ALTERNATIVO	-
ESCENARIO DE EXCEPCIÓN	4.1. falta de permisos en la ubicación de la carpeta seleccionada
POST-CONDICIONES	Se elimina el xml de la Elicitación

Fuente: Elaboración propia

CU - 00018: VERSIONAR ELICITACIÓN DE REQUERIMIENTO

Tabla 52. CU – Versionar la elicitación de requerimientos

ESCENARIO DE CASO DE USO 00018			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00018		
NOMBRE	Versionar Elicitación de Requerimiento		
OBJETIVO	Proporcionar la trazabilidad de cada elicitación a lo largo de la realización del proyecto		
DESCRIPCIÓN	El usuario ingresara la versión de la elicitación del requisito; el cual debe de ser incremental y mayor respecto a los anteriores valores de la versión. La versión inicialmente debe ser 1.0. La versión cambiara según el usuario haga sus modificaciones en la elicitación. El usuario es que hace el versionamiento.		
ACTORES	Usuario, Sistema		
RNF RELACIONADOS	Lenguaje , Base de Datos , Fácil mantenimiento , Escalabilidad del sistema , Flexibilidad del sistema , Arquitectura del Computador		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

PRE-CONDICIONES	El usuario debe de realizar una modificación de una elicitación para realizar el versionamiento de dicha elicitación
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “modificarElicitación” 2. La Interfaz llama a la función recuperarElicitación () 3. El Sistema pide los datos al XML 4. XML extrae los datos de la última versión actual de la Elicitación 5. XML envía un objeto Elicitación al sistema 6. El Sistema envía los datos a la Interfaz 7. La Interfaz muestra los datos al Usuario por la ventana “VM Elicitación” 8. El Usuario llena los datos nuevos 9. El usuario presiona el botón “btnVELVersionar” 10. La Interfaz Muestra la pantalla “VVersionarElem” 11. El usuario llena los datos de la versión 12. La Interfaz llama a la función versionar Elicitación () 13. El Sistema recupera la Elicitación a Versionar 14. El Sistema agrega la nueva versión 15. El Sistema envía a XML la nueva versión 16. XML guarda la nueva versión 17. XML retorna un valor booleano True 18. El Sistema retorna un valor booleano True a la Interfaz 19. La Interfaz Notifica al Usuario que la Elicitación ha sido Versionada
ESCENARIO ALTERNATIVO	9.1 El Usuario lleno los datos incorrectamente 9.2 La interfaz notifica al Usuario 9.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	17.1la versión generada debe ser mayor a la anterior 17.2valores nulos en los parámetros de entrada 17.3falta de permisos en la ubicación de la carpeta seleccionada 17.4 FIN

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

POST-CONDICIONES	Se versiona la Elicitación en el archivo xml Se guardan los valores de la educción actual

Fuente: Elaboración propia

CU - 00019: RESTAURAR ELICITACIÓN DE REQUERIMIENTO

Tabla 53. CU – Restaurar la elicitación de requerimientos

ESCENARIO DE CASO DE USO 00019			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.2	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00019		
NOMBRE	Restaurar Elicitación de Requerimiento		
OBJETIVO	Poder restaurar el requerimiento a nivel elemento a cualquier versión registrada por el sistema.		
DESCRIPCIÓN	Restaurar es una de las opciones dentro de cada requerimiento y es accesible desde las opciones de modificar.		
ACTORES	Usuario, sistema.		
RNF RELACIONADOS	Flexibilidad , Fácil Mantenimiento , Lenguaje , Base de Datos		
PRE-CONDICIONES	El usuario debe de encontrarse dentro del requerimiento a restaurar.		
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “modificar Elicitación” 2. La Interfaz llama a la función recuperar Elicitación () 3. El Sistema pide los datos al XML 4. XML extrae los datos de la última versión actual de la Elicitación 5. XML envía un objeto Elicitación al sistema		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	6. El Sistema envía los datos a la Interfaz 7. La Interfaz muestra los datos al Usuario por la ventana "VM Elicitación" 8. El Usuario llena los datos nuevos 9. El usuario presiona el botón "btnVELVersionar" 10. La Interfaz Muestra la pantalla "VVersionarElem" 11. El usuario llena los datos de la versión 12. La Interfaz llama a la función versionar Elicitación () 13. El Sistema recupera la Elicitación a Versionar 14. El Sistema agrega la nueva versión 15. El Sistema envía a XML la nueva versión 16. XML guarda la nueva versión 17. XML retorna un valor booleano True 18. El Sistema retorna un valor booleano True a la Interfaz 19. La Interfaz Notifica al Usuario que la Elicitación ha sido Versionada
ESCENARIO ALTERNATIVO	9.1 El Usuario lleno los datos incorrectamente 9.2 La interfaz notifica al Usuario 9.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	17.1la versión generada debe ser mayor a la anterior 17.2valores nulos en los parámetros de entrada 17.3falta de permisos en la ubicación de la carpeta seleccionada 17.4 FIN
POST-CONDICIONES	Se modifica el estado actual del xml por el de la versión seleccionada

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CUE - 00005: GESTION DE ESPECIFICACIÓN DE REQUERIMIENTOS

CU - 00020: CREAR ESPECIFICACIÓN DE REQUERIMIENTO

Tabla 54. CU – Crear la especificación de requerimientos

ESCENARIO DE CASO DE USO 000			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00020		
NOMBRE	Agregar Especificación de Requerimientos		
OBJETIVO	Proporcionar el detalle de una especificación de requerimiento.		
DESCRIPCIÓN	El usuario registra cada especificación de requerimiento. Agregando código, nombre, versión, descripción, observaciones, entre otros datos de cada especificación establecida por el usuario. En esta parte se registrarán los requerimientos del sistema orientados al desarrollador, tomando en cuenta que una elicitación puede contener una o más especificaciones.		
ACTORES	Usuario, Sistema		
RNF RELACIONADOS	Interfaz , Facilidad de Uso , Fácil mantenimiento , Escalabilidad del sistema		
PRE-CONDICIONES	- EL Usuario ejecutó el sistema - El Usuario debe de estar dentro de un proyecto existente o ya creado - Debe de haber creado una elicitación de requerimiento		
ESCENARIO PRINCIPAL	1. El usuario presiona el botón “btnVPEspecificación” 2. La Interfaz muestra la pantalla “VAEspecificación” 3. El usuario llena los datos 4. El usuario presiona el botón “btnAEspecificación” 5. La Interfaz llama a la función crearEspecificación ().		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	6. El Sistema crea el XML de la Especificación 7. El Sistema aumenta el contador de Especificación 8. El Sistema envía los datos al XML 9. XML guarda los datos de la nueva Especificación 10. XML envía notificación de confirmación al Sistema 11. El Sistema envía una notificación de creación a la Interfaz 12. La Interfaz Notifica al Usuario que la Especificación ha sido creada
ESCENARIO ALTERNATIVO	3.1 El Usuario lleno los datos incorrectamente 3.2 La interfaz notifica al Usuario 3.3 Se vuelve al Paso 2
ESCENARIO DE EXCEPCIÓN	5.1 Valores nulos en los parámetros de entrada 5.2 Notificación de ERROR 9.1 falta de Permisos en la ubicación de la carpeta seleccionada 9.2 Notificación de ERROR
POST-CONDICIONES	-

Fuente: Elaboración propia

CU - 00021: MODIFICAR LA ESPECIFICACIÓN DE REQUERIMIENTO

Tabla 55. CU – Crear la especificación de requerimientos

ESCENARIO DE CASO DE USO 00021			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00021		
NOMBRE	Modificar la Especificación de Requerimiento		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

OBJETIVO	Cambiar los datos de una especificación
DESCRIPCIÓN	El usuario puede modificar cualquier campo de la especificación de requisitos. Siguiendo las condiciones de crear una especificación
ACTORES	Usuario, sistema, Interfaz ,XML
RNF RELACIONADOS	Base de Datos , Lenguaje
PRE-CONDICIONES	La especificación de requisito debe de estar creado El usuario no puede cambiar el código de la especificación
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción "modificar Especificación" 2. Interfaz llama a la función recuperar Especificación (). 3. XML extrae los datos de la última versión actual de la Especificación 4. XML envía un objeto tipo Especificación al Sistema 5. El Sistema envía los datos del objeto Especificación a la interfaz 6. La interfaz Muestra la pantalla VMEspecificación 7. El Usuario llena los datos que desee Actualizar 8. El usuario presiona el botón "btnVESGuardar" 9. La Interfaz llama a la función modificar Especificación (..) 10. El Sistema modifica los datos en la versión actual 11. El Sistema envía los datos a XML 12. XML guarda los nuevos datos en la versión actual 13. XML envía un valor booleano True a la Interfaz 14. La interfaz notifica al Usuario de la modificación realizada
ESCENARIO ALTERNATIVO	7.1 El Usuario lleno los datos incorrectamente 7.2 La interfaz notifica al Usuario 7.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	13.1 El Sistema no puede actualizar la Especificación 13.2 El Sistema envía notificación a la Interfaz

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	13.3 La Interfaz Notifica al Usuario 13.4 FIN
POST-CONDICIONES	- Se Modifica la Elicitación actual del xml

Fuente: Elaboración propia

CU - 00022: ELIMINAR LA ESPECIFICACIÓN DE REQUERIMIENTO

Tabla 56. CU – Eliminar la especificación de requerimientos

ESCENARIO DE CASO DE USO 00021			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00022		
NOMBRE	Eliminar la Especificación de Requerimiento		
OBJETIVO	Desechar las especificaciones que no sirvan para el usuario		
DESCRIPCIÓN	El usuario puede eliminar las especificaciones que no necesite. Al seleccionar la eliminación de una especificación; el usuario debe de confirmar la realización de la eliminación.		
ACTORES	Usuario, Sistema		
RNF RELACIONADOS	Interfaz , Facilidad de Uso , Fácil mantenimiento , Escalabilidad del sistema		
PRE-CONDICIONES	La especificación debe de existir dentro del proyecto		
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “eliminar Especificación” 2. La interfaz llama a la función eliminar Especificación () 3. El Sistema retorna un valor booleano True 4. La Interfaz Notifica al usuario sobre la Eliminación		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ESCENARIO ALTERNATIVO	
ESCENARIO DE EXCEPCIÓN	2.1. Falta de permisos en la ubicación de la carpeta seleccionada 2.2. Fin del Programa
POST-CONDICIONES	- Se elimina el xml de la Especificación

Fuente: Elaboración propia

CU - 00023: MOSTRAR LA ESPECIFICACIÓN DE REQUERIMIENTO

Tabla 57. CU – Mostrar la especificación de requerimientos

ESCENARIO DE CASO DE USO 00023			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00023		
NOMBRE	Mostrar la Especificación de Requerimiento		
OBJETIVO	Ver las plantillas de todas las especificación con sus datos ingresados		
DESCRIPCIÓN	El usuario podrá visualizar cada especificación creada y llenada con sus datos. La estructura de visualización de los datos es según las plantillas proporcionada en Diseño. El sistema debe de mostrar automáticamente las especificaciones creadas.		
ACTORES	Usuario, sistema, Interfaz, XML		
RNF RELACIONADOS	Escalabilidad del sistema , Interfaz , Tiempo límite de espera para respuestas del sistema		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

PRE-CONDICIONES	Las especificación de requisitos deben de existir o haber sido creadas
ESCENARIO PRINCIPAL	1. El usuario se sitúa sobre la Especificación que desea ver 2. La interfaz llama a la función mostrar Especificación() 3. El Sistema pide los datos al XML 4. XML extrae los datos de la última versión actual de la Especificación 5. XML envía los datos al sistema 6. El Sistema envía los datos a la Interfaz 7. La Interfaz muestra los datos en pantalla en la ventana "VTEspecificación"
ESCENARIO ALTERNATIVO	-
ESCENARIO DE EXCEPCIÓN	4.1. falta de permisos en la ubicación de la carpeta seleccionada
POST-CONDICIONES	Se elimina el xml de la Especificación

Fuente: Elaboración propia

CU - 00024: VERSIONAR ESPECIFICACIÓN DE REQUERIMIENTO

Tabla 58. CU – Versionar la especificación de requerimientos

ESCENARIO DE CASO DE USO 00023			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00023		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

NOMBRE	Versionar Especificación de Requerimiento
OBJETIVO	Proporcionar la trazabilidad de cada especificación a lo largo de la realización del proyecto
DESCRIPCIÓN	El usuario ingresara la versión de la especificación del requisito; el cual debe de ser incremental y mayor respecto a los anteriores valores de la versión. La versión inicialmente debe ser 1.0. La versión cambiara según el usuario haga sus modificaciones en la especificación. El usuario es que hace el versionamiento.
ACTORES	Usuario, Sistema, Interfaz , XML
RNF RELACIONADOS	Lenguaje , Base de Datos , Fácil mantenimiento , Escalabilidad del sistema , Flexibilidad del sistema , Arquitectura del Computador
PRE-CONDICIONES	El usuario debe de realizar una modificación de una especificación para realizar el versionamiento de dicha especificación
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “modificar Especificación” 2. La Interfaz llama a la función recuperar Especificación() 3. El Sistema pide los datos al XML 4. XML extrae los datos de la última versión actual de la Especificación 5. XML envía un objeto Especificación al sistema 6. El Sistema envía los datos a la Interfaz 7. La Interfaz muestra los datos al Usuario por la ventana “VMEspecificación” 8. El Usuario llena los datos nuevos 9. El usuario presiona el botón “btnVESVersionar” 10. La Interfaz Muestra la pantalla “VVersionarElem” 11. El usuario llena los datos de la versión 12. La Interfaz llama a la función versionarEspecificación () 13. El Sistema recupera la Especificación a Versionar 14. El Sistema agrega la nueva versión 15. El Sistema envía a XML la nueva versión

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	16. XML guarda la nueva versión 17. XML retorna un valor booleano True 18. El Sistema retorna un valor booleano True a la Interfaz 19. La Interfaz Notifica al Usuario que la Especificación ha sido Versionada
ESCENARIO ALTERNATIVO	9.1 El Usuario lleno los datos incorrectamente 9.2 La interfaz notifica al Usuario 9.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	17.1la versión generada debe ser mayor a la anterior 17.2valores nulos en los parámetros de entrada 17.3falta de permisos en la ubicación de la carpeta seleccionada 17.4 FIN
POST-CONDICIONES	Se versiona la Elicitación en el archivo xml Se guardan los valores de la educción actual

Fuente: Elaboración propia

CU - 00025: RESTAURAR ESPECIFICACIÓN DE REQUERIMIENTO

Tabla 59. CU – Restaurar la especificación de requerimientos

ESCENARIO DE CASO DE USO 00025			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00025		
NOMBRE	Restaurar Especificación de Requerimiento		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

OBJETIVO	Poder restaurar el requerimiento a nivel elemento a cualquier versión registrada por el sistema.
DESCRIPCIÓN	Restaurar es una de las opciones dentro de cada requerimiento y es accesible desde las opciones de modificar.
ACTORES	Usuario, sistema, Interfaz, XML.
RNF RELACIONADOS	Flexibilidad , Fácil Mantenimiento , Lenguaje , Base de Datos
PRE-CONDICIONES	El usuario debe de encontrarse dentro del requerimiento a restaurar.
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción "modificar Especificación" 2. La Interfaz llama a la función recuperar Especificación () 3. El Sistema pide los datos al XML 4. XML extrae los datos de la última versión actual de la Especificación 5. XML envía un objeto Especificación al sistema 6. El Sistema envía los datos a la Interfaz 7. La Interfaz muestra los datos al Usuario por la ventana "VMEspecificación" 8. El Usuario llena los datos nuevos 9. El usuario presiona el botón "btnVESVersionar" 10. La Interfaz Muestra la pantalla "VVersionarElem" 11. El usuario llena los datos de la versión 12. La Interfaz llama a la función versionar Especificación() 13. El Sistema recupera la Especificación a Versionar 14. El Sistema agrega la nueva versión 15. El Sistema envía a XML la nueva versión 16. XML guarda la nueva versión 17. XML retorna un valor booleano True 18. El Sistema retorna un valor booleano True a la Interfaz 19. La Interfaz Notifica al Usuario que la Especificación ha sido Versionada

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ESCENARIO ALTERNATIVO	9.1 El Usuario lleno los datos incorrectamente 9.2 La interfaz notifica al Usuario 9.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	17.1la versión generada debe ser mayor a la anterior 17.2valores nulos en los parámetros de entrada 17.3falta de permisos en la ubicación de la carpeta seleccionada 17.4 FIN
POST-CONDICIONES	Se modifica el estado actual del xml por el de la versión seleccionada

Fuente: Elaboración propia

CUE - 00006: GESTION DE REQUERIMIENTO NO FUNCIONAL

CU - 00026: CREAR REQUERIMIENTO NO FUNCIONAL

Tabla 60. CU – Crear requerimiento no funcional

ESCENARIO DE CASO DE USO 00026			
PROYECTO	REMAN		
AUTOR	Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00026		
NOMBRE	Crear Requerimientos no Funcionales		
OBJETIVO	Proporcionar el detalle de un requisito no funcional.		
DESCRIPCIÓN	El usuario registra cada requisito no funcional. Agregando código, nombre, versión, descripción, observaciones, entre otros datos de cada requisito no funcional establecida por el usuario. En esta parte se registrarán los requerimientos del sistema orientados al analista.		
ACTORES	Usuario, Sistema, Interfaz, XML		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

RNF RELACIONADOS	Interfaz , Facilidad de Uso , Fácil mantenimiento , Escalabilidad del sistema
PRE-CONDICIONES	EL Usuario ejecuto el sistema El Usuario debe de estar dentro de un proyecto existente o ya creado
ESCENARIO PRINCIPAL	1. El usuario presiona el botón “btnNoFuncional” 2. La Interfaz muestra la pantalla “VANoFuncional” 3. El usuario llena los datos 4. El usuario presiona el botón “btnAEAgregar” 5. La Interfaz llama a la función crearRequisitoNoFuncional(). 6. El Sistema crea el XML del Requisito No Funcional 7. El Sistema aumenta el contador del Requisito No Funcional 8. El Sistema envía los datos al XML 9. XML guarda los datos del nuevo Requisito No Funcional 10. XML envía notificación de confirmación al Sistema 11. El Sistema envía una notificación de creación a la Interfaz 12. La Interfaz Notifica al Usuario que el Requisito No Funcional ha sido creado
ESCENARIO ALTERNATIVO	3.1 El Usuario lleno los datos incorrectamente 3.2 La interfaz notifica al Usuario 3.3 Se vuelve al Paso 2
ESCENARIO DE EXCEPCIÓN	5.1 Valores nulos en los parámetros de entrada 5.2 Notificación de ERROR 9.1 falta de Permisos en la ubicación de la carpeta seleccionada 9.2 Notificación de ERROR
POST-CONDICIONES	-

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CU - 00027: MODIFICAR REQUERIMIENTO NO FUNCIONAL

Tabla 61. CU – Modificar requerimiento no funcional

ESCENARIO DE CASO DE USO 00027			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00027		
NOMBRE	Modificar Requisito no Funcional		
OBJETIVO	Cambiar los datos de un requisito no funcional		
DESCRIPCIÓN	El usuario puede modificar cualquier campo del requisito no funcional. Siguiendo las condiciones de crear un requisito no funcional		
ACTORES	Usuario, sistema, Interfaz, XML.		
RNF RELACIONADOS	Base de Datos , Lenguaje		
PRE-CONDICIONES	El requisito no funcional debe de estar creado El usuario no puede cambiar el código del requisito no funcional		
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “modificarRequisitoNF” 2. Interfaz llama a la función recuperarRequisitoNF(). 3. XML extrae los datos de la última versión actual del Requisito No Funcional 4. XML envía un objeto tipo Requerimiento No Funcional al Sistema 5. El Sistema envía los datos del objeto Requerimiento No Funcional a la interfaz 6. La interfaz Muestra la pantalla VMNoFuncional. 7. El Usuario llena los datos que desee Actualizar 8. El usuario presiona el botón “btnVNFGuardar” 9. La Interfaz llama a la función modificarRequisitoNF(..)		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	10. El Sistema modifica los datos en la versión actual 11. El Sistema envía los datos a XML 12. XML guarda los nuevos datos en la versión actual 13. XML envía un valor booleano True a la Interfaz 14. La interfaz notifica al Usuario de la modificación realizada
ESCENARIO ALTERNATIVO	7.1 El Usuario lleno los datos incorrectamente 7.2 La interfaz notifica al Usuario 7.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	13.1 El Sistema no puede actualizar el Requerimiento No Funcional 13.2 El Sistema envía notificación a la Interfaz 13.3 La Interfaz Notifica al Usuario 13.4 FIN
POST-CONDICIONES	- Se Modifica el Requerimiento No Funcional actual del xml

Fuente: Elaboración propia

CU - 00028: ELIMINAR REQUERIMIENTO NO FUNCIONAL

Tabla 62. CU – Eliminar requerimiento no funcional

ESCENARIO DE CASO DE USO 00028			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00028		
NOMBRE	Eliminar Requisito no Funcional		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

OBJETIVO	Desechar los requisitos no funcionales que no sirvan para el usuario
DESCRIPCIÓN	El usuario puede eliminar los requisitos no funcionales que no necesite. Al seleccionar la eliminación de un requisito no funcional; el usuario debe de confirmarla realización de la eliminación.
ACTORES	Usuario, Sistema, Interfaz, XML
RNF RELACIONADOS	Interfaz , Facilidad de Uso , Facil mantenimiento , Escalabilidad del sistema
PRE-CONDICIONES	El requisito no funcional debe de existir dentro del proyecto
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “eliminar Requisito No Funcional” 2. La interfaz llama a la función eliminarRequisitoNF() 3. El Sistema retorna un valor booleano True 4. La Interfaz Notifica al usuario sobre la Eliminación
ESCENARIO ALTERNATIVO	
ESCENARIO DE EXCEPCIÓN	2.1. Falta de permisos en la ubicación de la carpeta seleccionada 2.2. Fin del Programa
POST-CONDICIONES	- Se elimina el XML del Requisito No Funcional

Fuente: Elaboración propia

CU - 00029: MOSTRAR REQUERIMIENTO NO FUNCIONAL

Tabla 63. CU – Mostrar requerimiento no funcional

ESCENARIO DE CASO DE USO 00029	
PROYECTO	REMAN
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00029		
NOMBRE	Mostrar Requisito no Funcional		
OBJETIVO	Ver las plantillas de todas los requisito no funcional con sus datos ingresados		
DESCRIPCIÓN	El usuario podrá visualizar cada requisito no funcional creada y llenada con sus datos. La estructura de visualización de los datos es según las plantillas proporcionada en Diseño. El sistema debe de mostrar automáticamente el requisito no funcional creado.		
ACTORES	Usuario, sistema, Interfaz, XML		
RNF RELACIONADOS	Escalabilidad del sistema , Interfaz , Tiempo límite de espera para respuestas del sistema		
PRE-CONDICIONES	Los requisitos no funcionales deben de existir o haber sido creadas		
ESCENARIO PRINCIPAL	1. El usuario se sitúa sobre el Requisito No Funcional que desea ver 2. La interfaz llama a la función mostrarRequisitoNF() 3. El Sistema pide los datos al XML 4. XML extrae los datos de la última versión actual del Requisito No Funcional 5. XML envía los datos al sistema 6. El Sistema envía los datos a la Interfaz 7. La Interfaz muestra los datos en pantalla en la ventana “VTNoFuncional”		
ESCENARIO ALTERNATIVO	-		
ESCENARIO DE EXCEPCIÓN	4.1. falta de permisos en la ubicación de la carpeta seleccionada		
POST-CONDICIONES	Se elimina el xml del Requisito No Funcional		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Fuente: Elaboración propia

CU - 00030: VERSIONAR REQUERIMIENTO NO FUNCIONAL

Tabla 64. CU – Versionar requerimiento no funcional

ESCENARIO DE CASO DE USO 00030			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00030		
NOMBRE	Versionar Requerimiento no Funcional		
OBJETIVO	Proporcionar la trazabilidad de cada requisito no funcional a lo largo de la realización del proyecto		
DESCRIPCIÓN	El usuario ingresara la versión del requisito no funcional; el cual debe de ser incremental y mayor respecto a los anteriores valores de la versión. La versión inicialmente debe ser 1.0. La versión cambiara según el usuario haga sus modificaciones en el requisito no funcional. El usuario es que realiza el versionamiento.		
ACTORES	Usuario, Sistema, Interfaz, XML		
RNF RELACIONADOS	Lenguaje , Base de Datos , Facil mantenimiento , Escalabilidad del sistema , Flexibilidad del sistema , Arquitectura del Computador		
PRE-CONDICIONES	El usuario debe de realizar una modificación de un requisito no funcional para realizar el versionamiento de dicho requisito no funcional		
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “modificarRequisitoNF” 2. La Interfaz llama a la función recuperarRequisitoNF () 3. El Sistema pide los datos al XML		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	4. XML extrae los datos de la última versión actual del Requisito No Funcional 5. XML envía un objeto RequisitoNF al sistema 6. El Sistema envía los datos a la Interfaz 7. La Interfaz muestra los datos al Usuario por la ventana "VMNoFuncional" 8. El Usuario llena los datos nuevos 9. El usuario presiona el botón "btnVNFVersionar" 10. La Interfaz Muestra la pantalla "VVersionarElem" 11. El usuario llena los datos de la versión 12. La Interfaz llama a la función versionarRequisitoNF () 13. El Sistema recupera el Requisito No Funcional a Versionar 14. El Sistema agrega la nueva versión 15. El Sistema envía a XML la nueva versión 16. XML guarda la nueva versión 17. XML retorna un valor booleano True 18. El Sistema retorna un valor booleano True a la Interfaz 19. La Interfaz Notifica al Usuario que el Requisito No Funcional ha sido Versionado
ESCENARIO ALTERNATIVO	9.1 El Usuario lleno los datos incorrectamente 9.2 La interfaz notifica al Usuario 9.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	17.1la versión generada debe ser mayor a la anterior 17.2valores nulos en los parámetros de entrada 17.3falta de permisos en la ubicación de la carpeta seleccionada 17.4 FIN
POST-CONDICIONES	Se versiona la Elicitación en el archivo xml Se guardan los valores de la educción actual

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CU - 00031: RESTAURAR REQUERIMIENTO NO FUNCIONAL

Tabla 65. CU – Restaurar requerimiento no funcional

ESCENARIO DE CASO DE USO 00031			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00031		
NOMBRE	Restaurar Requerimiento no Funcional		
OBJETIVO	Proporcionar la trazabilidad de cada requisito no funcional a lo largo de la realización del proyecto		
DESCRIPCIÓN	El usuario ingresara la versión del requisito no funcional; el cual debe de ser incremental y mayor respecto a los anteriores valores de la versión. La versión inicialmente debe ser 1.0. La versión cambiara según el usuario haga sus modificaciones en el requisito no funcional. El usuario es que realiza el versionamiento.		
ACTORES	Usuario, Sistema, Interfaz, XML		
RNF RELACIONADOS	Flexibilidad , Facil Mantenimiento , Lenguaje , Base de Datos		
PRE-CONDICIONES	El usuario debe de realizar una modificación de un requisito no funcional para realizar el versionamiento de dicho requisito no funcional		
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “modificarRequisitoNF” 2. La Interfaz llama a la función recuperarRequisitoNF () 3. El Sistema pide los datos al XML 4. XML extrae los datos de la últi.ma versión actual del Requerimiento No Funcional. 5. XML envía un objeto Requerimiento No Funcional al sistema 6. El Sistema envía los datos a la Interfaz		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	7. La Interfaz muestra los datos al Usuario por la ventana "VRestaura" 8. El Usuario llena los datos nuevos 9. El usuario presiona el botón "btnRestaurar" 10. La Interfaz Muestra la pantalla "VVersionarElem" 11. El usuario llena los datos de la versión 12. La Interfaz llama a la función versionarRequerimiento No Funcional () 13. El Sistema recupera el Requerimiento No Funcional a Versionar 14. El Sistema agrega la nueva versión 15. El Sistema envía a XML la nueva versión 16. XML guarda la nueva versión 17. XML retorna un valor booleano True 18. El Sistema retorna un valor booleano True a la Interfaz 19. La Interfaz Notifica al Usuario que el Requerimiento No Funcional ha sido Versionado
ESCENARIO ALTERNATIVO	9.1 El Usuario lleno los datos incorrectamente 9.2 La interfaz notifica al Usuario 9.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	17.1la versión generada debe ser mayor a la anterior 17.2valores nulos en los parámetros de entrada 17.3falta de permisos en la ubicación de la carpeta seleccionada 17.4 FIN
POST-CONDICIONES	Se modifica el estado actual del xml por el de la versión seleccionada

Fuente: Elaboración propia

CUE - 00007: GESTION DE ACTOR FUENTE

CU - 00032: CREAR ACTOR FUENTE

Tabla 66. CU – Crear actor fuente

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ESCENARIO DE CASO DE USO 00032			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00032		
NOMBRE	Agregar Actor Fuente		
OBJETIVO	Proporcionar detalles de las personas que participan como proveedores de información acerca del proyecto a realizarse		
DESCRIPCIÓN	El usuario podrá registrar a todas las personas que están involucradas en la organización para la cual se realiza el proyecto. El actor fuente es el que provee información acerca de los datos y procedimientos que el usuario del sistema debe de plasmar en las educciones, elicitaciones y especificaciones. El usuario podrá ingresar los datos generales del actor fuente		
ACTORES	Usuario, sistema, Interfaz, XML		
RNF RELACIONADOS	Interfaz , Facilidad de Uso , Fácil mantenimiento , Escalabilidad del sistema		
PRE-CONDICIONES	EL Usuario ejecuto el sistema		
ESCENARIO PRINCIPAL	1. El usuario presiona el botón “btnVPFuente” 2. La Interfaz muestra la pantalla “VFuente 3. El usuario llena los datos 4. El usuario presiona el botón “btnFGuardar” 5. La Interfaz llama a la función crearStakeholder(). 6. El Sistema crea el XML del actor fuente 7. El Sistema aumenta el contador del actor fuente 8. El Sistema envía los datos al XML 9. XML guarda los datos del actor fuente 10. XML envía notificación de confirmación al Sistema		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	11. El Sistema envía una notificación de creación a la Interfaz 12. La Interfaz Notifica al Usuario que el actor fuente ha sido creado
ESCENARIO ALTERNATIVO	3.1 El Usuario lleno los datos incorrectamente 3.2 La interfaz notifica al Usuario 3.3 Se vuelve al Paso 2
ESCENARIO DE EXCEPCIÓN	5.1 Valores nulos en los parámetros de entrada 5.2 Notificación de ERROR 9.1 falta de Permisos en la ubicación de la carpeta seleccionada 9.2 Notificación de ERROR
POST-CONDICIONES	-

Fuente: Elaboración propia

CU - 00033: MODIFICAR ACTOR FUENTE

Tabla 67. CU – Modificar actor fuente

ESCENARIO DE CASO DE USO 00033			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00033		
NOMBRE	Modificar Actor Fuente		
OBJETIVO	Cambiar los datos de un actor Fuente		
DESCRIPCIÓN	El usuario puede modificar cualquier campo de un actor fuente. siguiendo las condiciones de crear un actor fuente		
ACTORES	Usuario, sistema		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

RNF RELACIONADOS	Base de Datos , Lenguaje
PRE-CONDICIONES	EL actor debe de estar creado El usuario no puede cambiar el código del actor
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “modificar Actor Fuente” btnVPFuente 2. Interfaz llama a la función recuperarStakeholder(). 3. XML extrae los datos de la última versión actual de la Elicitación 4. XML envía un objeto tipo Actor Fuente al Sistema 5. El Sistema envía los datos del objeto Actor Fuente a la interfaz 6. La interfaz Muestra la pantalla VMFuente 7. El Usuario llena los datos que desee Actualizar 8. El usuario presiona el botón “btnVFGuardar” 9. La Interfaz llama a la función modificarActorFuente(..) 10. El Sistema modifica los datos en la versión actual 11. El Sistema envía los datos a XML 12. XML guarda los nuevos datos en la versión actual 13. XML envía un valor booleano True a la Interfaz 14. La interfaz notifica al Usuario de la modificación realizada
ESCENARIO ALTERNATIVO	7.1 El Usuario lleno los datos incorrectamente 7.2 La interfaz notifica al Usuario 7.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	13.1 El Sistema no puede actualizar El Actor Fuente 13.2 El Sistema envía notificación a la Interfaz 13.3 La Interfaz Notifica al Usuario 13.4 FIN
POST-CONDICIONES	- Se Modifica la Actor Fuente actual del xml

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CU - 00034: ELIMINAR ACTOR FUENTE

Tabla 68. CU – Eliminar actor fuente

ESCENARIO DE CASO DE USO 00034			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00035		
NOMBRE	Eliminar Actor Fuente		
OBJETIVO	Desechar las actores fuente que no sirvan para el usuario		
DESCRIPCIÓN	El usuario puede eliminar los actores fuente que no necesite. Al seleccionar la eliminación de un actor fuente; el usuario debe de confirmarla realización de la eliminación.		
ACTORES	Usuario, Sistema, Interfaz, Sistema		
RNF RELACIONADOS	Interfaz , Facilidad de Uso , Facil mantenimiento , Escalabilidad del sistema		
PRE-CONDICIONES	El Actor Fuente debe de existir		
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “eliminar Actor Fuente” 2. La interfaz llama a la función eliminarStakeholder() 3. El Sistema retorna un valor booleano True 4. La Interfaz Notifica al usuario sobre la Eliminación		
ESCENARIO ALTERNATIVO			
ESCENARIO DE EXCEPCIÓN	2.1. Falta de permisos en la ubicación de la carpeta seleccionada 2.2. Fin del Programa		
POST-CONDICIONES	- Se elimina el xml del Actor Fuente		

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CU - 00035: MOSTRAR ACTOR FUENTE

Tabla 69. CU – Mostrar actor fuente

ESCENARIO DE CASO DE USO 00035			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00035		
NOMBRE	Mostrar actor Fuente		
OBJETIVO	Ver las plantillas de todas los actores fuente con sus datos ingresados		
DESCRIPCIÓN	El usuario podrá visualizar cada actor fuente creado y llenado con sus datos. La estructura de visualización de los datos es según las plantillas proporcionada en Diseño. El sistema debe de mostrar automáticamente los actores fuente creados.		
ACTORES	Usuario, sistema		
RNF RELACIONADOS	Escalabilidad del sistema , Interfaz , Tiempo límite de espera para respuestas del sistema		
PRE-CONDICIONES	Los actores deben de existir o haber sido creadas		
ESCENARIO PRINCIPAL	1. El usuario se sitúa sobre el Actor Fuente que desea ver 2. La interfaz llama a la función mostrarStakeHolder() 3. El Sistema pide los datos al XML 4. XML extrae los datos de la última versión actual del Actor Fuente 5. XML envía los datos al sistema 6. El Sistema envía los datos a la Interfaz		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	7. La Interfaz muestra los datos en pantalla en la ventana "VTOrganizacion"
ESCENARIO ALTERNATIVO	-
ESCENARIO DE EXCEPCIÓN	4.1. falta de permisos en la ubicación de la carpeta seleccionada
POST-CONDICIONES	Se elimina el xml del Actor Fuente

Fuente: Elaboración propia

CUE - 00008: GESTION DE ACTOR ESPECIALISTA

CU - 00036: CREAR ACTOR ESPECIALISTA

Tabla 70. CU – Crear actor especialista

ESCENARIO DE CASO DE USO 00036			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00036		
NOMBRE	Agregar actor Especialista		
OBJETIVO	Proporcionar detalles de las personas que participan de la realización del proyecto de software		
DESCRIPCIÓN	El usuario podrá registrar a todas las personas que están involucradas en la organización que está realizando el proyecto. El actor especialista es el que recibe la información y realiza las educciones, elicitaciones y especificaciones. También estos		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	actores son asignados a proyectos. El usuario podrá registrar los datos generales de cada actor especialista.
ACTORES	Usuario, sistema
RNF RELACIONADOS	Interfaz , Facilidad de Uso , Facil mantenimiento , Escalabilidad del sistema
PRE-CONDICIONES	EL Usuario ejecuto el sistema
ESCENARIO PRINCIPAL	1. El usuario presiona el botón “btnVPEspecialista” 2. La Interfaz muestra la pantalla “VEspecialista” 3. El usuario llena los datos 4. El usuario presiona el botón “btnEGuardar” 5. La Interfaz llama a la función crearTeamProject(). 6. El Sistema crea el XML del actor Especialista 7. El Sistema aumenta el contador del actor Especialista 8. El Sistema envía los datos al XML 9. XML guarda los datos del actor Especialista 10. XML envía notificación de confirmación al Sistema 11. El Sistema envía una notificación de creación a la Interfaz 12. La Interfaz Notifica al Usuario que el actor Especialista ha sido creado
ESCENARIO ALTERNATIVO	3.1 El Usuario lleno los datos incorrectamente 3.2 La interfaz notifica al Usuario 3.3 Se vuelve al Paso 2
ESCENARIO DE EXCEPCIÓN	5.1 Valores nulos en los parámetros de entrada 5.2 Notificación de ERROR 9.1 falta de Permisos en la ubicación de la carpeta seleccionada 9.2 Notificación de ERROR
POST-CONDICIONES	-

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CU - 00037: MODIFICAR ACTOR ESPECIALISTA

Tabla 71. CU – Modificar actor especialista

ESCENARIO DE CASO DE USO 00038			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00037		
NOMBRE	Modificar Actor Especialista		
OBJETIVO	Cambiar los datos de un actor especialista		
DESCRIPCIÓN	El usuario puede modificar cualquier campo de un actor especialista. Siguiendo las condiciones de crear un actor especialista		
ACTORES	Usuario, sistema		
RNF RELACIONADOS	Base de Datos , Lenguaje		
PRE-CONDICIONES	EL actor debe de estar creado El usuario no puede cambiar el código del actor		
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “modificar Actor Especialista” btnVP Especialista 2. Interfaz llama a la función recuperarTeamProject(). 3. XML extrae los datos de la última versión actual del actor Especialista 4. XML envía un objeto tipo Actor Especialista al Sistema 5. El Sistema envía los datos del objeto Actor Especialista a la interfaz 6. La interfaz Muestra la pantalla VMFuente 7. El Usuario llena los datos que desee Actualizar 8. El usuario presiona el botón “btnVFGuardar”		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	9. La Interfaz llama a la función modificarActor Especialista (..) 10. El Sistema modifica los datos en la versión actual 11. El Sistema envía los datos a XML 12. XML guarda los nuevos datos en la versión actual 13. XML envía un valor booleano True a la Interfaz 14. La interfaz notifica al Usuario de la modificación realizada
ESCENARIO ALTERNATIVO	7.1 El Usuario lleno los datos incorrectamente 7.2 La interfaz notifica al Usuario 7.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	13.1 El Sistema no puede actualizar El Actor Especialista 13.2 El Sistema envía notificación a la Interfaz 13.3 La Interfaz Notifica al Usuario 13.4 FIN
POST-CONDICIONES	- Se Modifican la Actor Fuente actual del xml

Fuente: Elaboración propia

CU - 00038: ELIMINAR ACTOR ESPECIALISTA

Tabla 72. CU – Eliminar actor especialista

ESCENARIO DE CASO DE USO 00039			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.1	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00038		
NOMBRE	Eliminar Actor Especialista		
OBJETIVO	Desechar las actores especialista que no sirvan para el usuario		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

DESCRIPCIÓN	El usuario puede eliminar al actor especialista que no necesite. Al seleccionar la eliminación de un actor especialista; el usuario debe de confirmarla realización de la eliminación.
ACTORES	Usuario, Sistema, Interfaz
RNF RELACIONADOS	Interfaz , Facilidad de Uso , Facil mantenimiento , Escalabilidad del sistema
PRE-CONDICIONES	El Actor especialista debe de existir
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción "eliminar Actor Especialista" 2. La interfaz llama a la función eliminarTeamProject() 3. El Sistema retorna un valor booleano True 4. La Interfaz Notifica al usuario sobre la Eliminación
ESCENARIO ALTERNATIVO	
ESCENARIO DE EXCEPCIÓN	2.1. Falta de permisos en la ubicación de la carpeta seleccionada 2.2. Fin del Programa
POST-CONDICIONES	- Se elimina el xml del Actor Especialista

Fuente: Elaboración propia

CU - 00039: MOSTRAR ACTOR ESPECIALISTA

Tabla 73. CU – Mostrar actor especialista

ESCENARIO DE CASO DE USO 00040			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00039		
NOMBRE	Mostrar Actor Especialista		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

OBJETIVO	Ver las plantillas de todas los actores especialista con sus datos ingresados
DESCRIPCIÓN	El usuario podrá visualizar cada actor especialista creado y llenado con sus datos. La estructura de visualización de los datos es según las plantillas proporcionada en Diseño. El sistema debe de mostrar automáticamente los actores especialistas creados.
ACTORES	Usuario, sistema, Interfaz, XML
RNF RELACIONADOS:	Escalabilidad del sistema , Interfaz , Tiempo límite de espera para respuestas del sistema
PRE-CONDICIONES	Los actores especialistas deben de existir o haber sido creadas
ESCENARIO PRINCIPAL	1. El usuario se sitúa sobre el Actor Especialista que desea ver 2. La interfaz llama a la función mostrarTeamProject() 3. El Sistema pide los datos al XML 4. XML extrae los datos de la última versión actual del Actor Especialista 5. XML envía los datos al sistema 6. El Sistema envía los datos a la Interfaz 7. La Interfaz muestra los datos en pantalla en la ventana “VTOrganizacion”
ESCENARIO ALTERNATIVO	-
ESCENARIO DE EXCEPCIÓN	4.1. Falta de permisos en la ubicación de la carpeta seleccionada
POST-CONDICIONES	Se elimina el xml del Actor Especialista

Fuente: Elaboración propia

CUE - 00009: GESTION DE ORGANIZACIÓN

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CU - 00040: CREAR ORGANIZACIÓN

Tabla 74. CU – Crear organización

ESCENARIO DE CASO DE USO 00040			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00040		
NOMBRE	CREAR Organización		
OBJETIVO	Proporcionar el detalle de una organización		
DESCRIPCIÓN	El usuario registra las organizaciones las cuales participaran en un proyecto. El usuario ingresará los datos de la organización.		
ACTORES	Usuario, Sistema, Interfaz, XML		
RNF RELACIONADOS	Interfaz , Facilidad de Uso , Fácil mantenimiento , Escalabilidad del sistema		
PRE-CONDICIONES	EL Usuario ejecuto el sistema Los actores deben de estar creados		
ESCENARIO PRINCIPAL	1. El usuario presiona el botón “btnVPOrganizacion” 2. La Interfaz muestra la pantalla “VOrganizacion” 3. El usuario llena los datos 4. El usuario presiona el botón “btnOGuardar” 5. La Interfaz llama a la función crearOrganizacion (). 6. El Sistema crea el XML de la Organización 7. El Sistema aumenta el contador de la Organización 8. El Sistema envía los datos al XML 9. XML guarda los datos de la Organización 10. XML envía notificación de confirmación al Sistema 11. El Sistema envía una notificación de creación a la Interfaz 12. La Interfaz Notifica al Usuario la Organización ha sido creada		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ESCENARIO ALTERNATIVO	3.1 El Usuario lleno los datos incorrectamente 3.2 La interfaz notifica al Usuario 3.3 Se vuelve al Paso 2
ESCENARIO DE EXCEPCIÓN	5.1 Valores nulos en los parámetros de entrada 5.2 Notificación de ERROR 9.1 falta de Permisos en la ubicación de la carpeta seleccionada 9.2 Notificación de ERROR
POST-CONDICIONES	-

Fuente: Elaboración propia

CU - 00041: MODIFICAR ORGANIZACIÓN

Tabla 75. CU – Modificar organización

ESCENARIO DE CASO DE USO 00041			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00041		
NOMBRE	Modificar Organización		
OBJETIVO	Cambiar los datos de una organización		
DESCRIPCIÓN	El usuario puede modificar cualquier campo de una organización. Siguiendo las condiciones de crear una organización		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ACTORES	Usuario, sistema, Interfaz, XML
RNF RELACIONADOS	Base de Datos , Lenguaje
PRE-CONDICIONES	EL Usuario ejecuto el sistema Los actores deben de estar creados
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción "modificar Organización" btnVPOrganización 2. Interfaz llama a la función recuperar Organización (). 3. XML extrae los datos de la última versión actual de la Organización 4. XML envía un objeto tipo Organización al Sistema 5. El Sistema envía los datos del objeto Organización a la interfaz 6. La interfaz Muestra la pantalla VM Organización 7. El Usuario llena los datos que desee Actualizar 8. El usuario presiona el botón "btnVFGuardar" 9. La Interfaz llama a la función modificar Organización(..) 10. El Sistema modifica los datos en la versión actual 11. El Sistema envía los datos a XML 12. XML guarda los nuevos datos en la versión actual 13. XML envía un valor booleano True a la Interfaz La interfaz notifica al Usuario de la modificación realizada
ESCENARIO ALTERNATIVO	7.1 El Usuario lleno los datos incorrectamente 7.2 La interfaz notifica al Usuario 7.3 Se vuelve al Paso 6
ESCENARIO DE EXCEPCIÓN	-13.1 El Sistema no puede actualizar la Organización 13.2 El Sistema envía notificación a la Interfaz 13.3 La Interfaz Notifica al Usuario 13.4 FIN
POST-CONDICIONES	Se Modifica la Actor Fuente actual del xml

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CU - 00042: ELIMINAR ORGANIZACIÓN

Tabla 76. CU – Eliminar organización

ESCENARIO DE CASO DE USO 00043			
PROYECTO	REMAN		
AUTOR	Quispe Huamán, Kevin Percy		
VERSIÓN	0.1	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00043		
NOMBRE	Eliminar Organización		
OBJETIVO	Desechar las organizaciones que no sirvan para el usuario		
DESCRIPCIÓN	El usuario puede eliminar las organizaciones que no necesite. Al seleccionar la eliminación de una organización; el usuario debe de confirmarla realización de la eliminación.		
ACTORES	Usuario, Sistema		
RNF RELACIONADOS	Interfaz , Facilidad de Uso , Facil mantenimiento , Escalabilidad del sistema		
PRE-CONDICIONES	La organización debe de existir o debe de haberse creado		
ESCENARIO PRINCIPAL	1. El usuario selecciona la opción “eliminar Organización” 2. La interfaz llama a la función eliminar Organización () 3. El Sistema retorna un valor booleano True 4. La Interfaz Notifica al usuario sobre la Eliminación		
ESCENARIO ALTERNATIVO			
ESCENARIO DE EXCEPCIÓN	2.1. Falta de permisos en la ubicación de la carpeta seleccionada 2.2. Fin del Programa		
POST-CONDICIONES	- Se elimina el xml de la Organización		

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CU - 00043: MOSTRAR LA ORGANIZACIÓN

Tabla 77. CU – Mostrar organización

ESCENARIO DE CASO DE USO 00043			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00043		
NOMBRE	Mostrar Organización		
OBJETIVO	Ver las plantillas de todas las organizaciones con sus datos ingresados		
DESCRIPCIÓN	El usuario podrá visualizar cada organización creada y llenada con sus datos. La estructura de visualización de los datos es según las plantillas proporcionada en Diseño. El sistema debe de mostrar automáticamente la platilla de la organización creada.		
ACTORES	Usuario, sistema, Interfaz, XML		
RNF RELACIONADOS	Escalabilidad del sistema , Interfaz , Tiempo límite de espera para respuestas del sistema		
PRE-CONDICIONES	Las Organizaciones deben de existir o haber sido creadas		
ESCENARIO PRINCIPAL	1. El usuario se sitúa sobre la Organización que desea ver 2. La interfaz llama a la función mostrarOrganización() 3. El Sistema pide los datos al XML 4. XML extrae los datos de la última versión actual de la Organización 5. XML envía los datos al sistema 6. El Sistema envía los datos a la Interfaz 7. La Interfaz muestra los datos en pantalla en la ventana “VTOrganizacion”		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ESCENARIO ALTERNATIVO	-
ESCENARIO DE EXCEPCIÓN	4.1. falta de permisos en la ubicación de la carpeta seleccionada
POST-CONDICIONES	Se elimina el xml de la Organización

Fuente: Elaboración propia

CUE - 00010: GESTIÓN DE HISTORIAL DE LIBROS

CU - 00044: VERSIONAR LIBRO

Tabla 78. CU – Versionar libro

ESCENARIO DE CASO DE USO 00011			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00044		
NOMBRE	Versionar Libro de Educción de Requerimiento		
OBJETIVO	Proporcionar la trazabilidad del libro en general con todos sus requerimientos a lo largo de la realización del proyecto.		
DESCRIPCIÓN	El usuario ingresara la versión del libro; el cual debe de ser incremental y mayor respecto a los anteriores valores de la versión. La versión inicialmente debe ser 0.1. La versión		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	cambiara según el usuario haga sus modificaciones en la educción. El usuario es el que hace el versionamiento.
ACTORES	Usuario, Sistema, Interfaz, XML
RNF RELACIONADOS:	Lenguaje , Base de Datos , Facil mantenimiento , Escalabilidad del sistema , Flexibilidad del sistema , Arquitectura del Computador
PRE-CONDICIONES	El usuario debe estar dentro del proyecto con los libro a restaurar.
ESCENARIO PRINCIPAL	1. El usuario selecciona El Libro 2. El Usuario llena los datos acerca de la nueva Versión 3. El usuario presiona el Botón Guardar 4. La Interfaz envía los datos al sistema 5. El Sistema agrega la nueva versión 6. El Sistema envía a XML la nueva versión 7. El Sistema crea una nueva carpeta con el número de versión 8. El Sistema copia los archivos a la carpeta creada 9. XML guarda la nueva versión 10. El Sistema envía una notificación a la Interfaz 11. La Interfaz Notifica al Usuario que el Libro ha sido Versionada
ESCENARIO ALTERNATIVO	2.1 El Usuario lleno los datos incorrectamente 2.2 La interfaz notifica al Usuario 2.3 Se vuelve al Paso 1
ESCENARIO DE EXCEPCIÓN	7.1 El Sistema no puede crear la Carpeta 7.2 El Sistema envía notificación a la Interfaz 7.3 La Interfaz Notifica al Usuario 7.4 FIN
POST-CONDICIONES	-

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CU - 00045: RESTAURAR LIBRO

Tabla 79. CU – Restaurar libro

ESCENARIO DE CASO DE USO 00045			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00045		
NOMBRE	Restaurar Libro		
OBJETIVO	Proporcionar la trazabilidad del libro en general con todos sus requerimientos a lo largo de la realización del proyecto.		
DESCRIPCIÓN	El usuario ingresara la versión del libro; el cual debe de ser incremental y mayor respecto a los anteriores valores de la versión. La versión inicialmente debe ser 0.1. La versión cambiara según el usuario haga sus modificaciones en la educción. El usuario es el que hace el versionamiento.		
ACTORES	Usuario, Sistema, Interfaz, XML		
RNF RELACIONADOS:	Facilidad de Uso , Facil mantenimiento , Escalabilidad del sistema		
PRE-CONDICIONES	El usuario este ubicado en la interfaz del Sistema Versionar Libro El usuario haya seleccionado el libro que desea Restaurar		
ESCENARIO PRINCIPAL	El usuario selecciona la versión del Libro que desea obtener de la tabla histórico El usuario presiona el botón Restaurar La Interfaz solicita envía los datos al Sistema El Sistema elimina el contenido de la versión actual del libro El sistema Carga los datos de la versión que se quiere restaurar El Sistema copia los archivos de la versión restaurada a la carpeta actual del libro XML guarda la nueva versión		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

	El Sistema envía una notificación a la Interfaz La Interfaz Notifica al Usuario que el Libro ha sido restaurado
ESCENARIO ALTERNATIVO	-
ESCENARIO DE EXCEPCIÓN	-
POST-CONDICIONES	- Se versiona el libro elegido en el xml

Fuente: Elaboración propia

CUE - 00011: GESTION DE REPORTES [RESUMEN]

CU - 00046: EXPORTAR LIBRO

Tabla 80. CU – Exportar libro

ESCENARIO DE CASO DE USO 00046			
PROYECTO	REMAN		
AUTOR	Montero Nifla Max Dionicio Quispe Huamán, Kevin Percy		
VERSIÓN	0.3	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00046		
NOMBRE	Exportar Libro		
OBJETIVO	Exportar un libro seleccionado en formato pdf o en Word.		
DESCRIPCIÓN	Este caso de uso tiene como objeto el poder exportar un libro determinado en diferentes formatos		
ACTORES	Usuario, sistema		
RNF RELACIONADOS	Interfaz , Base de Datos , Arquitectura del Computador		
PRE-CONDICIONES	Estar ubicado en el libro deseado		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ESCENARIO PRINCIPAL	1. El Usuario presiona el botón "Exportar" 2. El Usuario Selecciona el Libro para ser exportado 3. El Usuario llena los datos necesarios para la exportación 4. La interfaz envía los datos al Sistema 5. El Sistema envía los datos al XML 6. XML carga los datos al Sistema 7. El Sistema Exporta los datos en un Formato determinado 8. El Sistema notifica A la Interfaz 9. La interfaz Notifica que el Libro ha sido exportado satisfactoriamente
ESCENARIO ALTERNATIVO	2.1 El usuario selecciona en formato WORD
ESCENARIO DE EXCEPCIÓN	6.1 EL sistema no puede cargar los datos debido a permisos de la carpeta 6.2 Notificación de error
POST-CONDICIONES	Se exporta un determinado Libro

Fuente: Elaboración propia

CU - 00047: GENERAR REPORTE

Tabla 81. CU – Generar reporte

ESCENARIO DE CASO DE USO 00047			
PROYECTO	REMAN		
AUTOR	Quispe Huamán, Kevin Percy		
VERSIÓN	0.1	Estado de desarrollo	Borrador/Revisado/Finalizado
DEFINICIÓN DEL CASO DE USO			
CÓDIGO	CU-00047		
NOMBRE	Generar Reporte		
OBJETIVO	Generación de reportes		

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

DESCRIPCIÓN	Generación de reportes
ACTORES	Sistema , Usuario
RNF RELACIONADOS	Tiempo límite de espera para casos excepcionales
PRE-CONDICIONES	Control del tiempo límite excepcional para cargar la información
ESCENARIO PRINCIPAL	El sistema debe ser capaz de crear reportes
ESCENARIO ALTERNATIVO	
ESCENARIO DE EXCEPCIÓN	
POST-CONDICIONES	Mostrar el reporte generado Regresar a la lista

Fuente: Elaboración propia

4.2.2.2. Empleo de casos de uso

Los casos de uso son artefactos de software que se caracterizan por proporcionar información sólida que conduce al entendimiento real del problema en investigación. En ella se explica con detalle el comportamiento y funcionalidades del producto a construir, así como los casos de excepción a tomar en cuenta.

Estos permitirán al desarrollador una visión cabal del modelo de negocio asumir en la construcción del producto final.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

- CASO GENERAL

La composición general del producto de software a construir permite establecer que los requerimientos no funcionales, en la etapa de educción, quedan definidos por la cantidad de proyectos que se desean abrir al mismo tiempo, así como la cantidad de información que se debe de almacenar por cada uno de ellos. Bajo este ángulo de vista permite dimensionar la base de datos correspondiente.

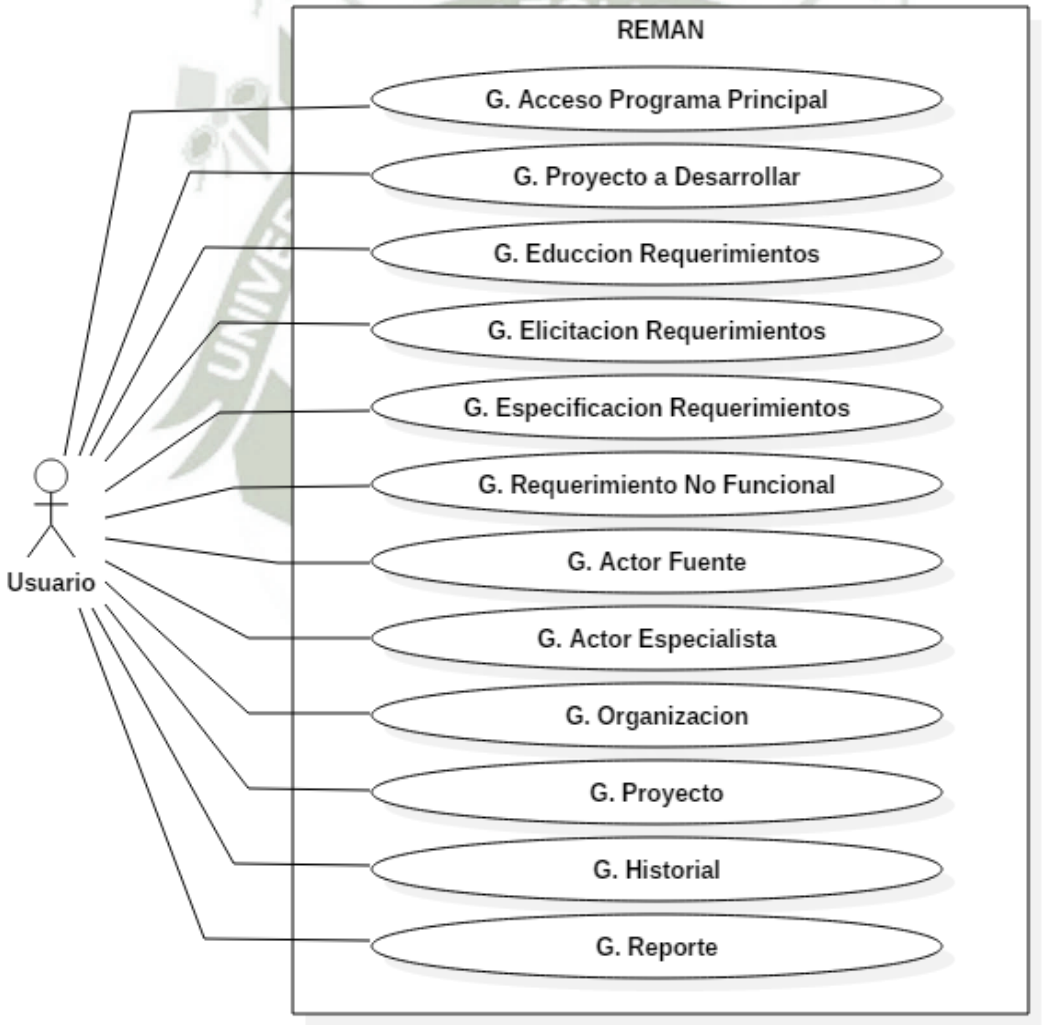


Figura 12. Caso de uso del producto REMAN
Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

- GESTION DE PROGRAMA PRINCIPAL

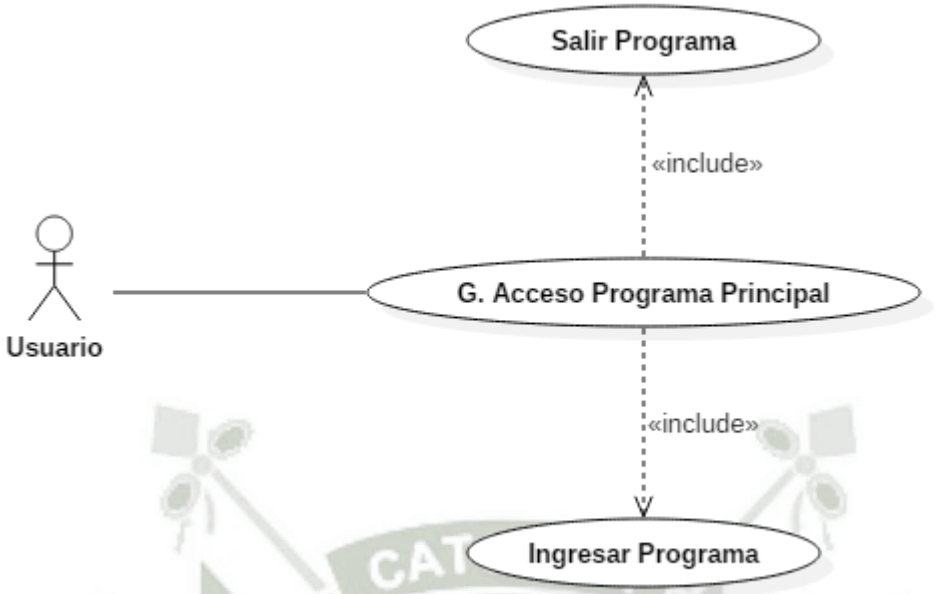


Figura 13. Caso de uso gestión del programa principal
Fuente: Elaboración propia

La figura 13 permite determinar que los niveles de seguridad del producto de software deben estar administrados por la persona encargada del producto. Esta persona es la encargada de administrar la seguridad del producto.

- GESTION DE PROYECTO A DESARROLLAR

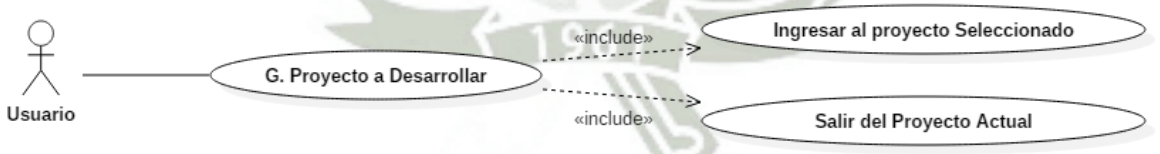


Figura 14. Caso de uso gestión de proyecto a desarrollar
Fuente: Elaboración propia

La figura 14 permite determinar que la gestión del proyecto debe incluir carpetas dedicadas a proyectos específicos con la finalidad de mantener la información de cada uno de ellos.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

- GESTION DE EDUCCION DE REQUERIMIENTOS

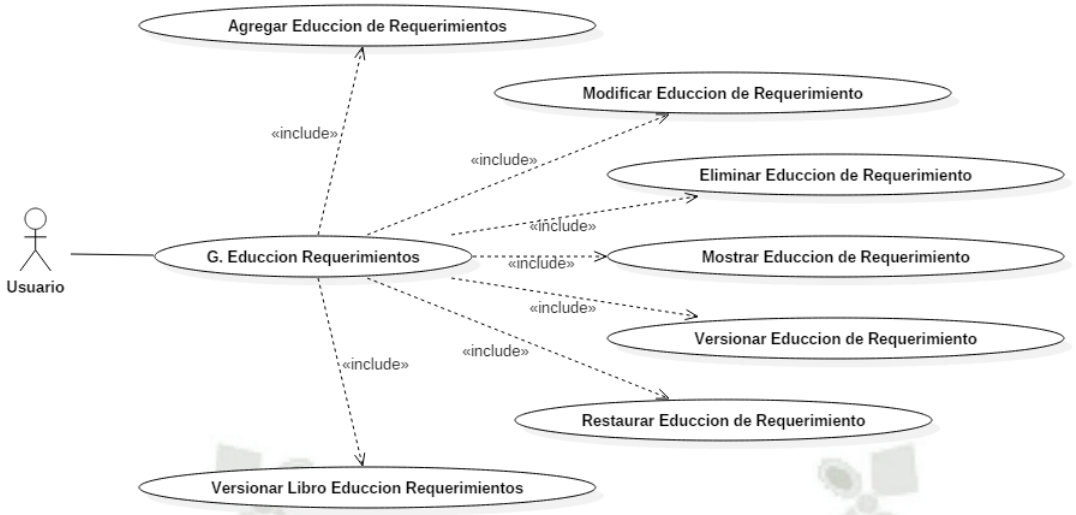


Figura 15. Caso de uso gestión de educción de requerimientos
Fuente: Elaboración propia

La figura 15 hace mención o asocia a los siguientes requisitos no funcionales:

- Como mantener el historial de información de cada proyecto seleccionado
- Como administrar, de manera certera, la base de datos para conjuntos de proyectos

- GESTION DE ELICITACIÓN DE REQUERIMIENTOS

La figura 16 muestra que el caso de uso denominado “Elicitación Requerimientos” contiene a un conjunto de actividades por lo que se emplea el “include”. No es un “extend” porque estas actividades no son una extensión del proceso.

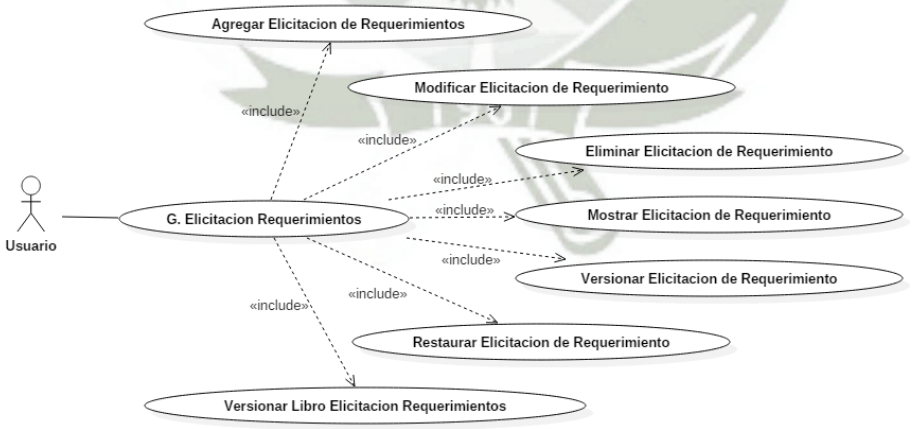


Figura 16. Caso de uso gestión de elicitación de requerimientos
Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

La figura 16 asocia los siguientes requisitos no funcionales:

- Como mantener la ruta de información de los requisitos educcionados relacionados con los requisitos elicitados
- La cantidad de información que deben administrar los libros
- Como mantener las tablas de las bases de datos asociadas: abiertas o cerradas al momento de la lectura

- GESTION DE ESPECIFICACIÓN DE REQUERIMIENTOS

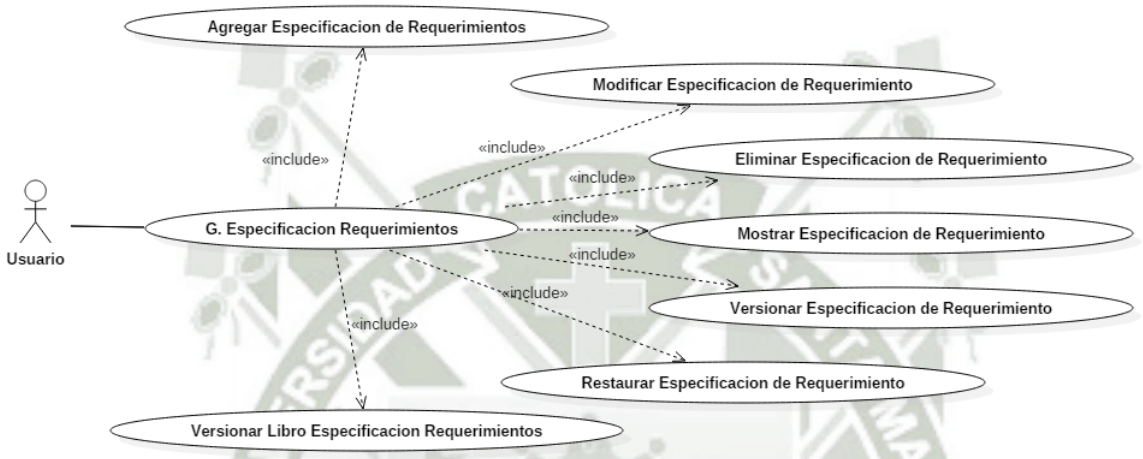


Figura 17. Caso de uso gestión de especificación de requerimientos
Fuente: Elaboración propia

Requisitos no funcionales asociados: ídem al de la figura 17 pero respecto a la especificación de requisitos de software no funcionales.

La especificación de requerimientos se gestiona bajo tres grandes ejes:

- La agregación de especificación de requerimientos
- La gestión de especificaciones de requerimientos
- El versionamiento del libro de especificación de requerimientos

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

- GESTION DE REQUERIMIENTO NO FUNCIONAL

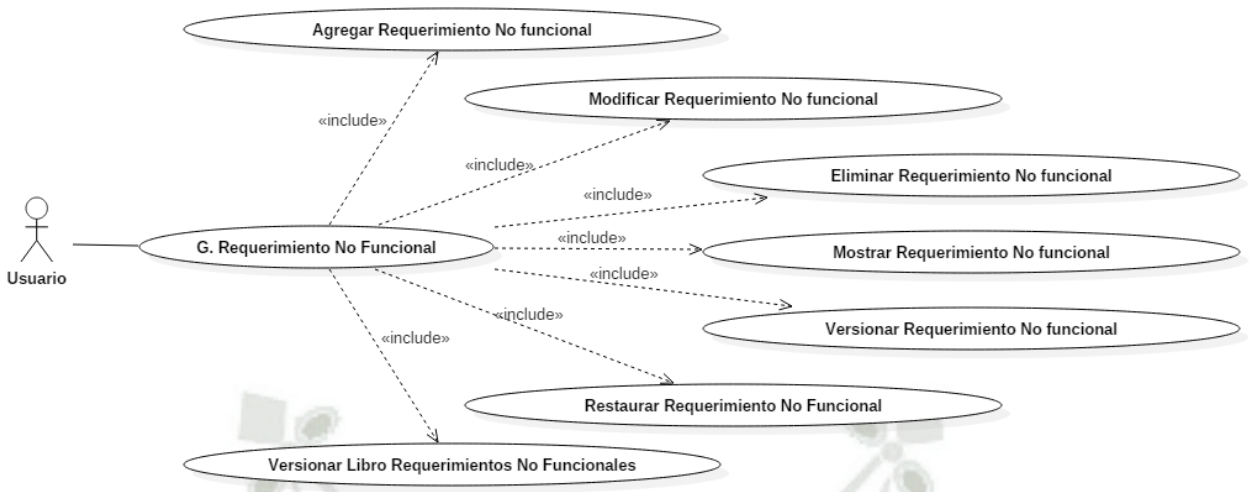


Figura 18. Caso de uso gestión de requerimiento no funcional
Fuente: Elaboración propia

Los requisitos no funcionales asociados son los mismos expuestos en el ítem 14 pero asociados explícitamente a los requerimientos no funcionales del producto.

- GESTION DE ACTORES FUENTE

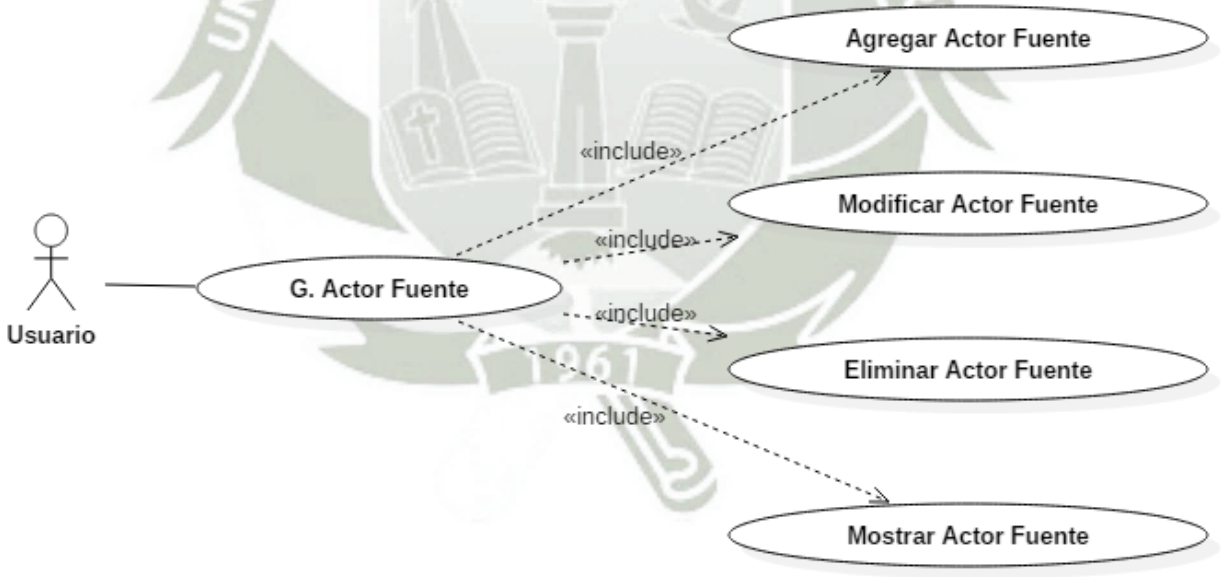


Figura 19. Caso de uso gestión de actores
Fuente: Elaboración propia

En esta figura se nota claramente que los requisitos no funcionales asociados son los posibles cambios de roles de los actores.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

- GESTION DE ACTORES ESPECIALISTAS

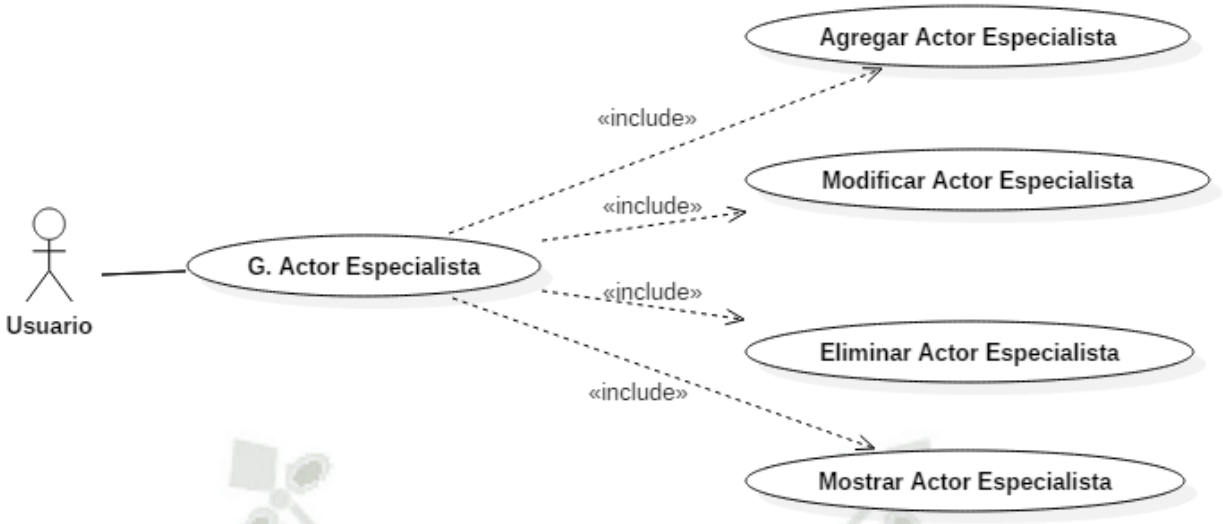


Figura 20. Caso de uso gestión de actores especialistas
Fuente: Elaboración propia

Su descripción es idéntica al de la figura 19.

- GESTION DE ORGANIZACIÓN

*Figura 21.*Caso de uso gestión de organización
Fuente: Elaboración propia

Los requisitos no funcionales asociados son:

- El comportamiento de la base de datos cuando se inserta o elimina una organización

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

- Los campos en las tablas cuando las organizaciones quedan definidas

- GESTION DE PROYECTO

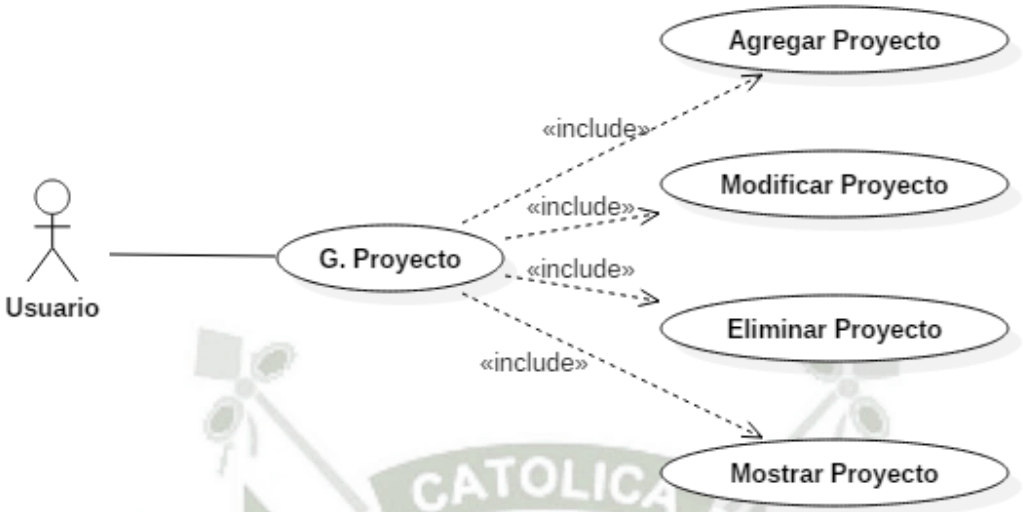


Figura 22. Caso de uso gestión de proyecto
Fuente: Elaboración propia

Los requisitos no funcionales son idénticos a los de la figura 21 con la excepción de que se aplican a los proyectos.

- GESTION DE HISTORIAL

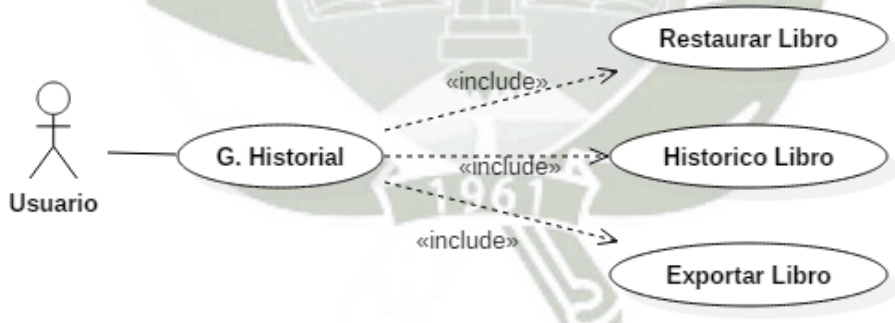


Figura 23. Caso de uso gestión de historial
Fuente: Elaboración propia

Los requisitos no funcionales asociados son:

- Navegar en los discos duros para extraer la información correspondiente a un determinado proyecto u organización

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

- En el caso de haber realizado una escritura en el disco duro, como rescatar la información correcta

- GESTION DE REPORTES

Figura 24. Caso de uso gestión de reportes
Fuente: Elaboración propia

Los reportes tienen inmersos requisitos no funcionales como el estado y calidad de la impresora.

La arquitectura para la visualización de datos se presenta en la figura 25.

Figura 25. Arquitectura de la visualización de datos
Fuente: Elaboración propia

4.2.3. Análisis de enlaces entre requisitos no funcionales

Del análisis de las diferentes etapas del modelo de asociación se nota claramente que un cambio en la arquitectura de software del producto (por ejemplo, pasar de un modelo de computadora personal a un modelo distribuido) puede generar una seria repercusión en la arquitectura de software y por lo tanto en los interfaces de usuario. Lo mismo ocurre con el problema de la escalabilidad quien afecta directamente en la arquitectura del producto.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Los conflictos entre requisitos no funcionales son detectados tal como se especifica en la figura 26 y sus dependencias en la figura 27.

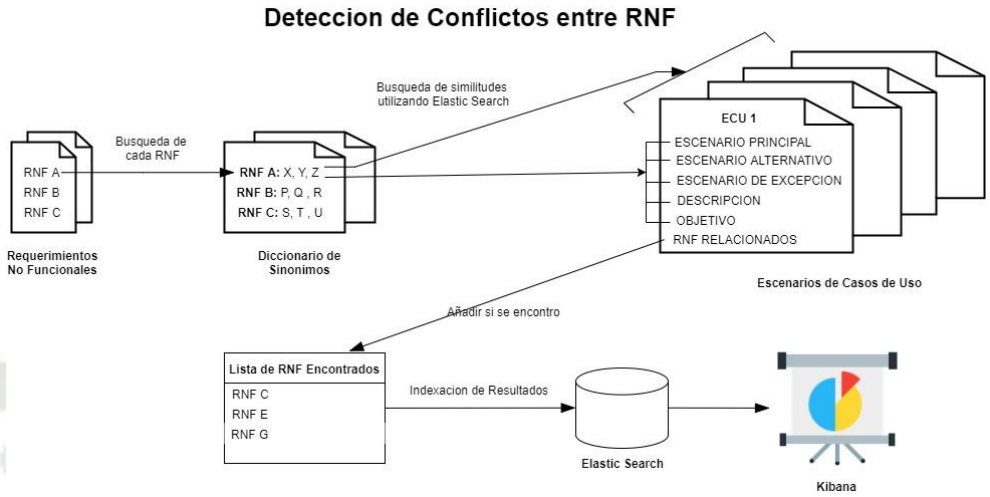


Figura 26. Detección de conflictos entre RNF
Fuente: Elaboración propia

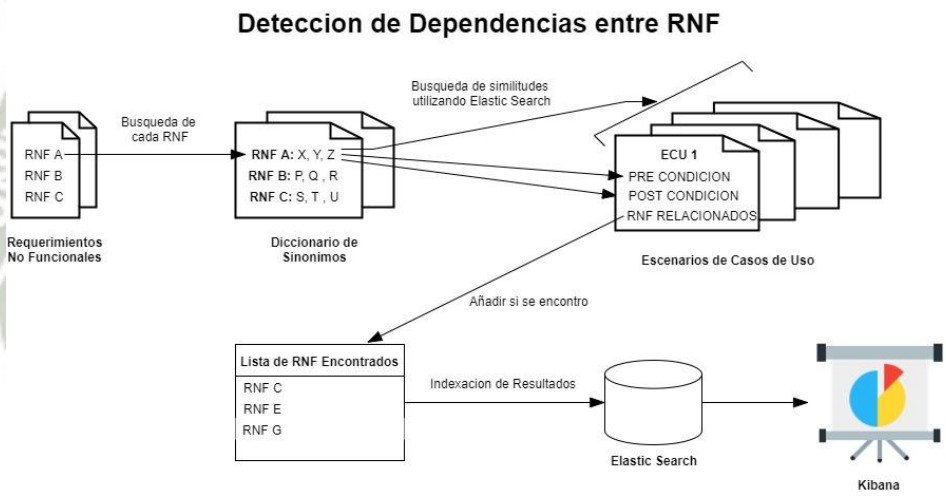


Figura 27. Detección de dependencias entre RNF
Fuente: Elaboración propia

4.2.4. Operaciones realizadas con Elastic Search

4.2.4.1. Búsqueda para la detección dependencias

Para la búsqueda de dependencias, la consulta se enfoca en la PreCondicion y la PostCondicion del escenario de caso de uso, utilizando como patrón el diccionario de sinónimos de los requerimientos no funcionales. Si la búsqueda es satisfactoria retorna el campo del escenario de caso de uso donde se encuentra los requerimientos no funcionales relacionados.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

```
var searchResponse = elasticClient.Search<IndexScenarioUseCase>(s => s
                .Source(sf => sf
                  .Includes(i => i
                    .Fields(
                      f => f.NfrRelation)))
                .Index("scenarios")
                .Query(q => q
                  .Match(m => m
                    .Field(f => f.PostCondition.ToLower())
                    .Query(synonym.ToLower())
                  ) || q
                  .Match(m => m
                    .Field(f => f.PreCondition.ToLower())
                    .Query(synonym.ToLower()))));
```

4.2.4.2. Búsqueda para la detección de conflictos

Para la búsqueda de los conflictos, la consulta se enfoca en el escenario Principal, escenario alternativo, escenario excepcional, el objetivo y la descripción del escenario de caso de uso, utilizando como patrón el diccionario de sinónimos de los requerimientos no funcionales. Si la búsqueda es satisfactoria retorna el campo del escenario de caso de uso donde se encuentra los requerimientos no funcionales relacionados.

```
var searchResponse = elasticClient.Search<IndexScenarioUseCase>(s => s
                .Source(sf => sf
                  .Includes(i => i
                    .Fields(
                      f => f.NfrRelation
                    )
                  )
                )
                .Index("scenarios")
                .AllTypes()
```


Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

```
.Query(q => q
    .Match(m => m
        .Field(f => f.PrincipalScenario.ToLower())
        .Query(synonym.ToLower())
        ) || q
    .Match(m => m
        .Field(f => f.AlternativeScenario.ToLower())
        .Query(synonym.ToLower())
    ) || q
    .Match(m => m
        .Field(f => f.ExceptionScenario.ToLower())
        .Query(synonym.ToLower())
    ) || q
    .Match(m => m
        .Field(f => f.Objective.ToLower())
        .Query(synonym.ToLower())
    ) || q
    .Match(m => m
        .Field(f => f.Description.ToLower())
        .Query(synonym.ToLower())
    )
)
);
```

4.2.4.3. Indexaciones

Elasticsearch utiliza índices tal, así como si fuesen tablas en una base de datos. Para la creación de estos índices utiliza Mapping dando como entrada una clase, en este caso serian 4: Escenario de Caso de Uso, Requerimiento No Funcional, Dependencias y Conflictos. Luego se indexará toda la información de los requerimientos no funcionales y los escenarios a los índices de Requerimientos no Funcionales y Escenarios de Casos de Uso respectivamente. Finalmente, luego del análisis se indexarán los resultados a los índices de Dependencias y Conflictos.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

1. Creación del índice para los Requerimientos no funcionales

```
createIndexResponse = elasticClient.CreateIndex("nfrs", c => c
        .Settings(s => s
            .NumberOfShards(0)
            .NumberOfReplicas(0)
        )
        .Mappings(m => m
            .Map<IndexNfr>(d => d
                .AutoMap()
            )
        )
    );
```

2. Creación del índice para los Escenarios de Casos de Uso

```
var createIndexResponse = elasticClient.CreateIndex("scenarios", c => c
        .Settings(s => s
            .NumberOfShards(0)
            .NumberOfReplicas(0)
        )
        .Mappings(m => m
            .Map<IndexScenarioUseCase>(d => d
                .AutoMap()
            )
        )
    );
```

3. Creación del índice para las Dependencias

```
var createIndexResponse = elasticClient.CreateIndex("dependencies", c => c
        .Settings(s => s
            .NumberOfShards(0)
            .NumberOfReplicas(0)
        )
```


Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

```
        .Mappings(m => m
            .Map<IndexDependencyResult>(d => d
                .AutoMap()
            )
        )
    );
```

4. Creación del índice para los Conflictos

```
var createIndexResponse = elasticClient.CreateIndex("conflicts", c => c
        .Settings(s => s
            .NumberOfShards(0)
            .NumberOfReplicas(0)
        )
        .Mappings(m => m
            .Map<IndexConflictResult>(d => d
                .AutoMap()
            )
        )
    );
```

5. Indexación de datos al índice de Requerimientos no funcionales

```
foreach (var indexNfr in indexNfrList)
        {
            elasticClient.Index(indexNfr, i => i
                .Index("nfrs")
                .Id(indexNfr.NfrId)
            );
        }
```

6. Indexación de datos al índice de Escenarios de Casos de Uso

```
foreach (var scenarioUseCase in indexScenarioUseCaseList)
        {
            elasticClient.Index(scenarioUseCase, i => i
                .Index("scenarios")
```


Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

```
            .Id(scenarioUseCase.ScenarioUseCaseId)
        );
    }
```

7. Indexación de resultados al índice de Dependencias

```
foreach (var dependencyResult in dependencyList)
{
        elasticClient.Index(dependencyResult, i => i
            .Index("dependencies")
            .Id(counter)
        );
}
```

8. Indexación de resultados al índice de Conflictos

```
foreach (var conflictResult in conflictList)
{
        elasticClient.Index(conflictResult, i => i
            .Index("conflicts")
            .Id(counter)
        );
}
```

4.2.5. Resultados encontrados

Los resultados obtenidos por medio de las herramientas propuestas, referentes a la cantidad de conflictos y dependencias encontradas se muestran a continuación (tabla 87 y tabla 88):

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Tabla 82. Conflictos entre requisitos no funcionales

RNF A ID	RNF A	RNF B ID	RNF B	Proyecto
RNF-0001-EL	Facilidad de Uso	RNF-0002-EL	Fácil mantenimiento	REMAN
RNF-0001-EL	Facilidad de Uso	RNF-0003-EL	Escalabilidad del sistema	REMAN
RNF-0001-EL	Facilidad de Uso	RNF-0004-EL	Flexibilidad del sistema	REMAN
RNF-0001-EL	Facilidad de Uso	RNF-0006-EL	Tiempo límite de espera para respuestas del sistema	REMAN
RNF-0001-EL	Facilidad de Uso	RNF-0011-EL	Lenguaje	REMAN
RNF-0001-EL	Facilidad de Uso	RNF-0013-EL	Interfaz	REMAN
RNF-0002-EL	Fácil mantenimiento	RNF-0003-EL	Escalabilidad del sistema	REMAN
RNF-0002-EL	Fácil mantenimiento	RNF-0004-EL	Flexibilidad del sistema	REMAN
RNF-0002-EL	Fácil mantenimiento	RNF-0011-EL	Lenguaje	REMAN
RNF-0002-EL	Fácil mantenimiento	RNF-0012-EL	Base de Datos	REMAN
RNF-0002-EL	Fácil mantenimiento	RNF-0014-EL	Arquitectura del Computador	REMAN
RNF-0002-EL	Fácil mantenimiento	RNF-0001-EL	Facilidad de Uso	REMAN
RNF-0003-EL	Escalabilidad del sistema	RNF-0001-EL	Facilidad de Uso	REMAN
RNF-0003-EL	Escalabilidad del sistema	RNF-0002-EL	Fácil mantenimiento	REMAN
RNF-0003-EL	Escalabilidad del sistema	RNF-0004-EL	Flexibilidad del sistema	REMAN
RNF-0003-EL	Escalabilidad del sistema	RNF-0011-EL	Lenguaje	REMAN
RNF-0003-EL	Escalabilidad del sistema	RNF-0012-EL	Base de Datos	REMAN
RNF-0003-EL	Escalabilidad del sistema	RNF-0014-EL	Arquitectura del Computador	REMAN
RNF-0004-EL	Flexibilidad del sistema	RNF-0003-EL	Escalabilidad del sistema	REMAN
RNF-0004-EL	Flexibilidad del sistema	RNF-0006-EL	Tiempo límite de espera para respuestas del sistema	REMAN
RNF-0004-EL	Flexibilidad del sistema	RNF-0013-EL	Interfaz	REMAN
RNF-0005-EL	Número mínimo de usuarios concurrentes	RNF-0003-EL	Escalabilidad del sistema	REMAN
RNF-0005-EL	Número mínimo de usuarios concurrentes	RNF-0006-EL	Tiempo límite de espera para respuestas del sistema	REMAN
RNF-0005-EL	Número mínimo de usuarios concurrentes	RNF-0013-EL	Interfaz	REMAN

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

RNF-0008-EL	Autentificación para el uso del sistema	RNF-0006-EL	Tiempo límite de espera para respuestas del sistema	REMAN
RNF-0008-EL	Autentificación para el uso del sistema	RNF-0007-EL	Tiempo límite de espera para casos excepcionales	REMAN
RNF-0008-EL	Autentificación para el uso del sistema	RNF-0009-EL	Número máximo de autentificaciones fallidas	REMAN
RNF-0008-EL	Autentificación para el uso del sistema	RNF-0010-EL	Canal seguro de autentificación	REMAN
RNF-0008-EL	Autentificación para el uso del sistema	RNF-0015-EL	Autorización para el uso del sistema	REMAN
RNF-0011-EL	Lenguaje	RNF-0012-EL	Base de Datos	REMAN
RNF-0011-EL	Lenguaje	RNF-0013-EL	Interfaz	REMAN
RNF-0011-EL	Lenguaje	RNF-0014-EL	Arquitectura del Computador	REMAN

Fuente: Elaboración propia

Tabla 83. Dependencias entre requisitos no funcionales

RNF Padre ID	**RNF Padre**	**RNF Hijo ID**	**RNF Hijo**	**Proyecto**
RNF-0001-EL	Facilidad de Uso	RNF-0012-EL	Base de Datos	REMAN
RNF-0001-EL	Facilidad de Uso	RNF-0013-EL	Interfaz	REMAN
RNF-0001-EL	Facilidad de Uso	RNF-0014-EL	Arquitectura del Computador	REMAN
RNF-0002-EL	Fácil mantenimiento	RNF-0004-EL	Flexibilidad del sistema	REMAN
RNF-0002-EL	Fácil mantenimiento	RNF-0011-EL	Lenguaje	REMAN
RNF-0002-EL	Fácil mantenimiento	RNF-0012-EL	Base de Datos	REMAN
RNF-0003-EL	Escalabilidad del sistema	RNF-0001-EL	Facilidad de Uso	REMAN
RNF-0003-EL	Escalabilidad del sistema	RNF-0002-EL	Fácil mantenimiento	REMAN
RNF-0003-EL	Escalabilidad del sistema	RNF-0004-EL	Flexibilidad del sistema	REMAN
RNF-0003-EL	Escalabilidad del sistema	RNF-0011-EL	Lenguaje	REMAN
RNF-0003-EL	Escalabilidad del sistema	RNF-0012-EL	Base de Datos	REMAN
RNF-0003-EL	Escalabilidad del sistema	RNF-0014-EL	Arquitectura del Computador	REMAN
RNF-0004-EL	Flexibilidad del sistema	RNF-0011-EL	Lenguaje	REMAN
RNF-0004-EL	Flexibilidad del sistema	RNF-0012-EL	Base de Datos	REMAN
RNF-0005-EL	Número mínimo de usuarios concurrentes	RNF-0006-EL	Tiempo límite de espera para respuestas del sistema	REMAN

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

RNF-0005-EL	Número mínimo de usuarios concurrentes	RNF-0007-EL	Tiempo límite de espera para casos excepcionales	REMAN
RNF-0005-EL	Número mínimo de usuarios concurrentes	RNF-0008-EL	Autentificación para el uso del sistema	REMAN
RNF-0005-EL	Número mínimo de usuarios concurrentes	RNF-0009-EL	Número máximo de autentificaciones fallidas	REMAN
RNF-0005-EL	Número mínimo de usuarios concurrentes	RNF-0010-EL	Canal seguro de autentificación	REMAN
RNF-0005-EL	Número mínimo de usuarios concurrentes	RNF-0015-EL	Autorización para el uso del sistema	REMAN
RNF-0006-EL	Tiempo límite de espera para respuestas del sistema	RNF-0007-EL	Tiempo límite de espera para casos excepcionales	REMAN
RNF-0008-EL	Autentificación para el uso del sistema	RNF-0004-EL	Flexibilidad del sistema	REMAN
RNF-0008-EL	Autentificación para el uso del sistema	RNF-0009-EL	Número máximo de autentificaciones fallidas	REMAN
RNF-0008-EL	Autentificación para el uso del sistema	RNF-0010-EL	Canal seguro de autentificación	REMAN
RNF-0008-EL	Autentificación para el uso del sistema	RNF-0006-EL	Tiempo límite de espera para respuestas del sistema	REMAN
RNF-0008-EL	Autentificación para el uso del sistema	RNF-0007-EL	Tiempo límite de espera para casos excepcionales	REMAN
RNF-0008-EL	Autentificación para el uso del sistema	RNF-0015-EL	Autorización para el uso del sistema	REMAN
RNF-0015-EL	Autorización para el uso del sistema	RNF-0004-EL	Flexibilidad del sistema	REMAN
RNF-0015-EL	Autorización para el uso del sistema	RNF-0008-EL	Autentificación para el uso del sistema	REMAN
RNF-0015-EL	Autorización para el uso del sistema	RNF-0009-EL	Número máximo de autentificaciones fallidas	REMAN
RNF-0015-EL	Autorización para el uso del sistema	RNF-0010-EL	Canal seguro de autentificación	REMAN

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

4.2.6. Resultados con Kibana

Ya con los resultados con ElasticSearch se procede a obtener diferentes visualizaciones de los datos con Kibana.

4.2.6.1. Creación de los índices

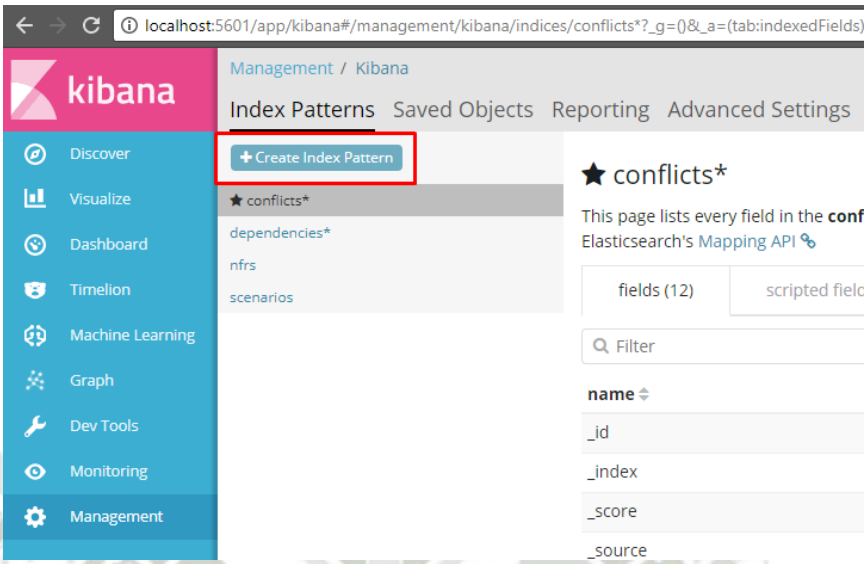


Figura 28. Creación de los índices
Fuente: Elaboración propia

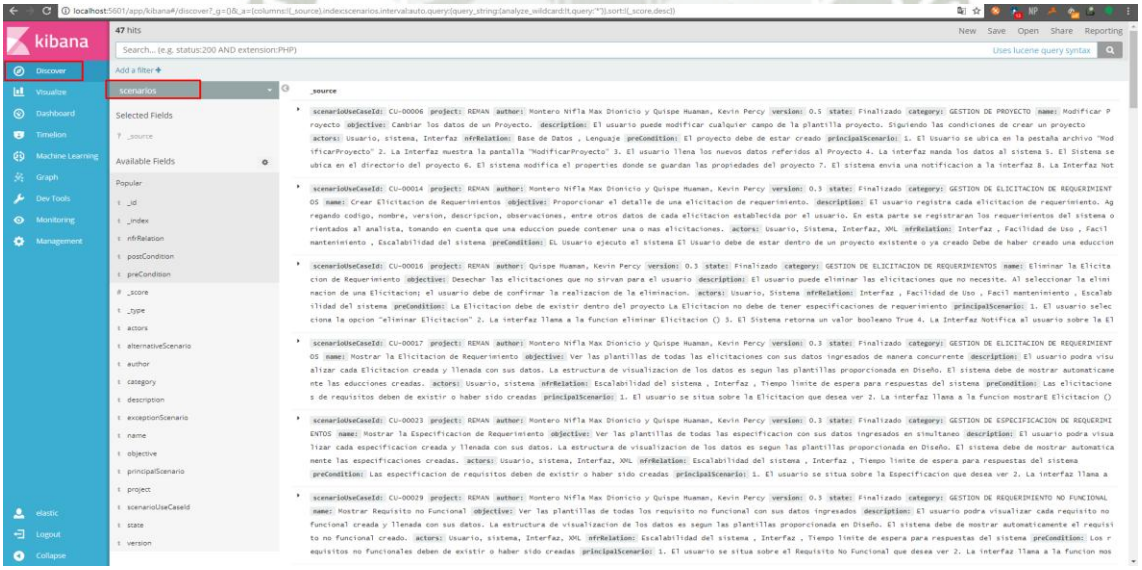


Figura 29. Escenarios de Casos de uso indexados por ElasticSearch
Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

4.2.6.2. Visualización escrita de los datos indexados

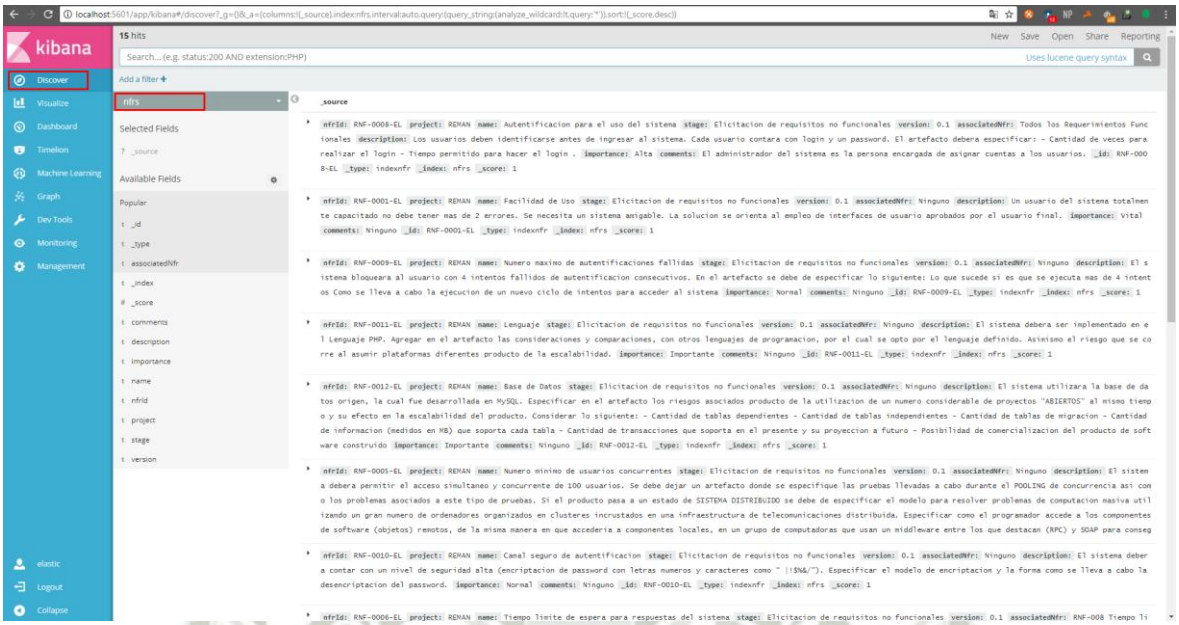


Figura 30. Requerimientos no Funcionales indexados por ElasticSearch
Fuente: Elaboración propia

4.2.6.3. Visualización gráfica de los datos indexados

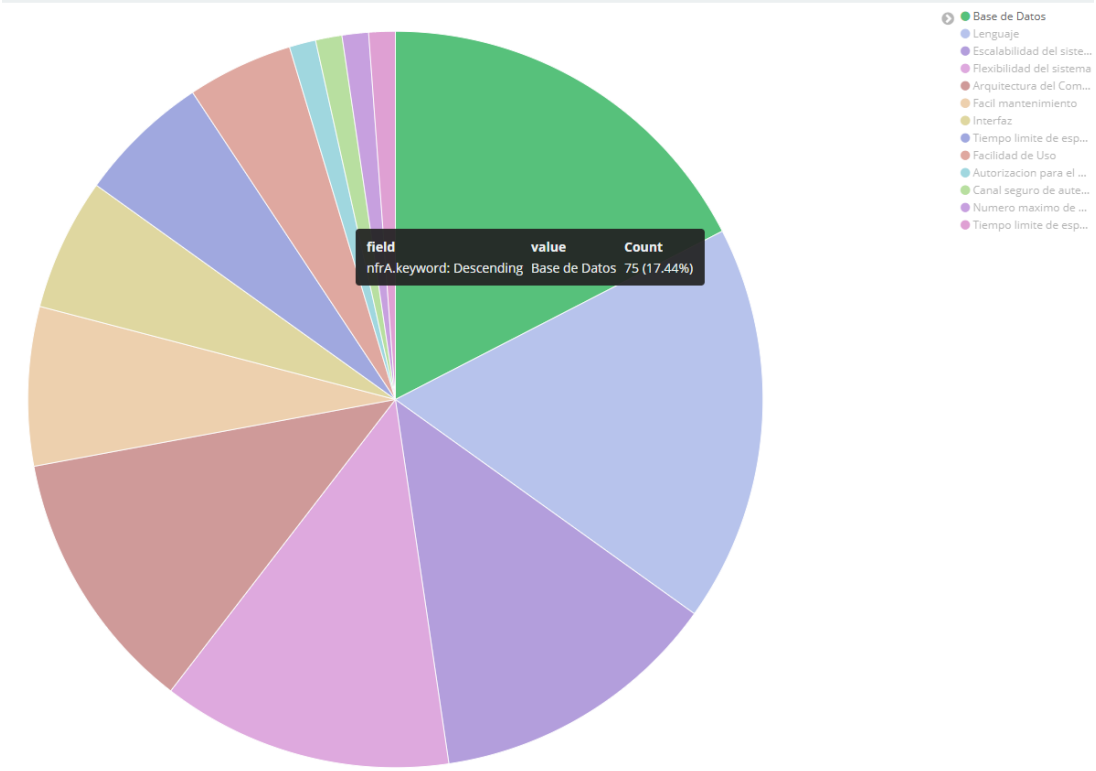


Figura 31. Pie Chart sobre la cantidad de conflictos encontrados
Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

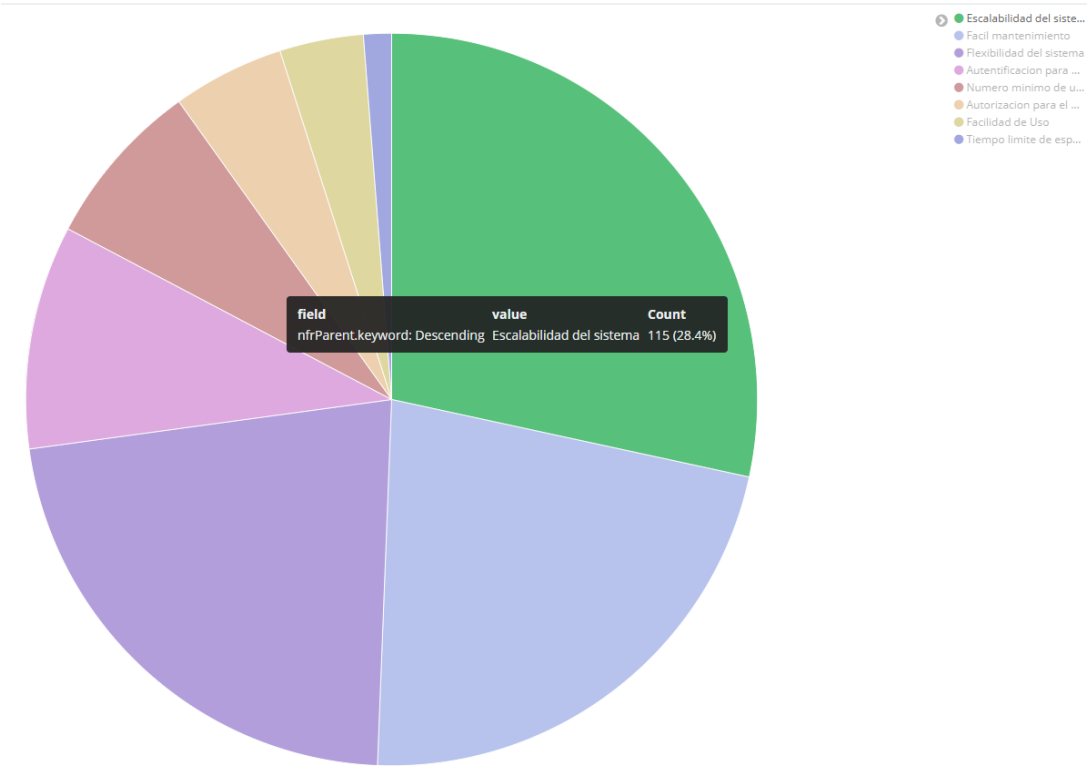


Figura 32. Pie Chart sobre la cantidad de Dependencias Padre
Fuente: Elaboración propia

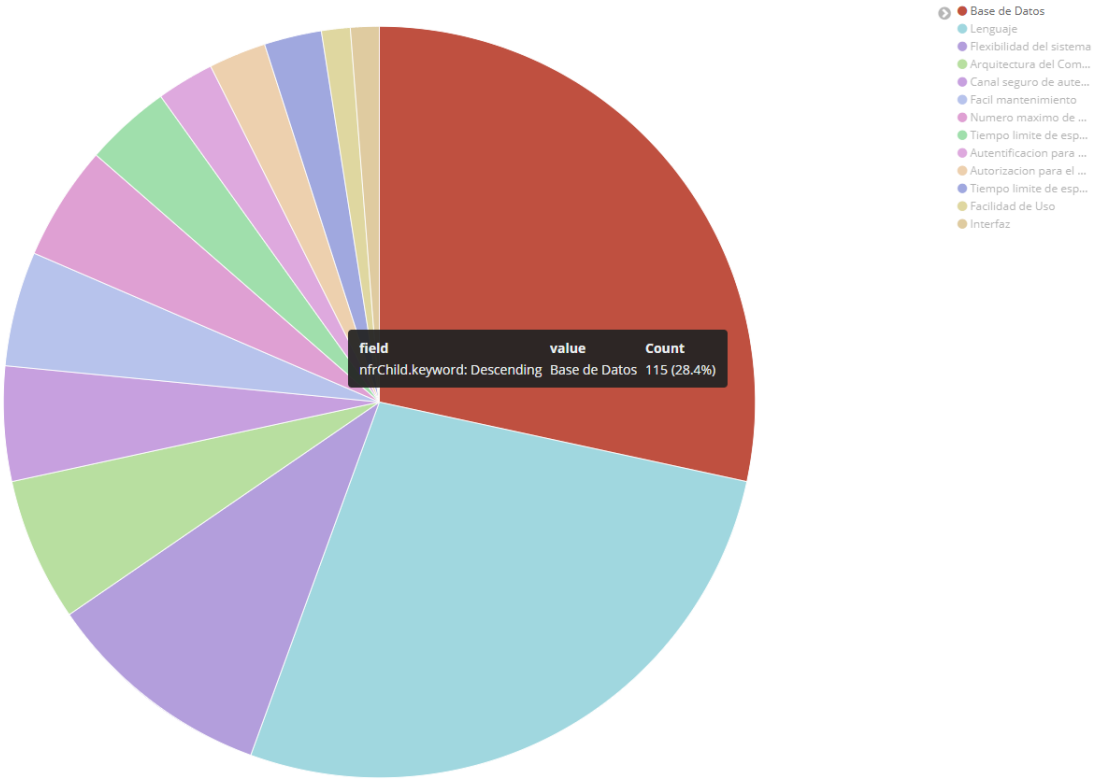


Figura 33. Pie Chart sobre la cantidad de Dependencias Hijo
Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

4.2.6.4. Dashboard

Se pueden añadir múltiple visualizaciónes en un solo dashboard, en la siguiente figura se muestran las visualizaciones sobre la cantidad de dependencias tanto padre como hijo y los conflictos del lado A y del lado B

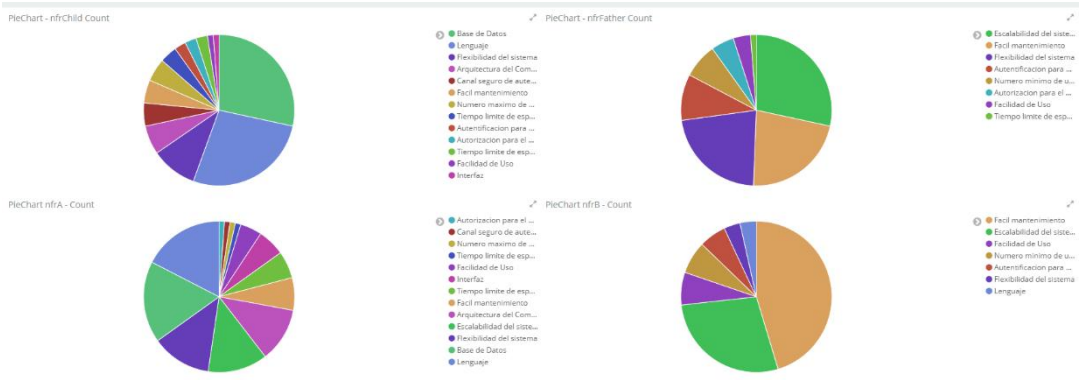


Figura 34. *Pie Chart sobre la cantidad de Dependencias y Conflictos*
Fuente: Elaboración propia

4.2.6.5. Visualización de Grafos

En la siguiente figura se muestra la relación de conflictos entre los requerimientos no funcionales que se tomaron en cuenta con REMAN, en donde la relación conflictiva es más grave cuando la relación de uno a otro es más gruesa.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

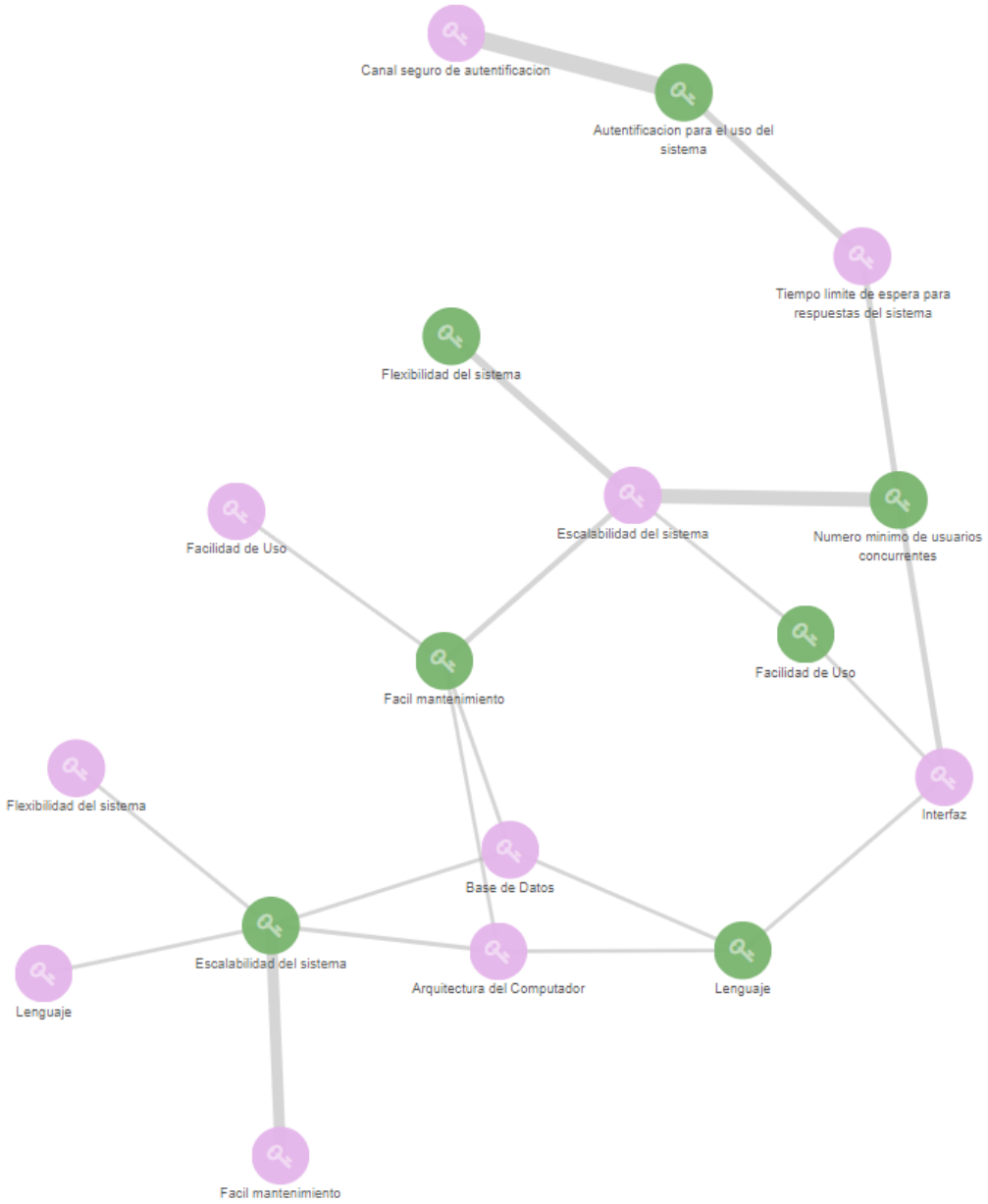


Figura 35. Grafo de relaciones en cuanto a conflictos
Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

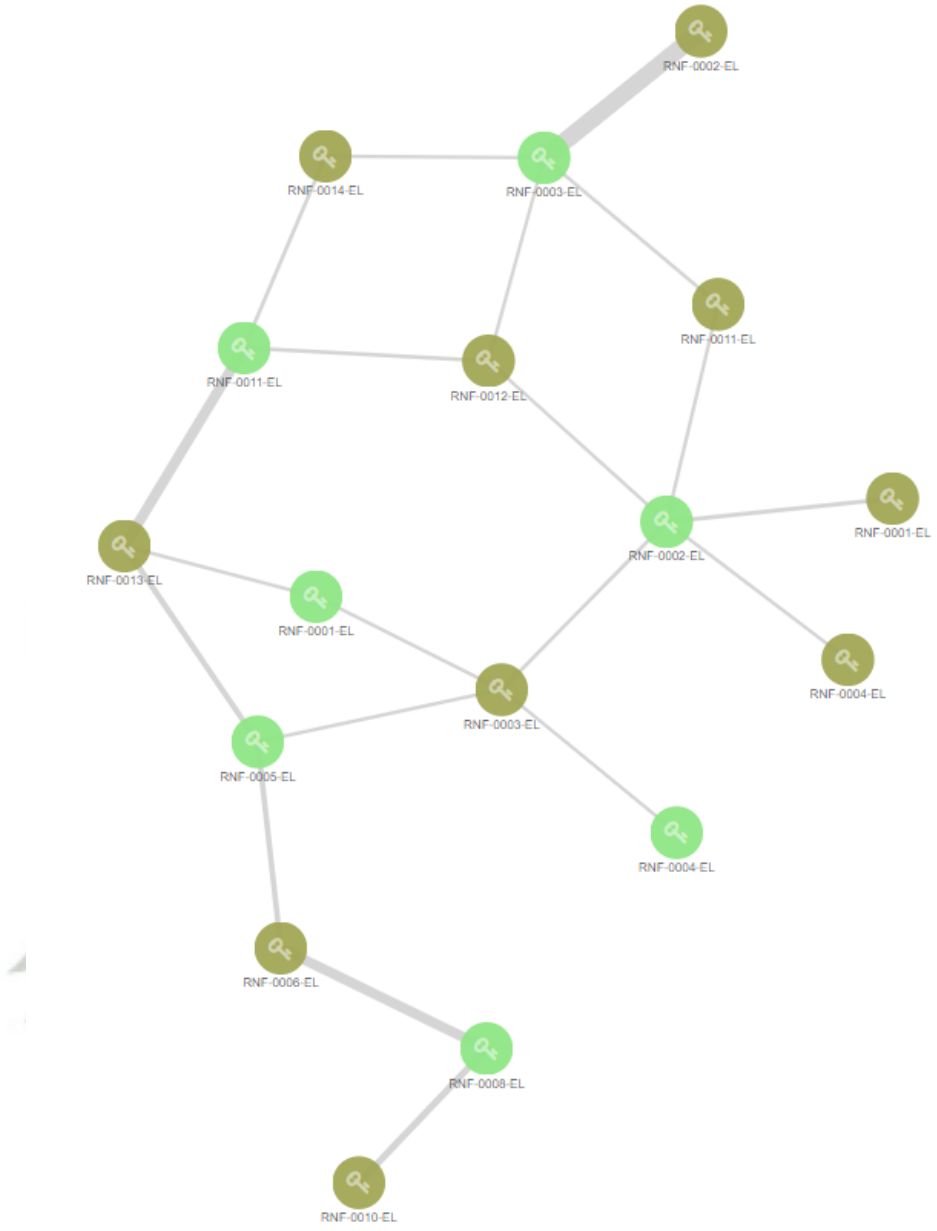


Figura 36. Grafo de relaciones en cuanto a conflictos de acuerdo al codigo
Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Por otro lado, en la siguiente figura se muestra la relación de dependencias entre los requerimientos no funcionales que se tomaron en cuenta con REMAN, en donde el padre vendría a ser el verde y el hijo el rosado, dando a conocer que la relación de dependencia es más grave cuando la relación de uno a otro es más gruesa

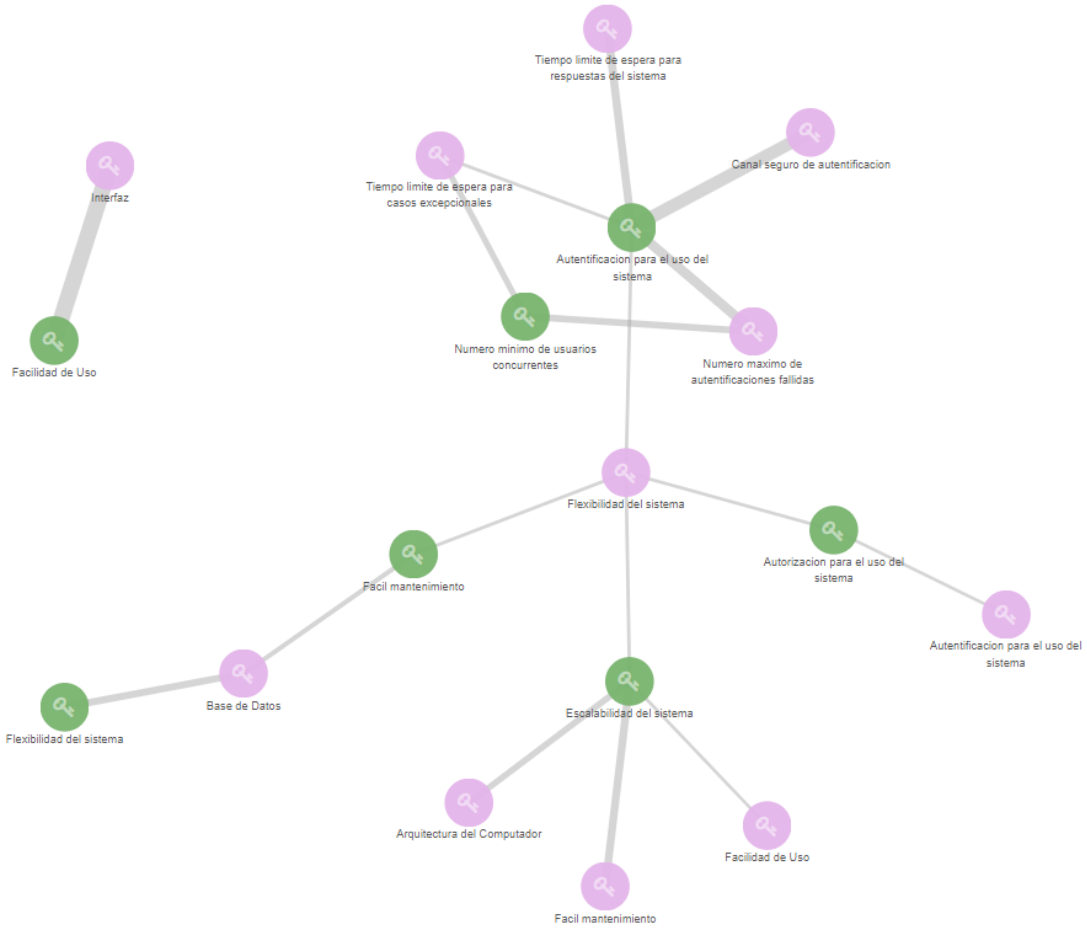


Figura 37. Grafo de relaciones en cuanto a conflictos (Padre: verde, Hijo: rosado)
Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

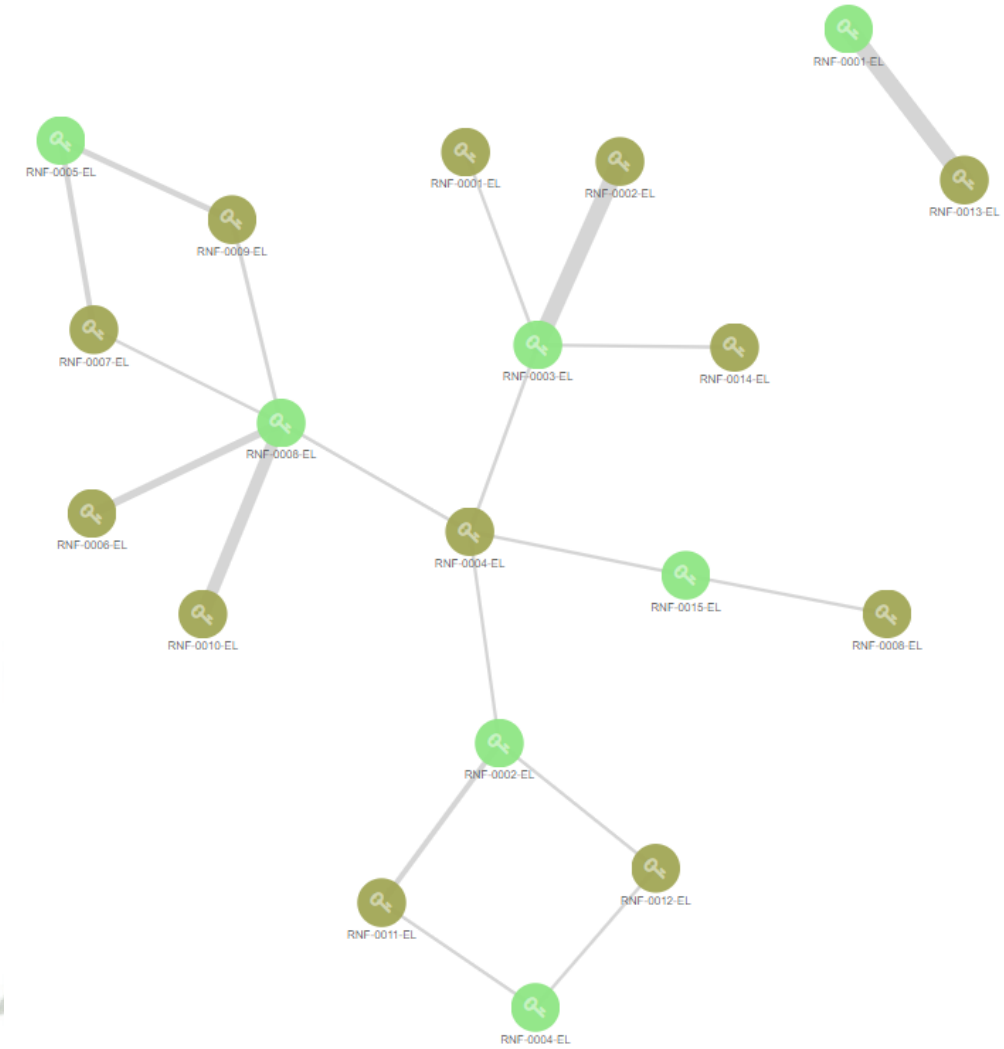


Figura 38. Grafo de relaciones en cuanto a conflictos de acuerdo al código (Padre: verde, Hijo: rosado)
Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CAPÍTULO 5: EVALUACIÓN

5.1. Evaluación por expertos

Para evaluar el modelo de asociación propuesto se tuvo contacto con un conjunto de personas que se dedicaban al desarrollo de software y que preferentemente elaboraren los artefactos de análisis del sistema en su etapa de la Ingeniería de Requisitos. Aunque varios de ellos empleaban metodologías y herramientas diferentes para resolver el problema de los requisitos, estas diferencias no perjudicaron al proyecto de investigación debido a que sólo se enfocaba a los requisitos no funcionales. Aunque emplearan diferentes metodologías para la toma de requerimientos funcionales.

Ninguno de los expertos mencionó emplear alguna metodología con respecto a los requerimientos no funcionales solo indicaron el uso de CASOS DE USO para representar los mismos o alguna plantilla de trabajo creadas por ellos mismos. Estas diferencias no influyeron en el proceso propuesto porque se trataba de identificar como asociaban los requerimientos no funcionales.

5.2. Perfil del Experto

Los expertos presentan el siguiente perfil:

- Estudios universitarios en Ciencias de la Computación, Ingeniería de Sistemas, Ingeniería de Software, Ingeniería en Informática o algún híbrido de ellos.
- Experiencia en la codificación de productos de software.
- Experiencia en el manejo de herramientas para la toma de los requisitos.
- Experiencia en el manejo de lenguajes de programación.
- Experiencia en los procesos de construcción de productos de software bajo modelos, metodologías o técnicas de la Ingeniería de Software.
- Experiencia en la administración de proyectos de software.
- Experiencia en la gestión de proyectos de software.

Bajo este perfil se pudo encontrar a 3 profesionales que trabajan liderando los siguientes proyectos:

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

- Desarrollo e integración de tecnologías de información en la Universidad Nacional de San Agustín de Arequipa. M.Sc. Paúl Mendoza del Carpio (experto N° 1).
- Software Factory de la Universidad La Salle. M.Sc. Medardo Delgado Paredes (experto N° 2).
- Fábrica de Software de la Universidad Nacional de San Agustín de Arequipa. M.Sc. Percy Huertas Niquén (experto N° 3)

A los tres profesionales se les explicó el modelo de asociación propuesto y se les solicitó su aplicación en algún módulo de los proyectos que vienen liderando. Finalmente, el modelo de asociación fue aplicado a los Planes de estudio y Asistencia de estudiantes.

La tabla 84 muestra los resultados:

Tabla 84. Módulos a los que se aplica el modelo de asociación

ORGANIZACIÓN	PROYECTO	MÓDULO	TIEMPO	CANTIDAD DE CONFLICTOS / DEPENDENCIAS
Universidad Nacional de San Agustín	Desarrollo e integración de tecnologías de información	Planes de estudio	90 días	44/42
Universidad La Salle	Servicios de Desarrollo del Sistema de Procedimiento	Seace	45 días	22/21
Universidad Nacional de San Agustín	Automatización de procesos académicos de la UNSA	Asistencia de alumnos	60 días	50/44

Fuente: Elaboración propia

Después de su aplicación se sostuvo entrevistas personales con los expertos para poder escuchar las opiniones sobre el uso del modelo de asociación. No se confeccionó cuestionarios debido a la rigidez de este.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

5.3. Resultados

Los resultados de las entrevistas se muestran a continuación:

Tabla 85. Resultado de entrevistas

EXPERTO	VENTAJAS DEL MODELO	DESVENTAJAS
1	• Ordena adecuadamente los requerimientos no funcionales • Asocia los problemas de los requerimientos funcionales en función de los requerimientos no funcionales • Se especifica mucho mejor los requerimientos funcionales • Los escenarios cobran especial relevancia ya que permiten entender la horizontalidad de los requerimientos.	• El procedimiento agrega tiempo en el desarrollo por lo que implica un mayor tiempo en la construcción del producto de software eliminando los conceptos de las metodologías ágiles
2	• Los casos de uso permiten explicar con mejor detalle los requerimientos no funcionales • Es posible un mejor detalle de los requerimientos funcionales debido a la especificación de los requerimientos no funcionales • Existe un mejor detalle cuando se comprueba los productos a partir de los requisitos y su asociación con los interfaces de usuario • Permite un mayor cuidado y consideración cuando la arquitectura es la base para los productos y sus modificaciones o escalabilidad	• No integra métodos formales como respuesta a un análisis de requerimientos que se encuentran definidos en casos de uso y especificados por medio de escenarios
3	• Existe una mejor especificación del catálogo de requisitos • Permite una mejor trazabilidad de los requerimientos ya que asocia los requerimientos no funcionales con los requerimientos funcionales • Existe una mejor especificación en la usabilidad del producto • Permite una mayor flexibilidad en la arquitectura del producto	• No presenta un modelo de trazabilidad integral en donde se especifique los efectos secundarios de los requerimientos funcionales en función de los requerimientos no funcionales

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Los factores identificados que producen conflictos en los requerimientos no funcionales fueron los siguientes:

Tabla 86. Identificación de factores

EXPERTO	FACTORES IDENTIFICADOS
1	• Mala visión del analista con respecto a los requerimientos no funcionales • Desconocimiento del efecto principal de un requerimiento no funcional • Falta de experiencia sobre los lenguajes de programación • Falta de experiencia en aspectos de tecnologías de información
2	• Desconocimiento en el uso de los casos de uso • Mala interpretación y percepción de los escenarios • Mala interpretación de la arquitectura • Mala adecuación de los interfaces de usuario ya que normalmente se emplea la percepción del desarrollador • Los analistas se dedican en su integridad a los requerimientos funcionales y dejan al final los requerimientos no funcionales; nunca los trabajan en forma paralela
3	• Desconocimiento del efecto e implicancia de la trazabilidad • No se contextualiza los requerimientos no funcionales • Falta de experiencia en la percepción de requerimientos no funcionales • No se involucra al usuario en el proceso de análisis del producto de software a construir

Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

CONCLUSIONES

PRIMERA: Las características asociadas a los requerimientos no funcionales deben ser precisas para así lograr una mejor redacción de los requerimientos funcionales. Los requerimientos no funcionales deben ser presentados en su totalidad para obtener un mejor sentido en la construcción del producto de software.

SEGUNDA: La identificación del conjunto de factores que generan conflictos entre los requerimientos no funcionales permite un verdadero ordenamiento en la elaboración del catálogo de requisitos funcionales y las condiciones propuestos en los casos de uso.

TERCERA: Los ítems propuestos en el modelo de asociación permiten una forma adecuada y secuencial de mantener ordenado el catálogo de requisitos siempre que sean considerados los requerimientos no funcionales y su respectiva trazabilidad.

CUARTA: Los escenarios y los casos de uso permiten lograr entender los conflictos entre los requerimientos no funcionales y su repercusión en los requerimientos funcionales. Los escenarios permiten observar los problemas transversales de los requerimientos y los casos de uso su especificación en forma puntual.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

RECOMENDACIONES

PRIMERA: Incluir temas relacionados a la Ingeniería de Software como por ejemplo métodos formales, patrones ontológicos de dominio entre otros.

SEGUNDA: Construir una herramienta de software que automatice el modelo de asociación logrando una interpretación en lenguaje natural de los requisitos no funcionales y una transformación automática para lograr las dependencias de los requisitos no funcionales.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

BIBLIOGRAFIA

(Alarcón, 2008) Alarcón Pedro Pablo. Metodologías Ágiles desde la Perspectiva de la Especificación de Requisitos Funcionales y No-Funcionales. JISBD, 333-344

(Bertolami, 2001) Bertolami Mabel, Centeno M. Elena, LEL y Escenarios de la Administración de Recepción del Hotel. Caso de estudio desarrollado en el marco del Magister en Ingeniería del Software, UNLP. 2001.

(Bo et al, 2011) Bo Wei, Zhi Jin, and Didar Zowghi. An Automatic Reasoning Mechanism for NFR Goal Models. Fifth IEEE International Conference on Theoretical Aspects of Software Engineering. 2011.

(Chaves, 2011) Chaves, M. A. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. InterSedes, 6(10). 2011.

(Chin-Lun, 2016) Chin-Lun Liu. CDNFRE: Conflict detector in non-functional requirement evolution based on ontologies. Computer Standards & Interfaces. Elsevier. 2016.

(Chun et al, 1999) Chun Lawrence et. al. NON-FUNCTIONAL REQUIREMENTS IN SOFTWARE ENGINEERING. SPRINGER SCIENCE+BUSINESS MEDIA, LLC. 1999.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

(Gil, 2002) Gustavo Daniel Gil, "Herramienta Para Implementar LEL Y Escenarios", Tesis Magíster En Ingeniería De Software, Facultad de Informática, Universidad Nacional de la Plata, Argentina 2002.

(Gómez, 2011) Gómez María del Carmen. Notas del curso de Análisis de Requerimientos. Universidad Autónoma Metropolitana. México. 2011.

(Jaconson et al, 2000) Jacobson, I., Booch, G., Rumbaugh, J. El Proceso Unificado de Desarrollo de Software. Addison-Wesley, 2000.

(Kaplan, 2002) Kaplan,V.,Hadad, G., Oliveros, A., Uso de Lexico Extendido del Lenguaje (LEL) y de Escenarios para la Elicitacion de Requerimientos. Aplicación a un Caso Real, Informe de Investigación Dpto. de Investigación de la Universidad de Belgrano, Argentina. 2002

(Mairinza, 2010) Mairinza Dewi et. al. Conflict Characterization and Analysis of Non Functional Requirements: An Experimental Approach. Faculty of Engineering and Information TechnologyUniversity of Technology Sydney, Australia. 2010.

(Ming-Xun et al, 2012) Ming-Xun, Xin-Xing Luo, Xiao-Hong Chen, Desheng Dash Wu. A non-functional requirements tradeoff model in Trustworthy Software. Information Sciences. Elsevier. 2012.

(Mora, 1999) Mora, Julia. "Transformación y gestión curricular". En: Memorias Seminario Taller Evaluación y Gestión Curricular, Universidad de Antioquía. 1999.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

(Pohl, 2013) Pohl, K. The three dimensions of requirements engineering. In Seminal Contributions to Information Systems Engineering (pp. 63-80). Springer Berlin Heidelberg. 2013.

(Pressman, 2007) Pressman Roger. Ingeniería de Software. Un enfoque práctico. Sexta edición. Prentice Hall. México. 2007.

(Rao, 2011) Rao Ananda. Four Layered Approach to Non-Functional Requirements Analysis. IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 6, No 2. ISSN (Online): 1694-0814. 2011.

(Ruidias, 2014) Ruidias Héctor, Caliusco María, Gali María. Construcción basada en ontologías del Léxico Extendido del Lenguaje. 15th Argentine Symposium on Software Engineering, ASSE 2014. 43 JAIIO - ASSE 2014 - ISSN: 1850-2792 - Página 188.

(Siqueira, 2015) Siqueira Simoes Guilherme. Gestión de Requerimientos: El talón de Aquiles de los proyectos. SG Conference & Expo. 2015.

(Supakkul, 2010) Supakkul Sam, et. al. An NFR Pattern Approach to Dealing with NFRs. 18th IEEE International Requirements Engineering Conference. 2010.

(Tabassum, 2014) Tabassum Mirza et. al. Determining Interdependency Among Non-functional Requirements to Reduce Conflict. 3rd INTERNATIONAL CONFERENCE ON INFORMATICS, ELECTRONICS & VISION. 2014.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

(Tuffley, 2005) Angela Tuffley CIT3190 IT Project Course. Requirements Elicitation & Management. 2005.

(Valencia, 1996) Valencia, Carlos. Gerencia de Proyectos. Seminario para profesores Universidad de Antioquía. 1996.

(ULaSalle, 2017) Universidad La Salle. Proyecto: Servicios de Desarrollo del Sistema de Procedimiento. Arequipa, 2017.

(Unsa, 2017a) Universidad Nacional de San Agustín. Proyecto: Desarrollo en Integración Tecnológica UNSA. Arequipa. 2017.

(Unsa, 2017b) Universidad Nacional de San Agustín. Proyecto: Automatización de procesos académicos de la UNSA. Arequipa. 2017.

(Velasco, 2009) Velasco Malillo, Honorio M.; Díaz De Rada, Ángel. La lógica de la investigación etnográfica: un modelo de trabajo para etnógrafos de la escuela (6ª edición). Editorial Trotta. p. 303. ISBN 9788481646283.

(Wiegers, 2013) Wiegers Karl. Software Requirements. Microsoft Press. 2013.

(Yin, 2013) Yin Robert. Case Study Research: Design and Methods. Second Edition. Applied Social Research Methods Series. Volume 5. 2013 SAGE Publications.

(Zoltan, 1995) Zoltan, Szabó. Seminario sobre Gestión Tecnológica. SENA. 1995.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

ANEXO 1

ESPECIFICACION DEL MODELO AUTOMATIZADO MAIC

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

1. INFORMACION GENERAL

NOMBRE DEL PRODUCTO: MAIC App

ACTORES: Todo aquel que tenga conocimientos en Proyectos de Ingeniería de Software y Requerimientos.

DESCRIPCIÓN DEL PRODUCTO: Aplicación Web construida para el manejo y presentación del Modelo de Asociación e Identificación de Requerimientos No Funcionales empleando escenarios y Casos de Uso.

PROPOSITO: el propósito principal de esta aplicación es presentar y dar a conocer dicho modelo de manera eficiente y automatizada con la ayuda del caso de estudio REMAN.

REPOSITORIO DE CODIGO

Link: https://github.com/gon009/MAIC

2. REGLAS DE NEGOCIO

- Un requerimiento no funcional no puede tener conflicto consigo mismo.

3. ELICITACION

3.1. Preguntas

- ¿Sera una aplicación web, escritorio o consola?

  Se realizó un cuadro de comparación, y se tomó la decisión de hacerla Web

	Ventajas	**Desventajas**
Consola	• Más rápido. • Consume menos recursos.	• Interfaz no es muy intuitiva. • Un usuario con poco conocimiento de computación tendría problemas.
Escritorio	• No necesita conexión a internet. • Fácil de usar	• Se tendría que instalar la aplicación cada vez en un nuevo ordenador
Web	• Accesible desde cualquier ordenador con un navegador web	• Costo de seguridad • Costo por hosting y dominio

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

4. CRITERIOS DE ACEPTACION

Característica: Detector de conflictos

Entonces que introduzco los Requerimientos No funcionales

Y los escenarios de casos de uso

Y los casos de uso

Como experto en Requerimientos

Quiero ver los conflictos entre los requerimientos no funcionales

Escenario: Ingresar información

Dado que estoy en la aplicación

Cuando le doy clic a "Crear uno nuevo"

Entonces me debe aparecer la vista para registrar un elemento

Escenario: Mostrar Detalles

Dado que estoy en la aplicación

Cuando le doy clic a "Detalles"

Entonces me debe aparecer la vista con los detalles del elemento

Escenario: Eliminar información

Dado que estoy en la aplicación

Cuando le doy clic a "Eliminar"

Entonces me debe aparecer la vista para eliminar el elemento

Escenario: Editar Información

Dado que estoy en la aplicación

Cuando le doy clic a "Editar"

Entonces me debe aparecer la vista para editar elemento

Escenario: Mostrar Conflictos

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

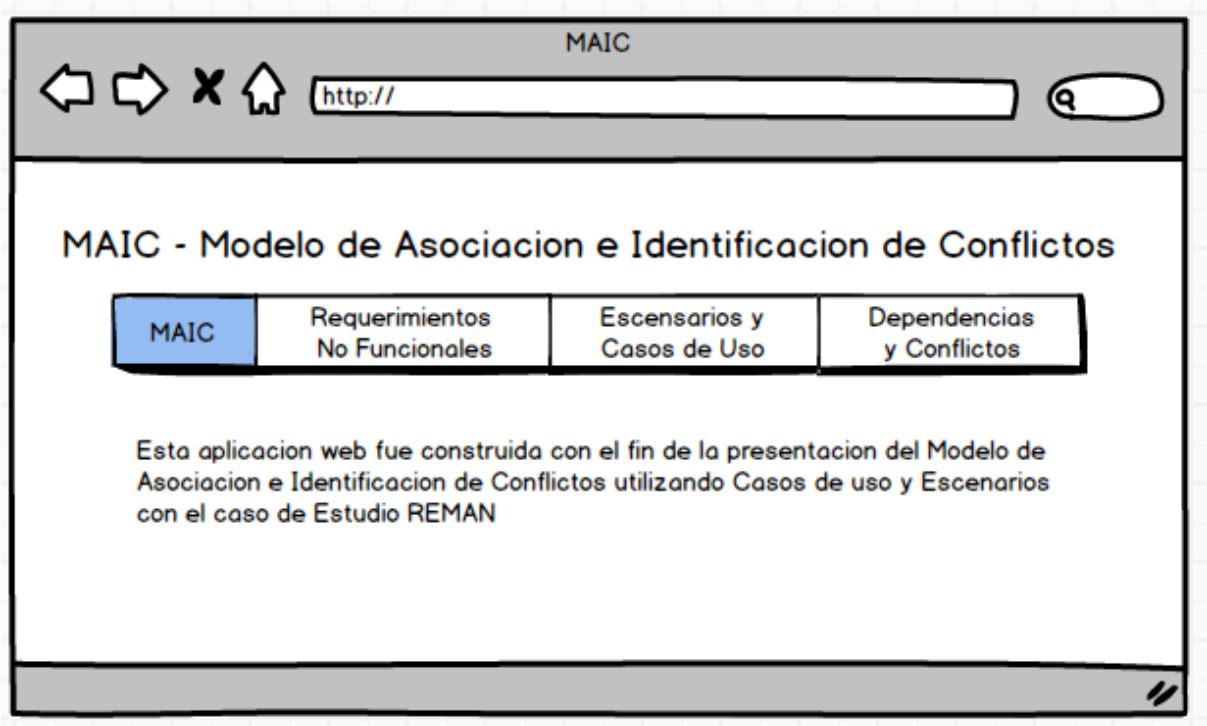
UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Dado que estoy en la aplicación

Cuando le doy clic a "Reportes"

Entonces debo ver los reportes de conflictos entre requerimientos no funcionales

5. PROTOTIPO



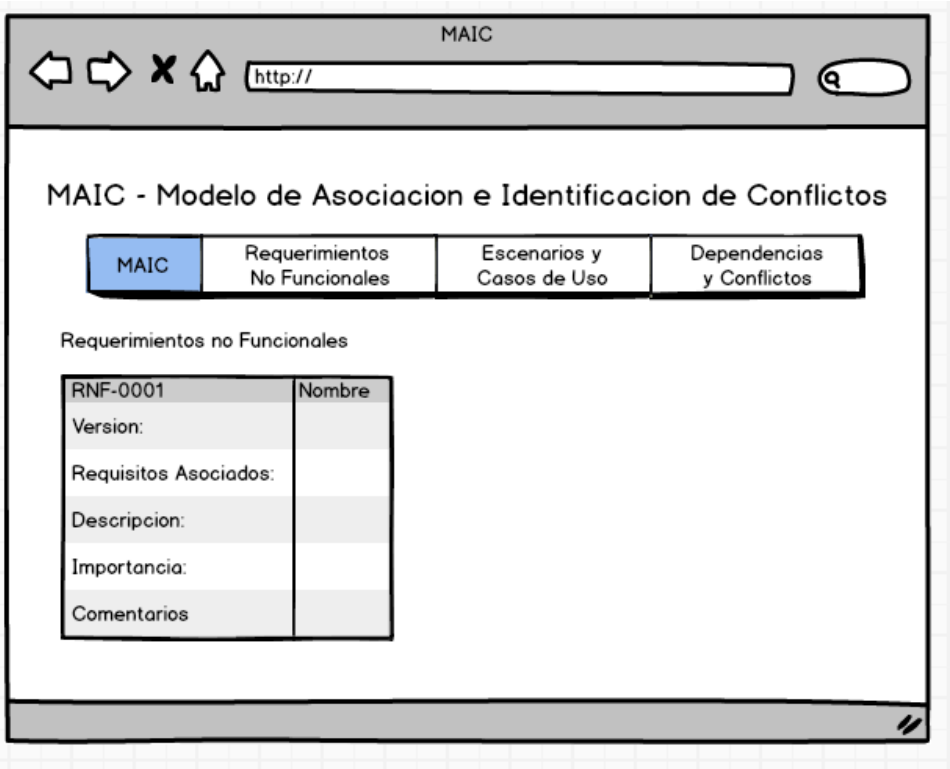
Prototipo 1. Pantalla Principal

Fuente: Elaborado por el autor

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

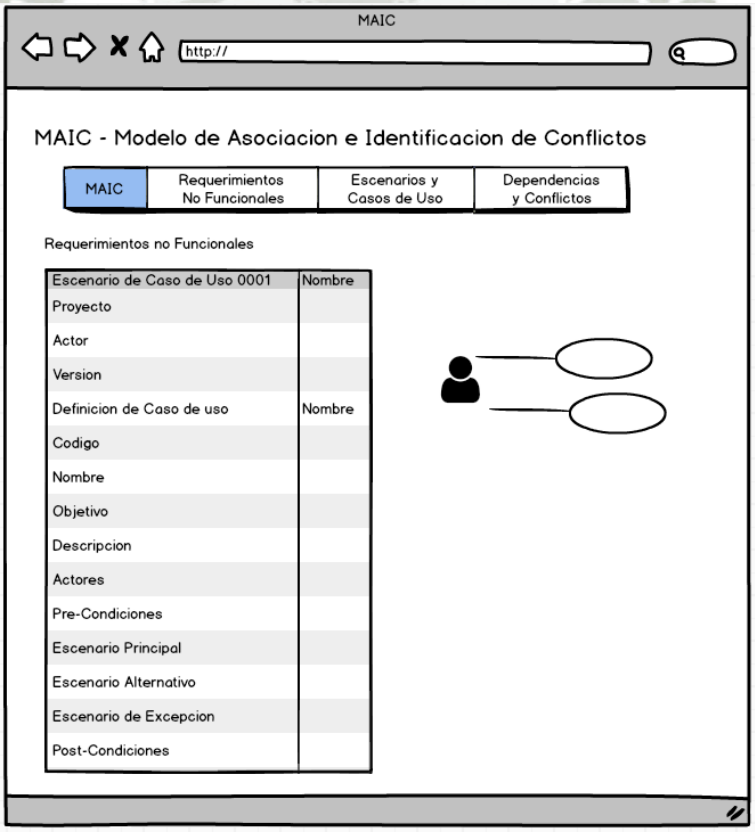
REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA



Prototipo 2. Pantalla de Requerimientos No Funcionales

Fuente: Elaborado por el autor



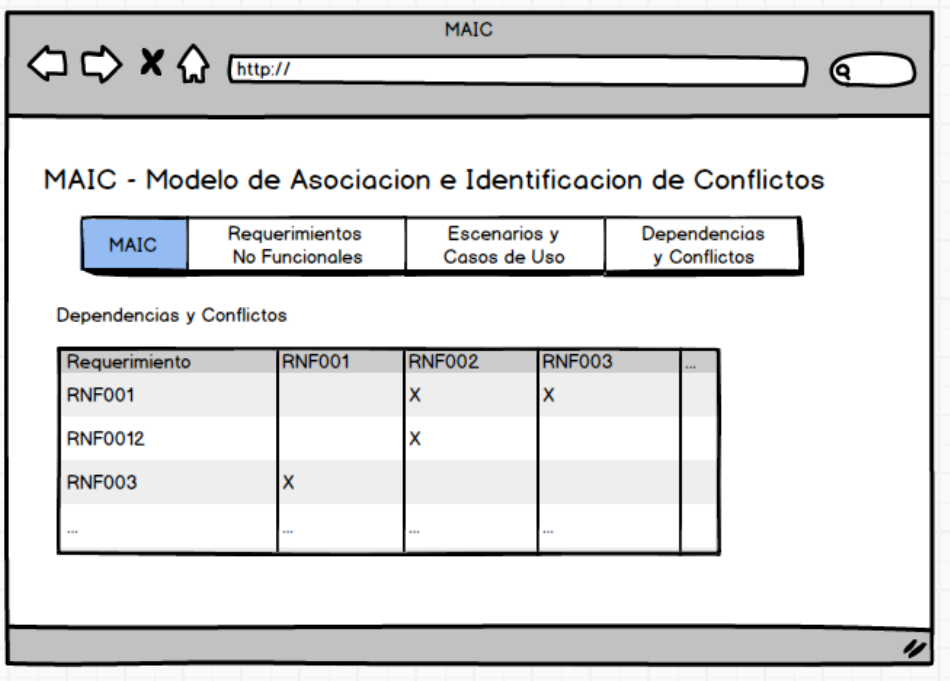
Prototipo 3. Pantalla de Escenarios y Casos de Uso

Fuente: Elaborado por el autor

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

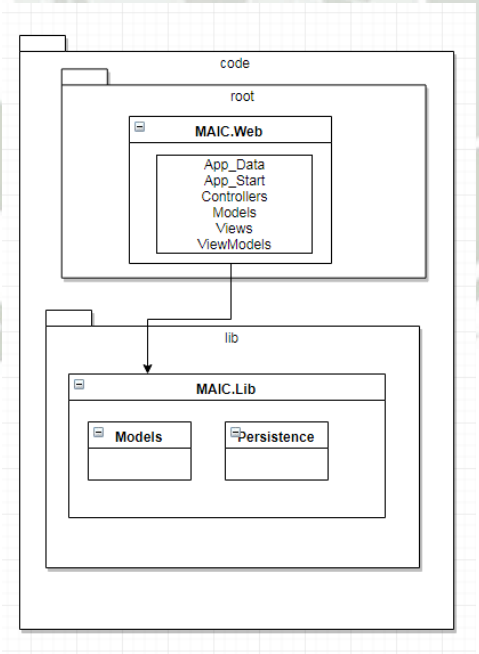
UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA



Prototipo 4. Pantalla de Conflictos

Fuente: Elaborado por el autor

6. MODELOS

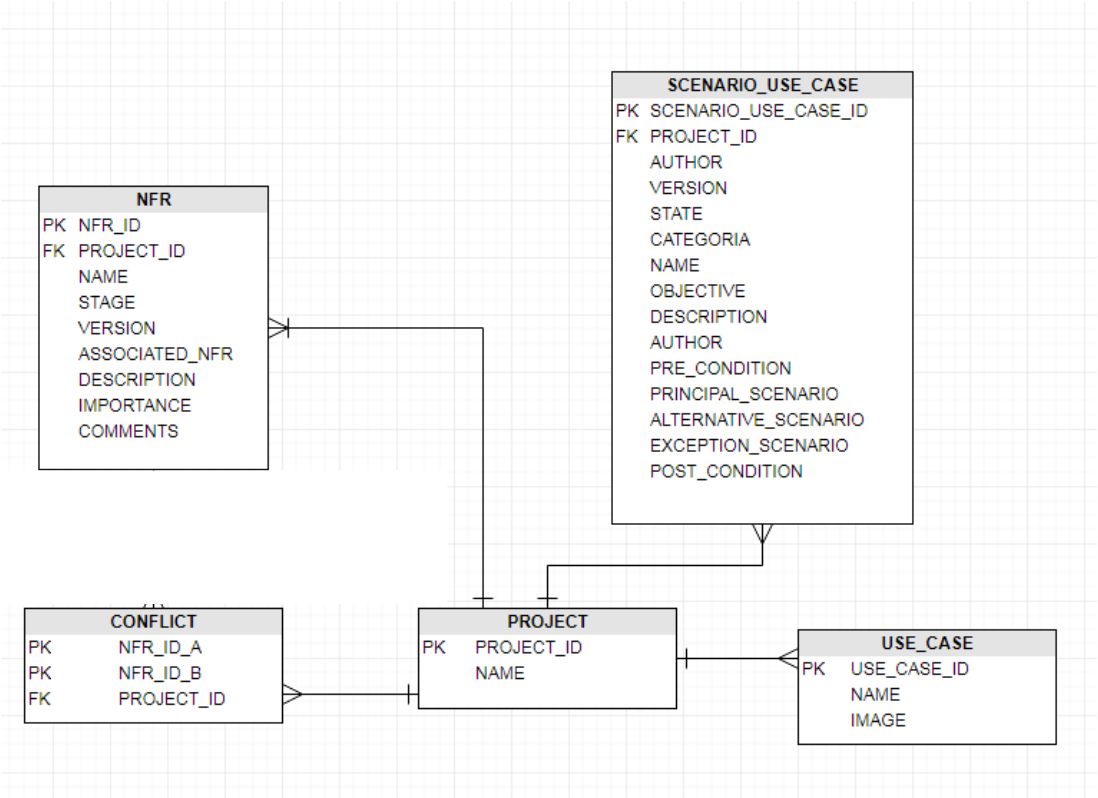


Modelo 1. Arquitectura de la Aplicación

Fuente: Elaborado por el autor

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA



Modelo 2. Modelo de Base de Datos

Fuente: Elaborado por el autor

7. OTROS

7.1. Detalles

- Se utiliza MVC como patrón de diseño
- Se utiliza una metodología ágil para su desarrollo, SCRUM
- Se utiliza Entity Framework para el mapeo con la base de datos
- Se utiliza SQL Server 2014 Enterprise
- Se utiliza el lenguaje C#
- Se utiliza el IDE: Visual Studio 2017
- Se utiliza GitHub como repositorio en la nube
- Se utiliza Git para el control de versiones

7.2. Glosario

- **MVC**: Modelo-vista-controlador (MVC) es un patrón de arquitectura de software, que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

- **Entity Framework:** es un conjunto de tecnologías de ADO.NET que permiten el desarrollo de aplicaciones de software orientadas a datos.
- **GitHub**: GitHub es una forja (plataforma de desarrollo colaborativo) para alojar proyectos utilizando el sistema de control de versiones Git.
- **Git**: es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente
- **MAIC**: siglas de "Modelo de Asociación e Identificación de Conflictos"

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

ANEXO 2

ELASTICSEARCH

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

1. Concepto

1.1 Elasticsearch: es un motor de búsqueda basado en Lucene. Proporciona un motor de búsqueda de texto completo distribuido y apto para múltiples usuarios con una interfaz web HTTP y documentos JSON sin esquema. Elasticsearch se desarrolla en Java y se lanza como código abierto bajo los términos de la licencia de Apache. Los clientes oficiales están disponibles en Java, .NET (C #), PHP, Python, Apache Groovy y muchos otros idiomas. Elasticsearch es el motor de búsqueda empresarial más popular seguido por Apache Solr, también basado en Lucene.

Elasticsearch cuenta con dos clientes APIs para .NET: Elasticsearch.Net y NEST.

1.1.1 Elasticsearch.NET

Elasticsearch.Net es un cliente de muy bajo nivel, sin dependencias, que no tiene opiniones sobre cómo compila y representa sus solicitudes y respuestas. Se ha abstraído lo suficiente como para que todos los puntos finales de la API Elasticsearch se representen como métodos, pero no demasiado para obstaculizar la forma en que desea construir sus objetos json / solicitud / respuesta

1.1.2 NEST

NEST es un cliente de alto nivel que tiene la ventaja de haber mapeado todos los objetos de solicitud y respuesta, viene con una DSL de consulta fuertemente tipada que asigna 1 a 1 con la consulta DSL Elasticsearch, y aprovecha funciones específicas de .NET como resultados covariantes y mapeo automático de POCO. NEST usa y expone internamente el cliente Elasticsearch.Net de bajo nivel.

1.2 Lucene: es una API de código abierto para recuperación de información, originalmente implementada en Java por Doug Cutting. Está apoyado por el Apache Software Foundation y se distribuye bajo la Apache Software License. Lucene tiene versiones para otros lenguajes incluyendo Delphi, Perl, C#, C++, Python, Ruby y PHP.

Es útil para cualquier aplicación que requiera indexado y búsqueda a texto completo. Lucene ha sido ampliamente usado por su utilidad en la implementación de motores de búsquedas. Por ello, a veces se confunde Lucene con un motor de búsquedas con funciones de "crawling" y análisis de documentos en HTML incorporadas.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

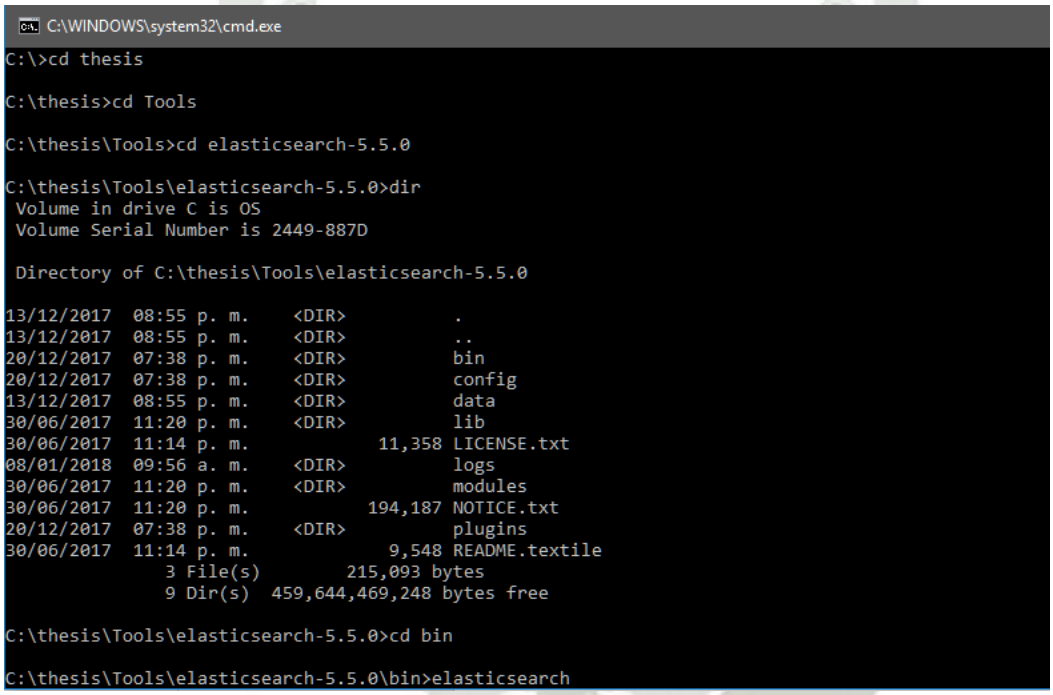
REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

El cliente que se estará utilizando será: Elasticsearch versión 5.5 con NEST como API para .NET versión 5.5 de igual manera.

2. Instalación

- Se descarga el motor de búsqueda de la pagina web: https://www.elastic.co/downloads/elasticsearch
- Se extrae en una carpeta y media cmd accedemos a su ubicación
- Con el comand cd nos colocamos dentro de la carpeta /bin
- Escribimos elasticsearch y damos enter



- El servicio tardara unos pocos segundos e iniciara

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

```
C:\thesis\Tools\elasticsearch-5.5.0>cd bin

C:\thesis\Tools\elasticsearch-5.5.0\bin>elasticsearch
[2018-01-10T10:13:35,942][INFO ][o.e.n.Node               ] [] initializing ...
[2018-01-10T10:13:36,096][INFO ][o.e.e.NodeEnvironment    ] [tt45CDa] using [1] data paths, mounts [[O
S (C:)]], net usable_space [428.1gb], net total_space [940.7gb], spins? [unknown], types [NTFS]
[2018-01-10T10:13:36,102][INFO ][o.e.e.NodeEnvironment    ] [tt45CDa] heap size [1.9gb], compressed or
dinary object pointers [true]
[2018-01-10T10:13:37,044][INFO ][o.e.n.Node               ] node name [tt45CDa] derived from node ID [
tt45CDaSTvmoj3VjTmfd3w]; set [node.name] to override
[2018-01-10T10:13:37,051][INFO ][o.e.n.Node               ] version[5.5.0], pid[924], build[260387d/20
17-06-30T23:16:05.735Z], OS[Windows 10/10.0/amd64], JVM[Oracle Corporation/Java HotSpot(TM) 64-Bit Ser
ver VM/1.8.0_151/25.151-b12]
[2018-01-10T10:13:37,059][INFO ][o.e.n.Node               ] JVM arguments [-Xms2g, -Xmx2g, -XX:+UseCon
cMarkSweepGC, -XX:CMSInitiatingOccupancyFraction=75, -XX:+UseCMSInitiatingOccupancyOnly, -XX:+DisableE
xplicitGC, -XX:+AlwaysPreTouch, -Xss1m, -Djava.awt.headless=true, -Dfile.encoding=UTF-8, -Djna.nosys=t
rue, -Djdk.io.permissionsUseCanonicalPath=true, -Dio.netty.noUnsafe=true, -Dio.netty.noKeySetOptimizat
ion=true, -Dio.netty.recycler.maxCapacityPerThread=0, -Dlog4j.shutdownHookEnabled=false, -Dlog4j2.disa
ble.jmx=true, -Dlog4j.skipJansi=true, -XX:+HeapDumpOnOutOfMemoryError, -Delasticsearch, -Des.path.home
=C:\thesis\Tools\elasticsearch-5.5.0]
[2018-01-10T10:13:40,206][INFO ][o.e.p.PluginsService     ] [tt45CDa] loaded module [aggs-matrix-stats
]
[2018-01-10T10:13:40,211][INFO ][o.e.p.PluginsService     ] [tt45CDa] loaded module [ingest-common]
[2018-01-10T10:13:40,216][INFO ][o.e.p.PluginsService     ] [tt45CDa] loaded module [lang-expression]
[2018-01-10T10:13:40,220][INFO ][o.e.p.PluginsService     ] [tt45CDa] loaded module [lang-groovy]
[2018-01-10T10:13:40,225][INFO ][o.e.p.PluginsService     ] [tt45CDa] loaded module [lang-mustache]
```

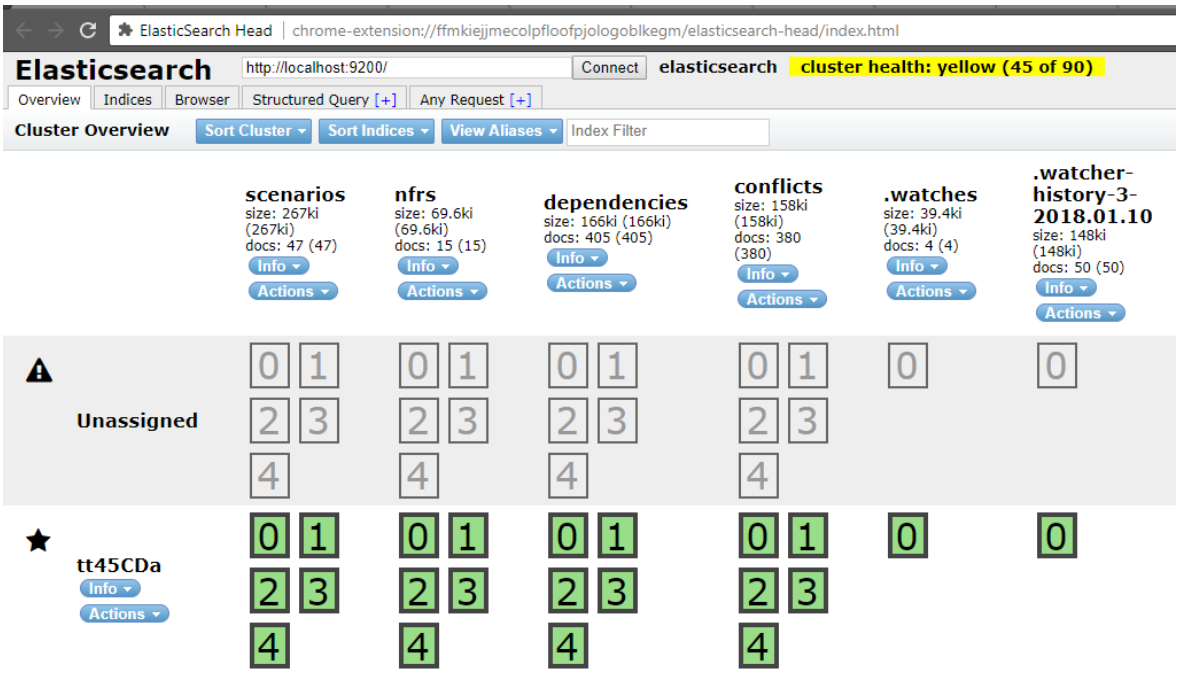
– Para comprobar que el servicio este levantado con éxito, nos dirigimos a su dirección por default: localhost:9200 y nos tendría que aparecer algo así:


```
{
  "name" : "tt45CDa",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "eCvL3NNkQRuoBWO-eXJ3Cw",
  "version" : {
    "number" : "5.5.0",
    "build_hash" : "260387d",
    "build_date" : "2017-06-30T23:16:05.735Z",
    "build_snapshot" : false,
    "lucene_version" : "6.6.0"
  },
  "tagline" : "You Know, for Search"
}
```


– Para obtener una mejor vista sobre los índices en elastic search, se puede instalar la siguiente extensión de Chrome: https://chrome.google.com/webstore/detail/elasticsearch-head/ffmkiejjmecolpfloofpjologoblkegm

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA



3. Ventajas y Desventajas

3.1 Ventajas

Se podrían enumerar varias ventajas que brinda esta herramienta. Algunas de las más destacables son las siguientes:

- Al estar desarrollado en Java, es sumamente compatible con casi todas las plataformas.
- Tiene una gran velocidad de respuesta.
- Esta distribuido, lo que lo hace fácilmente escalable y adaptable a las distintas situaciones.
- Simple realizar respaldos de los datos que se tienen.
- Utiliza objetos JSON como respuesta, por lo que es fácil de invocar a partir de varios lenguajes de programación.

3.2 Desventajas

Como todo, ElasticSearch posee algunas (pocas) desventajas:

- Solo soporta como tipos de respuesta JSON, lo que, si bien lo hace accesible, también lo limita al no involucrar otros lenguajes, como podrían ser: CSV o XML.
- Puede suceder que en algunas situaciones caiga en un caso de "split brain".

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

4. Conexión y consultas

NEST permite ejecutar una consulta de búsqueda y recuperar resultados de búsqueda que coinciden con la consulta.

Se están utilizando dos búsquedas, una para detectar las dependencias y otra para detectar los conflictos.

4.1 Conexión

La conexión a Elasticsearch con Elasticsearch.Net y NEST es fácil, pero es muy posible que desee cambiar el comportamiento de conexión predeterminado. Hay varias opciones de configuración disponibles en ConnectionSettings (y ConnectionConfiguration para Elasticsearch.Net) que se pueden usar para controlar cómo interactúan los clientes con Elasticsearch.

Para conectarse a Elasticsearch ejecutando localmente en http: // localhost: 9200 es tan simple como crear instancias de una nueva instancia del cliente

```
var client = new ElasticClient ();
```

A menudo es posible que deba pasar opciones de configuración adicionales al cliente, como la dirección de Elasticsearch si se está ejecutando en una máquina remota. Aquí es donde entran ConnectionSettings; una instancia puede ser instanciada para proporcionar al cliente diferentes configuraciones.

```
var settings = new ConnectionSettings (new Uri ("http://example.com:9200"))
    .DefaultIndex ("personas");
var client = new ElasticClient (configuraciones);
```

4.2 Mapeo

NEST ofrece una función llamada mapeo automático que puede inferir automáticamente los campos de las clases en código, y crear así los índices de acuerdo a las clases.

A continuación un ejemplo con la clase: Empleado y Compañía:

Clase Compañía:

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

```
public class Company
{
    public string Name { get; set; }
    public List<Employee> Employees { get; set; }
}
```

Clase Empleado:

```
public class Employee
{
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public int Salary { get; set; }
    public DateTime Birthday { get; set; }
    public bool IsManager { get; set; }
    public List<Employee> Employees { get; set; }
    public TimeSpan Hours { get; set; }
}
```

AutoMapeo y creación de índice automático

```
var descriptor = new CreateIndexDescriptor("myindex")
    .Mappings(ms => ms
        .Map<Company>(m => m.AutoMap())
        .Map<Employee>(m => m.AutoMap())
    );
```


Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

4.3 Búsqueda

La API de búsqueda le permite ejecutar una consulta de búsqueda y recuperar resultados de búsqueda que coinciden con la consulta.

Una vez que haya indexado los datos en Elasticsearch, se va a querer buscarlos. Elasticsearch ofrece una poderosa consulta DSL para definir consultas para ejecutar en Elasticsearch. Esta DSL está basada en JSON y está expuesta en NEST en la forma de una API Fluent y una sintaxis de inicializador de objetos

Existen consultas tan básicas como un GET de todo el índice:

```
var searchResponse = client.Search<Project>(s => s
    .Query(q => q
        .MatchAll()
    )
);
```

Como consultas combinadas con filtros:

```
var searchResponse = client.Search<Project>(s => s
    .Query(q => q
        .Bool(b => b
            .Must(mu => mu
                .Match(m => m
                    .Field(f => f.LeadDeveloper.FirstName)
                    .Query("Russ")
                ), mu => mu
                .Match(m => m
                    .Field(f => f.LeadDeveloper.LastName)
                    .Query("Cam")
                )
```


Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

```
            )
            .Filter(fi => fi
                .DateRange(r => r
                    .Field(f => f.StartedOn)
                    .GreaterThanOrEquals(new DateTime(2017, 01, 01))
                    .LessThan(new DateTime(2018, 01, 01))
                )
            )
        )
    )
);
```

Encuentra documentos donde “LeadDeveloper First Name” es: “Russ” y donde “LeadDeveloper Last Name” es “Cam” y donde el “Project” se encuentre iniciado en el 2017

6. Enlaces

https://www.elastic.co/

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

ANEXO 3

KIBANA

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

1. **Concepto**

   Kibana es una herramienta open-source perteneciente a Elastic, que permite visualizar y explorar datos que se encuentran indexados en ElasticSearch, es decir, un plugin de ElasticSearch. Kibana; también es conocido por el stack ELK:

   - Elasticsearch
   - Logstash
   - Kibana

   El stack ELK (Elasticsearch, Logstash y Kibana) es una herramienta ideal para la agregación de logs, visualización y análisis, aunque no está diseñado específicamente para este propósito. Individualmente, la instalación de Elasticsearch, Logstash y Kibana y configurarlos para que se comuniquen entre sí, no es una tarea fácil. Afortunadamente, con Docker y Docker Compose podemos crear nuestros contenedores con el stack ELK de forma fácil.

2. **Entorno**

   La configuración de kibana es muy simple, solo indicamos donde se encuentra elasticsearch, donde tiene más configuración logstash y donde tenemos que definir los inputs y filters.

   Kibana tiene varias opciones para la visualización de data y de acuerdo a inputs y filtros podemos obtener diferentes maneras de jugar con los datos que se encuentran indexados.

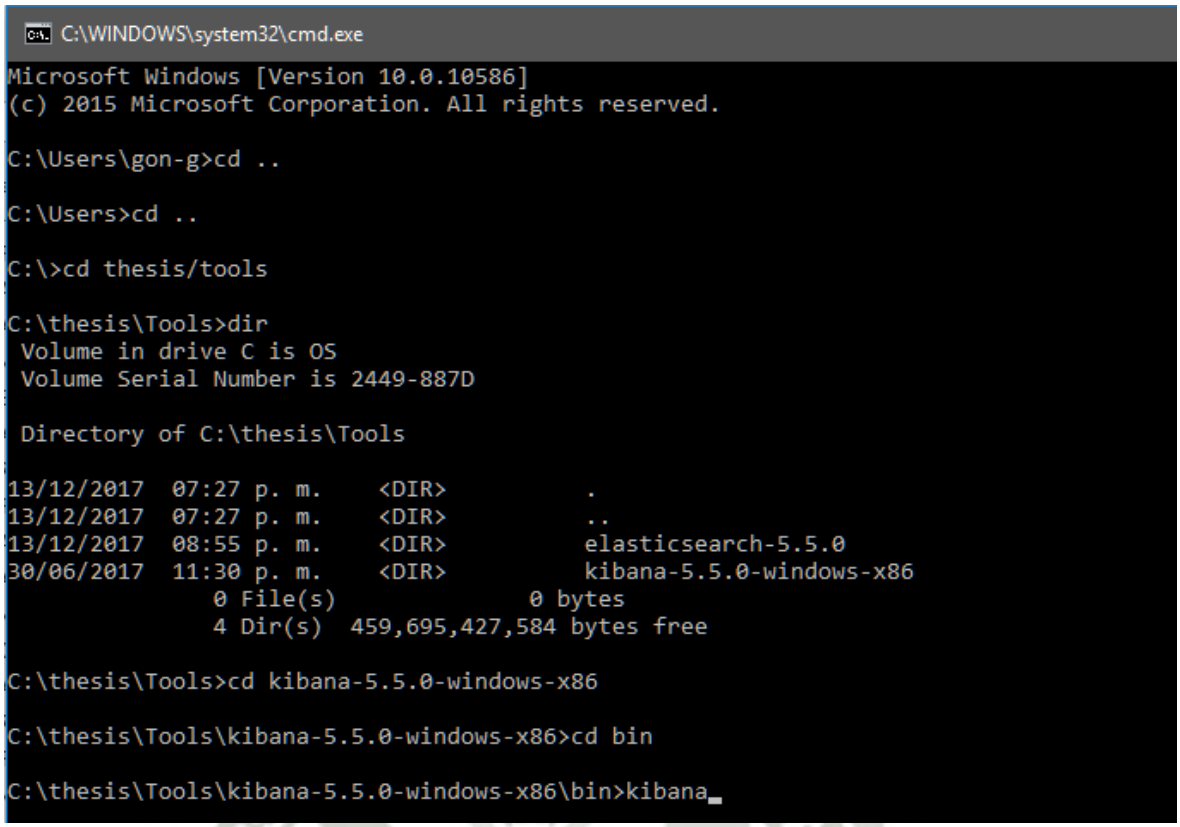
3. **Instalación**

   – Se descarga el motor de búsqueda de la pagina web: https://www.elastic.co/downloads/kibana
   – Se extrae en una carpeta y media cmd accedemos a su ubicación
   – Con el comand cd nos colocamos dentro de la carpeta /bin
   – Escribimos kibana y damos enter.

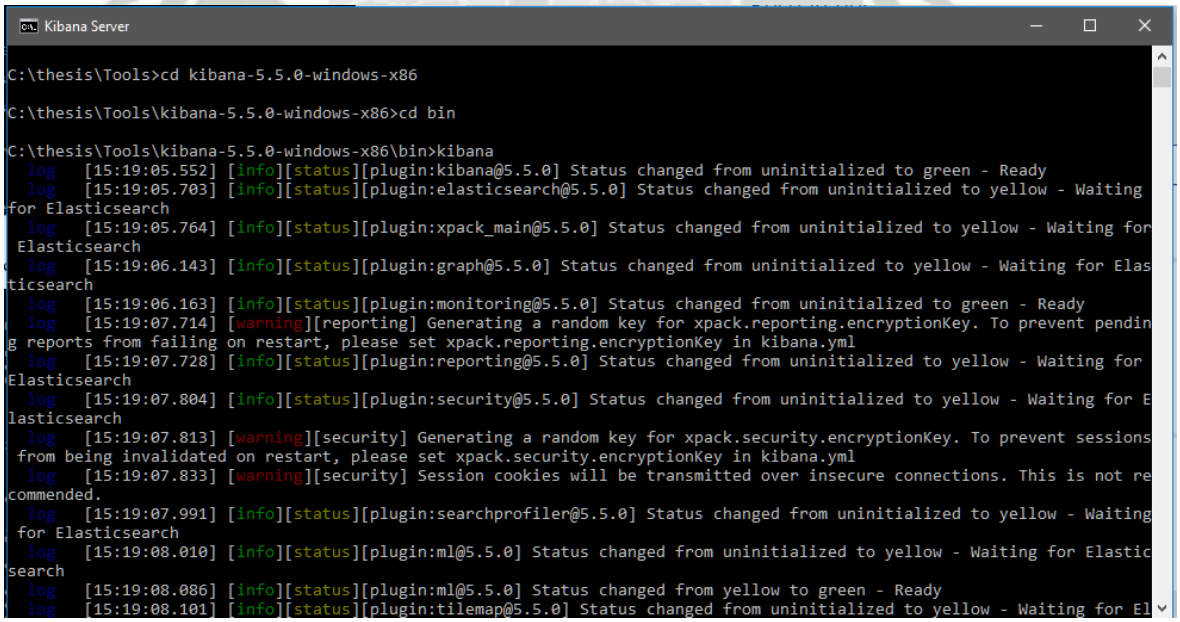
Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA



– El servicio tardara unos pocos segundos e iniciara

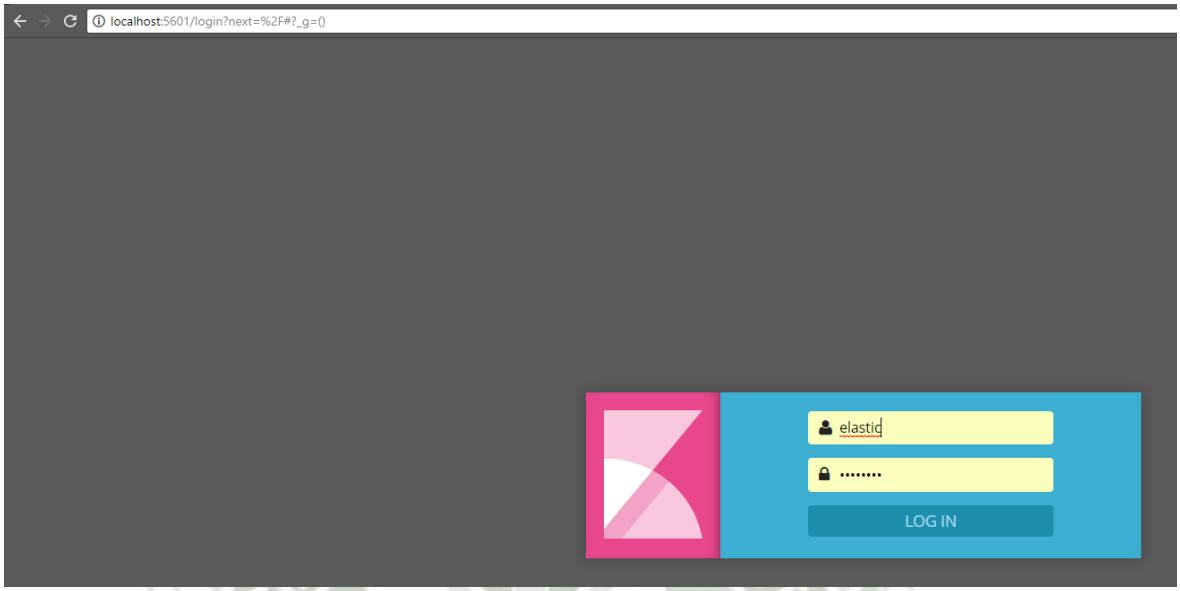


Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

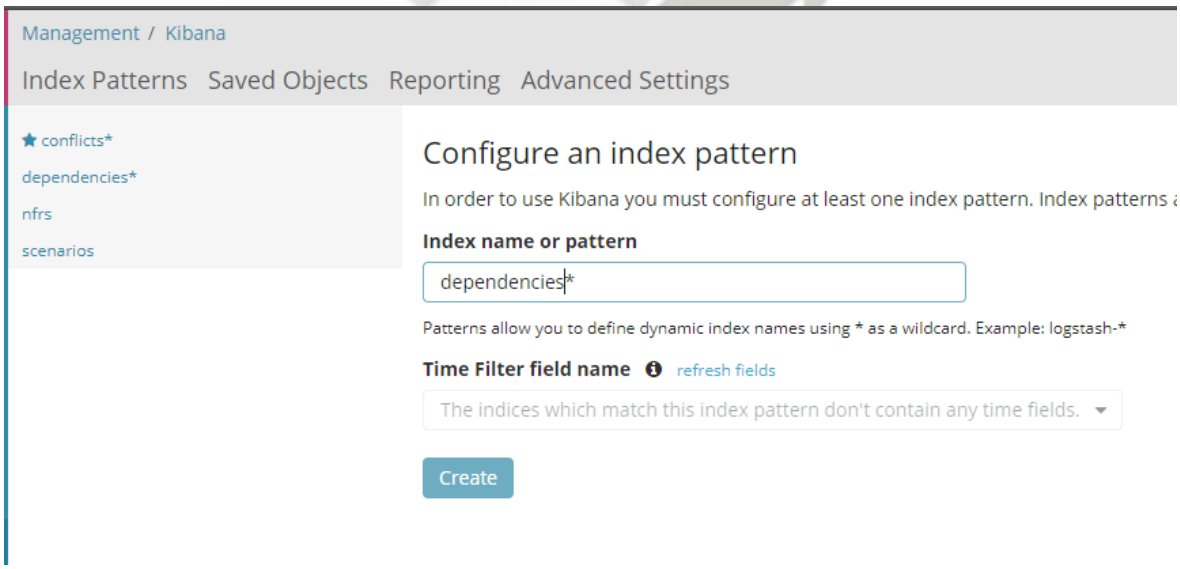
UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

– Para comprobar que el servicio este levantado con éxito, nos dirigimos a su dirección por default: localhost:5600 y nos tendría que aparecer algo asi:



4. Interfaces

4.1 Management: aquí realiza su configuración de Kibana en tiempo de ejecución, incluida la configuración inicial y la configuración continua de patrones de índice, configuraciones avanzadas que modifican los comportamientos de Kibana y los diversos "objetos" que puede guardar en Kibana, como búsquedas, visualizaciones y tableros.

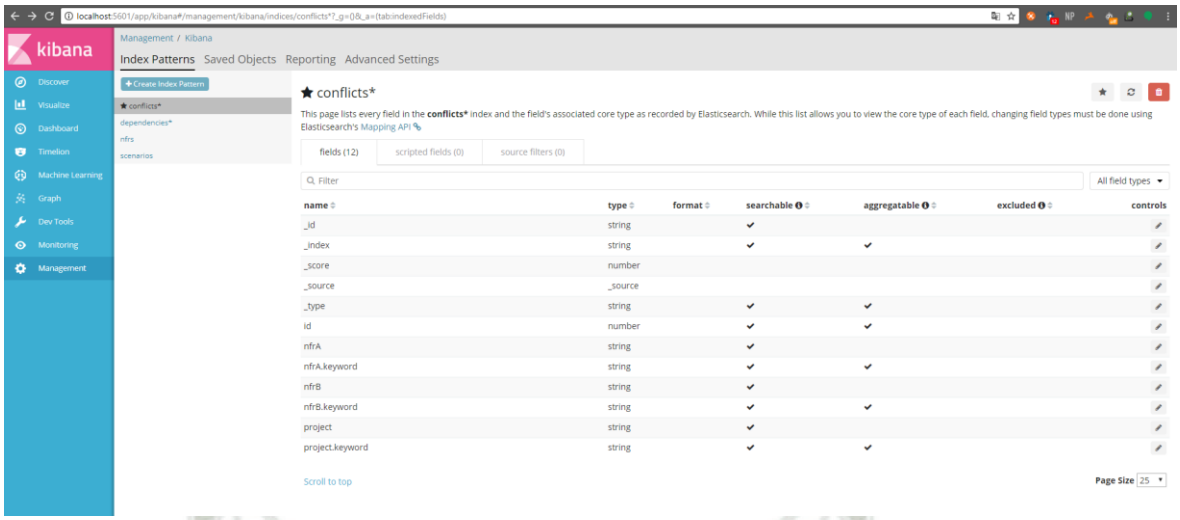


Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

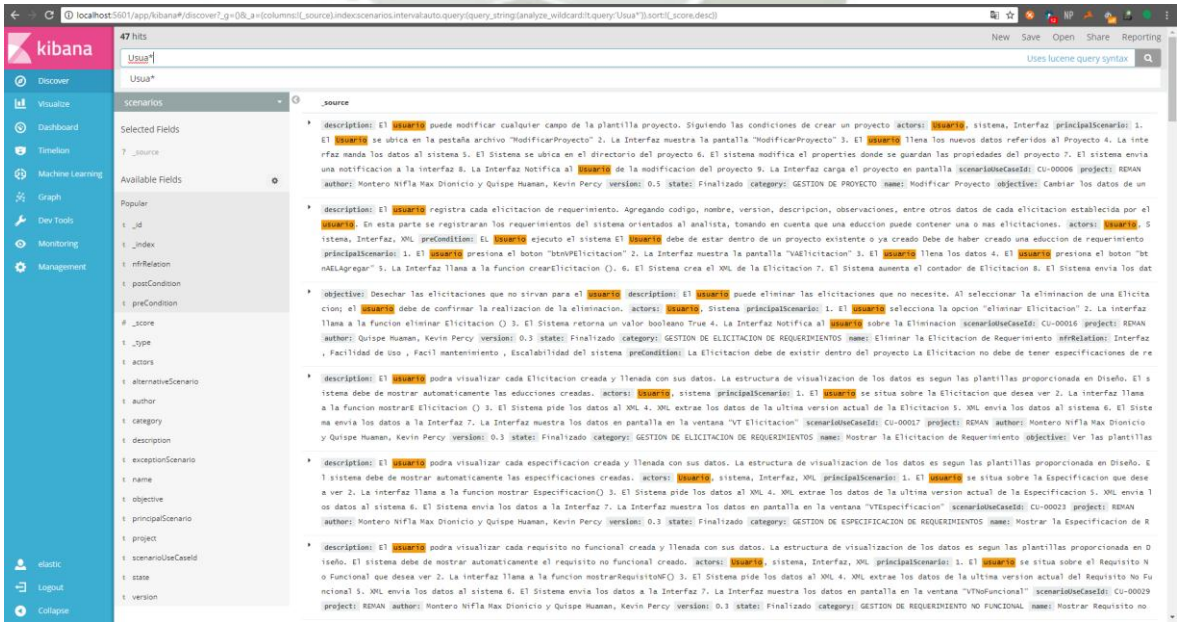
UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

De esta manera se tendrá 4 índices de elasticsearch conectados con kibana



4.2 Discover

Se puede explorar de forma interactiva sus datos desde la página Descubrir. Se tiene acceso a cada documento en cada índice que coincida con el patrón de índice seleccionado. Puede enviar consultas de búsqueda, filtrar los resultados de búsqueda y ver los datos del documento. También puede ver la cantidad de documentos que coinciden con la consulta de búsqueda y obtener estadísticas de valores de campo. Si se configura un campo de tiempo para el patrón de índice seleccionado, la distribución de documentos a lo largo del tiempo se muestra en un histograma en la parte superior de la página.



Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

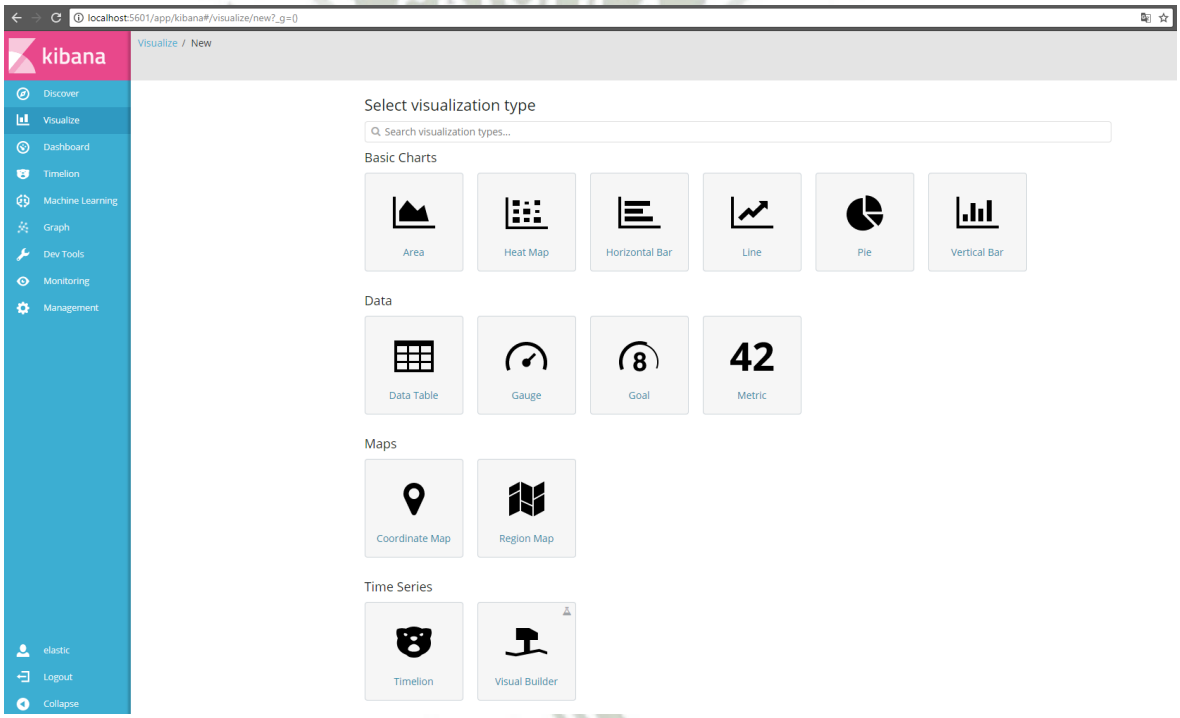
UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

4.3 Visualize

Visualize le permite crear visualizaciones de los datos en sus índices.

Las visualizaciones de Kibana se basan en consultas de Elasticsearch. Al usar una serie de agregaciones de Elasticsearch para extraer y procesar sus datos, puede crear gráficos que le muestren las tendencias, los picos y los saltos que necesita conocer.

- Graficos: área, mapas de calor, barras horizontales, lineales, pie, barras verticales
- Data: tablas de datos, metricas, gauge, goal
- Mapas: mapas coordinados, mapas de región
- Series de tiempo: timelion, visual builder.
- Otros: markdown, tag cloud



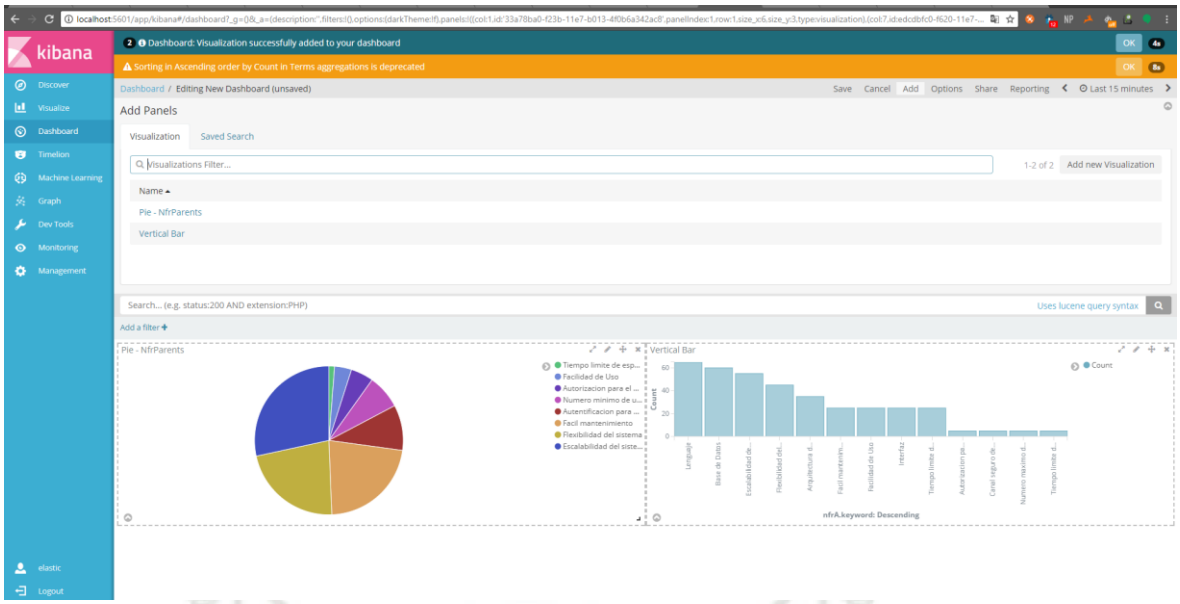
4.4 Dashboard

Un dashboard muestra una colección de visualizaciones previamente guardadas

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

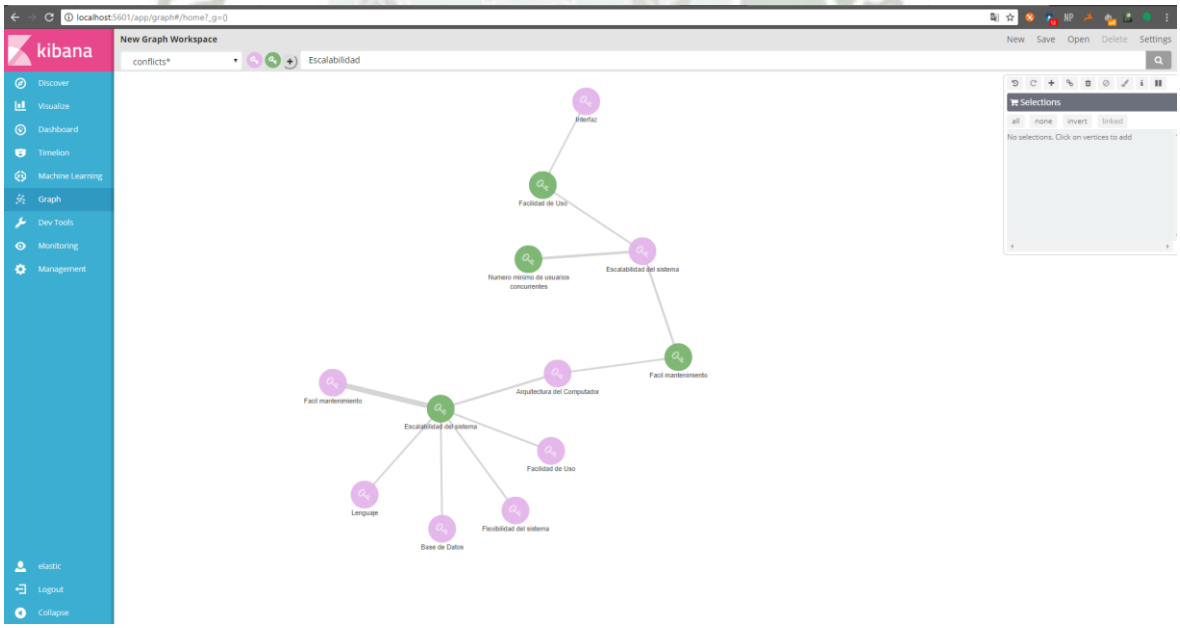
REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA



4.5 Graphs

Los grafos permiten descubrir cómo se relacionan los elementos en un índice Elasticsearch. Puede explorar las conexiones entre los términos indexados y ver qué conexiones son las más significativas. Esto puede ser útil en una variedad de aplicaciones, desde detección de fraude hasta motores de recomendación.



5. Enlaces

https://www.elastic.co/products/kibana

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

https://github.com/elastic/kibana/blob/master/LICENSE.md

https://www.timroes.de/2015/02/07/kibana-4-tutorial-part-1-introduction/

https://www.digitalocean.com/community/tutorials

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis