Universidad de las Ciencias Informáticas. Facultad 3.

Título: Estrategia para el desarrollo de requisitos no funcionales de software en proyectos productivos de la Universidad de las Ciencias Informáticas.

Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informáticas.

Autores: Lisett Pérez Quintero
Carlos Manuel Verdecia Guerra.

Tutora: MSc. Karina Pérez Teruel
Tutor asesor: MSc. José Raúl Rodríguez Galera

Ciudad de La Habana, mayo de 2008

Declaración de Autoría:

Nosotros, Lisett Pérez Quintero y Carlos Manuel Verdecia Guerra declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los ___días del mes de _____del año____.

_____ _____

Lisett Pérez Quintero Carlos Manuel Verdecia Guerra

_____ _____

Karina Pérez Teruel José Raúl Rodríguez Galera

"Hombre es algo más que ser torpemente vivo:

es entender una misión, ennoblecerla y cumplirla."

José Martí

AGRADECIMIENTOS

Unas letras, pero quiero sepas que son más que letras lo que en este mensaje enmiendo, porque redacto lo que mi corazón confiesa eterno después de tantos años de estudio y esfuerzos inolvidables.

Eso eres tú, quien a mi lado estuvo, sumándole a mi vida las más sinceras e intensas emociones, quien en su mirada me dio apoyo a la razón de continuar luchando por los sueños que siempre he defendido, y he juntado tanto amor, que en nombre de ese amor quiero que cada una de esas personas especiales sepa que agradezco haberle conocido la hermosura de su alma.

Enmiendo a ellas más que mis agradecimientos, mi amor infinito, según dice Dulce María Loynaz: “La palabra noble es ciertamente un indicio; la obra útil es ya una esperanza. Pero solo el amor revela- como a golpe de luz- la hermosura de un alma.” A esas personas, muchas gracias, especialmente por existir y haberlas conocido.

A la Revolución artífice de tantos admirables proyectos y a Fidel, por habernos dado la maravillosa oportunidad de participar en esta obra de infinito amor, la Universidad de las Ciencias Informáticas.

A mis padres, por su constante amor y apoyo, por su confianza y aliento. A ellos por enseñarme con tanto esmero, dedicación y principalmente por la inspiración en ese ejemplo tan admirable que son, y significan para mí.

A mis tíos y primos, por su cariño, especialmente a mi tío Fernando Quintero, por su amor, por su trayectoria y aporte en el desarrollo de la computación. Y a sus hijos por ser más que mis primos mis hermanos de siempre.

A mis abuelos, por el cariño recibido, siempre presente. A mi abuela Rosa por su ternura, y enseñarme a admirar aun más la magia del arte y su bello sentido en nuestra vida. A mi abuelo Juanito, por el cariño que me entregó durante estos años en La Habana.

A María de Los Ángeles Alvares, por ser también mi familia, por el inmenso cariño compartido con mis padres y su constante apoyo. Por su abrazo siempre sincero.

A mis tutores Karina y José Raúl, por todo su apoyo, su dedicación, por darnos la oportunidad de aprender con ellos tantas cosas admirables y por ser además unos maravillosos amigos.

A mis profes por todas las enseñanzas inolvidables.

A mis amigos, por su lealtad, apoyo, por compartir tantos momentos imborrables y bellos.

A mi compañero de tesis, Carlos. A mi guitarra por su compañía aún cuando estaba lejos. A la poesía. Y al amor, que siempre logra engendrar la maravilla.

Lisett

DEDICATORIA

A la Revolución, que ha hecho posible la realización de tantas obras de infinito amor como lo es nuestra admirable Universidad de la Ciencias Informáticas.

A mis padres, por todo ese maravilloso amor infinito y constante, por la inmensa admiración que les tengo. A ellos, por apoyarme tanto en la defensa de mis sueños y por enseñarme con tanta dedicación y ternura. Por constituir para mí ejemplos a seguir y darme en sus sonrisas tanta inspiración. Las palabras no me alcanzarían para describirles tantas razones que mi corazón contiene para adorarlos tanto.

Lisett Pérez Quintero

AGRADECIMIENTOS

Por su paciencia, dedicación, consagración, por sus consejos, por escucharme, por las madrugadas de trabajo, quiero agradecer por todo a mi amiga y compañera de tesis, Lisett.

A mis padres, Sara y Carlos, ellos son la razón por la cual estoy aquí hoy, gracias por sus consejos, por ayudarme y apoyarme en todo, por su preocupación. Todo lo que tengo se los debo a ustedes. Los quiero.

A mi hermano Néstor, por todo lo que has hecho por mí. Nunca podré pagártelo. Por ser otro padre para mí. Y a mi cuñada Isa, por todo tu apoyo y confianza.

A mi sobrina Karen, más que eso, haz sido una hermana para mí.

A mis tíos y primos. Principalmente al primo Jorge.

A mi novia Lourdes, supiste entenderme cuando más lo necesitaba.

A la hermana que conocí en la UCI, Caridad, sin ti no hubiera llegado al final. Estuviste conmigo siempre, en las buenas y fundamentalmente en las malas. Gracias Cary.

A mi hermana Angie, por todos estos años de amistad, aunque estés lejos, todos tus consejos fueron fuente de inspiración.

A mis hermanos del grupo 3112, Anabel, Iosmel y Yonelbys, en ustedes encontré una verdadera familia, espero seguir viéndolos.

A los amigos y hermanos de la vieja escuela. Rodolfo, Henry, Kappa, Kenia. Nada ha cambiado aunque estemos más viejos.

Al grupo 3301, nunca encontré grupo tan unido como ese.

Al "piquete de la pegada" y al equipo del apto 11205.

A los que estuvieron conmigo en los últimos momentos en la Facultad Regional de Artemisa. Gracias realmente.

A mis tutores, Karina y José Raúl, por todo lo que aprendí al lado de ustedes, por sus ideas y colaboración. Nunca los olvidaré.

A las viejas amistades de Holguín que me brindaron su apoyo en todo momento y confiaron en mí.

A todos aquellos profesores y compañeros que conocí en la UCI y no me alcanza la hoja para ponerlos y que de una forma u otra han sido parte de este logro.

Carlos

DEDICATORIA

Dedico este trabajo de diploma:

A mis padres, por todo su apoyo incondicional, por confiar en mí en todo momento, por su amor y dedicación. Por sus consejos. Y sobretodo por sentirme orgulloso de ser su hijo.

A mi hermano y mi sobrina, aunque estén lejos siempre han estado en mi corazón.

A mi novia.

Carlos Manuel Verdecia Guerra.

RESUMEN

La Universidad de las Ciencias Informáticas, es una universidad productiva que tiene la misión de producir software y servicios informáticos. Su modelo de formación vinculación estudio-trabajo, tributa a lograr este objetivo. Para obtener estos productos con la eficiencia y calidad deseada uno de los factores principales es desarrollar una adecuada ingeniería de requisitos. De acuerdo con ello es esencial el desarrollo (elicitación, análisis, especificación y validación) de los requisitos no funcionales del software siendo estos cualidades y restricciones operativas que el sistema de software debe cumplir. Dada la significación que tienen, es necesario elaborar una estrategia para el desarrollo de los requisitos no funcionales aplicables a proyectos productivos de la Universidad de las Ciencias Informáticas.

Para la realización de la investigación se realizó el estudio de elementos, conceptos, ideas, que tributan a la eficiencia y calidad de procesos involucrados en la ingeniería de requisitos, principalmente para el caso de los requisitos no funcionales de software. Teniendo en cuenta las tendencias internacionales se hizo un estudio de las condiciones reales en que se efectúan estas actividades en los proyectos productivos de la universidad, identificando estas condiciones mediante métodos empíricos que sugiere la metodología de la investigación científica. Se obtuvo como resultado de la investigación una estrategia para el desarrollo de requisitos no funcionales en la Universidad de las Ciencias Informáticas con el objetivo de contribuir al éxito y desarrollo eficiente de estos proyectos logrando un aporte a la calidad que estos procesos determinan.

PALABRAS CLAVES:

Requisito de software, Requisito no funcional de software, Estrategia.

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

La ingeniería de requisitos se ha convertido en el mundo desde hace algunos años, (principalmente desde hace menos de una década), en una de las áreas más interesantes y de gran actividad en la ingeniería de software. Estableciéndose un reconocimiento cada vez más generalizado de la importancia de la misma y de los riesgos en que se incurren si esta se realiza de forma incorrecta, o insuficiente en determinados proyectos de software.

Ese auge es entonces visto como algo lógico al considerarse que la ingeniería de requisitos tiene como objetivo establecer los principios y métodos para la identificación de los requisitos de un sistema de software. (Alan M. Davis, Fellow, IEEE, Oscar Dieste, Member, IEEE, Ann M. Hickey, Member, IEEE, Natalia Juristo, Fellow, IEEE and Ana M. Moreno, Senior Member, IEEE)

En la ingeniería de requisitos se identifica el propósito del sistema, teniendo en cuenta que significa una columna vertebral de las ideas que quieren realizarse y verse logradas en el software que se espera producir, identificando además el contexto en el que será usado.

Esas necesidades interpretadas para dar paso a la creación del software que se solicita, deben ser comprendidas de la manera más eficiente posible, de forma que se convierta en una descripción precisa, consistente, y con la mayor completitud posible para su entendimiento que incluye la adecuada documentación para un seguimiento exitoso. Esto conlleva a que la ingeniería de requisitos proponga actividades que en su conjunto posibiliten lograr con calidad el desarrollo de este proceso.

Estos requerimientos o requisitos de software que son confirmadamente a través del tiempo una pieza fundamental en el proceso de desarrollo de software, dan lugar a la existencia del los llamados requisitos no funcionales y los requisitos funcionales de software.

Los requisitos funcionales son declaraciones de los servicios que debe proporcionar el sistema. Especifica la manera en que este debe reaccionar a determinadas entradas e incluso pueden declarar explícitamente lo que el sistema no debe hacer. (Kybele, Grupo de investigación, 2007)

Los requisitos no funcionales no se refieren a funcionalidades específicas que debe tener el sistema. Son muchas veces interpretados como características o condiciones que el sistema debe cumplir. Constituyen desde un punto de vista restricciones dirigidas a los servicios o funciones ofrecidas por el sistema, por ejemplo teniendo en cuenta el tiempo de respuesta, la capacidad de almacenamiento, la integridad, etc. Son de gran importancia para el sistema en su totalidad, constituyendo aspectos significativos para su arquitectura, y surgen de las necesidades del usuario. Se plantea que generalmente estos requisitos suelen ser más críticos que los funcionales, pues si no son satisfechos el sistema no resulta útil. Se requiere de una identificación correcta y seguimiento de los mismos. (Kybele, Grupo de investigación, 2007)

Mundialmente, “se introdujo un avance significativo cuando los requerimientos no funcionales fueron tratados como objetivos competitivos extensivamente refinados y amenazantes entre cada uno como un intento para arribar a la solución aceptable. El NFR Framework es uno de los pocos trabajos que tratan con requerimientos no funcionales, iniciando desde etapas tempranas del desarrollo de software a través de una amplia perspectiva. El NFR Framework ve a los objetivos no funcionales como objetivos que pueden generar conflictos entre ellos y deben ser representados como objetivos del software a ser satisfechos. El concepto de softgoal (objetivo del software) fue introducido para alcanzar con la naturaleza abstracta e informal del requerimiento no funcional.” (DIGIÓN, Leda Beatriz, 2007)

CMMI, es un modelo de guía para la mejora o evaluación de los procesos de desarrollo y mantenimiento de sistemas y productos de software, utiliza veintidós aéreas de procesos para evaluar y mejorar, una de estas aéreas precisamente, se encarga del desarrollo de requisitos, correspondiente a la categoría de ingeniería, y se localiza en el nivel 2 Repetible. Dentro de ello involucrados los requisitos funcionales y los no funcionales, en etapas de elicitación, análisis, especificación, y validación.

La UCI está estrechamente vinculada a la ingeniería de requisitos como uno de los aspectos esenciales a abarcar en los proyectos productivos para el desarrollo exitoso de los mismos, todo esto mediante una serie de técnicas y métodos que tributan a la calidad del proceso productivo de software en la universidad. Una correcta aplicación de la ingeniería de requisitos

enmarca tanto a clientes como a desarrolladores en el entorno que se desenvolverán dentro del ciclo de vida del proyecto que se efectúe.

Aunque se pueden identificar métodos usados en los proyectos que tributan al eficiente desempeño en torno a este factor, en el desarrollo de los requisitos no funcionales (elicitación, análisis, especificación, y validación), desempeñan un papel positivo pero se identifican en algunos casos como insuficientes pudiendo incluso influir negativamente en la calidad de los procesos de la ingeniería de software que se desarrolla en los mismos.

Es frecuente escuchar entre los conocedores del mundo del software que muchos proyectos en el mundo fracasan por el incorrecto tratamiento a los requisitos, realizando un inadecuado desarrollo de los mismos.

Al remitirse a proyectos productivos de las facultades de la universidad para percibir cuestiones que pudiesen significar en relación a la situación descrita se observan casos que los que los lenguajes, técnicas y herramientas mayormente empleadas en el desarrollo de los requisitos no funcionales se identifican como elementos que pudieran potenciarse, e incluso fortalecerse, al encontrarse en un camino donde la calidad en avanzada de la producción en los proyectos incluye este factor.

En vista de esto, se consolidan estrategias, alternativas en los proyectos para realizar las etapas del desarrollo de los RNF, pero reconociéndose la necesidad de modelos para ello que satisfagan la orientación en pasos para esta área de la ingeniería de requisitos que muy pocas veces cuenta con amplios marcos para su comprensión y facilidades de modelado a través de técnicas universales u estándares. Además de la existencia de un predominio de ideas de considerarse la necesidad de potenciar las técnicas empleadas en la elicitación, el análisis, la especificación, y la validación, reforzándose las mismas en la construcción de modelos que estandaricen las consideraciones más empleadas a modo de buenas prácticas en los proyectos productivos.

Por tal causa es intención de esta investigación buscar alternativas a modo de estrategia que permitan desarrollar correctamente los requisitos no funcionales que se dan lugar en los

procesos de desarrollo de software de proyectos productivos de la Universidad de las Ciencias Informáticas.

Todo esto da paso a presentar como **problema científico** de la investigación la cuestión de ¿cómo contribuir al desarrollo de los requisitos no funcionales en proyectos productivos de la Universidad de las Ciencias Informáticas?

El **objeto de estudio** es la ingeniería de requisitos, y el **campo de acción** está dirigido a las estrategias utilizadas para el desarrollo de los requisitos no funcionales.

Los autores determinan el siguiente **objetivo:** Elaborar una estrategia para el desarrollo de los requisitos no funcionales aplicables a proyectos productivos de la Universidad de las Ciencias Informáticas.

Hipótesis: Si se aplica una estrategia correctamente concebida para el desarrollo de los requisitos no funcionales en los proyectos productivos de la UCI se contribuirá al desarrollo eficiente de estos proyectos logrando un aporte a la calidad que estos procesos determinan.

Tareas a desarrollar:

- Sistematización del estado del arte en los lenguajes, técnicas y herramientas mayormente empleadas para la captura de requisitos no funcionales y trabajo con los mismos.
- Aplicación de técnicas de recopilación de información para determinar el estado actual de la situación identificada.
- Diseño de la estrategia que responde al objetivo de la investigación.
- Validación de la estrategia para el desarrollo de los requisitos no funcionales aplicables en los proyectos productivos de la universidad.

Para llevar a cabo esta investigación se utilizarán los siguientes **métodos**:

Métodos Teóricos:

Análisis-Síntesis para la elaboración de lo que constituye el marco teórico referencial así como la propuesta presentada a modo de estrategia.

Inducción-Deducción para la identificación de aspectos e ideas fundamentales así como el trabajo en función del procesamiento del objetivo de la investigación.

Histórico-Lógico para el análisis de la trayectoria completa del fenómeno, su condicionamiento a los diferentes periodos de la historia, esto revela las etapas principales de su desenvolvimiento y las conexiones históricas fundamentales, y basarse en el estudio histórico del fenómeno. (Hernández León, et al., 2002)

Método Empírico:

Medición como procedimiento que se realiza con el objetivo de obtener información numérica acerca de una propiedad o cualidad del objeto donde se comparan magnitudes medibles y conocidas.

Métodos más específicos para la recolección de datos:

- **Entrevistas.**
- **Encuestas.**

Este trabajo tiene como aporte esencial la propuesta de una estrategia que potencie al carácter eficiente y la calidad de la ingeniería de requisitos (requisitos no funcionales) usada en proyectos productivos, y que tribute a la gestión del conocimiento de los recursos humanos profesionales y estudiantes vinculados a estos proyectos, teniendo en cuenta el intercambio de experiencias y el seguimiento de buenas prácticas en torno a esta rama de la ingeniería.
El trabajo de diploma cuenta con introducción, dos capítulos, conclusiones, recomendaciones, bibliografía, anexos y glosario de términos.
El capítulo uno es la fundamentación teórica de la investigación, que contiene aspectos relacionados a los conceptos de ingeniería de requisitos, en el caso de los requisitos no funcionales, el trabajo con los mismos en el proceso de desarrollo de software y aspectos relacionados a estos.
El capítulo dos contiene la descripción de la situación existente analizada y la propuesta de estrategia para el desarrollo de los requisitos no funcionales en proyectos de la UCI.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción.

En este capítulo se hace referencia a aspectos relacionados con la ingeniería de requisitos, fundamentalmente para los requisitos no funcionales de software. Dando lugar a la descripción de conceptos relacionados con los mismos, la importancia que tienen en el proceso de desarrollo de software, y el equilibrio que puede establecerse entre ellos. Se detalla en qué consiste el desarrollo de requisitos de software, basado en el modelo de capacidad y madurez CMMI. Se argumentan las actividades que propone este desarrollo, actividades tales como elicitación, análisis, especificación y validación. También se puntualiza en este capitulo sobre las técnicas, herramientas y lenguajes que se utilizan a nivel mundial para el desarrollo de los requisitos no funcionales. Todo esto teniendo en cuenta posibles errores que pueden cometerse, la repercusión que pueden tener esos errores en el producto software que se está desarrollando en un determinado proyecto, y cómo se puede llevar a cabo una buena práctica de esta en contribución a la calidad y la eficiencia del sistema a construir.

1.2 Requisitos de software.

Primeramente se debe entender lo qué es un requisito. La real academia de la lengua española ha definido el término “requisito” como una “circunstancia o condición necesaria para algo”. Muchas veces se escucha hablar para lo que supone ser un mismo significado con las palabras “requisitos” y “requerimiento”, y aunque gramaticalmente puedan sugerir la misma interpretación no es menos cierto que muchas personas prefieren usar la primera palabra argumentándose en lo personal que les parece una palabra más apropiada, más correcta, y además más corta y suave de tomar una idea y tomar como sustantivo esta idea de algo que es requerido y por tanto al ser requerido se convierte en requisito. El uso de la palabra “requisito” da lugar a la interpretación, y abstracción de la idea de algo que se espera cumplir, por tanto un requerimiento muchas veces se interpreta como algo para que alguien lo tome como requisito, pero si algo es identificado desde un principio como una cuestión que es necesaria cumplir es anotada desde entonces como un requisito.

Los requisitos de software estarían entonces determinados por las necesidades del usuario, lo que él describe como lo que debe cumplir el sistema que solicita, las características que lo deben identificar y sus funcionalidades en algunos casos. Se identifica como servicios u obligaciones que debe cumplir el sistema.
Los requisitos son fundamentales en su aporte visible en las trazas que comunican elementos que se van desarrollando una vez que el tiempo va avanzando y determinan de tal forma una vía de control que sostiene en gran nivel las posibilidades de cambios que pudiesen ocurrir si son analizados correctamente y con el adecuado grado de abstracción que conlleva a su objetividad. Por tanto, la necesidad de que estas actividades se realicen con la madurez y responsabilidad adecuada garantizaron el advenimiento de lo que se conoce como "ingeniería de requisitos".

La ingeniería de requisitos consiste en un conjunto de actividades y transformaciones que pretenden comprender las necesidades de un sistema software y convertir la declaración de estas necesidades en una descripción completa, precisa y documentada de los requerimientos del sistema siguiendo un determinado estándar. (Marqués, 2005)

Existen las llamadas "Cuatro P" valoradas de gran importancia en la gestión de proyectos de software (personal, producto, proceso, y proyecto) estas asumen papeles protagónicos dentro del camino al éxito del software que se desea realizar. El personal debe sentirse motivado, entusiasmado en las expectativas de hacer un software de alta calidad, y estar organizados en equipos con una buena comunicación lograda a través de una coordinación adecuada. "Los requisitos del producto deben comunicarse desde el cliente al desarrollador, dividirse (descomponerse) en las partes que los constituyen y distribuirse para que trabaje el equipo de software". (Pressman, 2002)

Por lo que a través de los años se ha podido constatar que los requerimientos o requisitos son la pieza fundamental en un proyecto de desarrollo de software, pues marcan un punto de partida para actividades como la planeación, básicamente en lo que se refiere a las estimaciones de tiempo y costo, así como la definición de recursos necesarios y la elaboración de cronogramas. Son una forma de verificar si se alcanzaron o no los objetivos establecidos

en el proyecto, pues son un reflejo detallado de las necesidades de los clientes o usuarios del sistema. (Arias Chaves, 2007)

El glosario estándar de ingeniería de software de la IEEE (IEEE Standard, 1990) presenta la definición más completa de un requisito.

1. Condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo.
2. Condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente.
3. Una representación documentada de una condición o capacidad como en 1 o 2.

Es importante tener en cuenta que los requisitos de software deben ser concisos para poderse entender claramente y no haya confusiones respecto a su interpretación, deben ser completos y consistentes y sin ambigüedades que tiendan a confundirlos. Deben ser posibles de probar y de verificar, pues de lo contrario daría lugar a una idea no posible de concretar. Deben especificarse por escrito, como contrato que evidencia un acuerdo entre dos partes.

1.2.1 Clasificación de los requisitos de software.

Existen dos tipos de requisitos, los requisitos de usuario, son descripciones ya sea en leguaje natural o diagramas de lo que se espera que el sistema posea así como las restricciones que podrá presentar el mismo. También están los requisitos del sistema, los cuales establecen con más detalle las funciones, servicios y restricciones operativas que el sistema tendrá. Estos requisitos se conocen como requisitos de software, y se pueden dividir en dos partes: requisitos funcionales o requisitos no funcionales, los que tienen diversas clasificaciones que se abordarán más adelante en este trabajo. Dentro del sistema se encuentran también los requisitos del dominio, los cuales surgen del dominio de la aplicación del sistema y no de las necesidades del usuario. Estos se suelen tratar como otros requisitos funcionales o no funcionales, pues reflejarán características y restricciones del dominio de la aplicación. (Straud Barros, 2005)

"Los requisitos funcionales (RF) describen lo que el sistema o el software deben hacer. Esta funcionalidad es la capacidad proporcionada y utilizable por uno o más componentes de un sistema, en ocasiones se le llaman a los requisitos funcionales conductuales u operacionales, ya que estos especificarán las entradas (estímulos) al sistema, los rendimientos (contestaciones) del sistema y las relaciones conductuales entre ellos. Los RF en ocasiones especifican los que el sistema no debe hacer." (Young, 2004). Constituyen sentencias sobre los servicios que ha de proporcionar el sistema, así como debe reaccionar el sistema ante determinadas entradas del usuario. Los requisitos no funcionales (RNF) son propiedades o cualidades que el producto debe tener (Young, 2004)

1.3 Requisitos no funcionales.

Los RNF especifican propiedades del sistema que el sistema debe cumplir, como restricciones del entorno o de la implementación, rendimiento, dependencia de la plataforma, facilidad de mantenimiento, extensibilidad, fiabilidad. (Jacobson, y otros, 2000) Tienen que ver con las características del sistema, que incluye también interfaces de usuarios.

Una de las clasificaciones más comunes que se asumen en proyectos productivos a nivel mundial está dada en aspectos como rendimiento/capacidad, seguridad, fiabilidad, portabilidad, mantenibilidad/ escalabilidad, reusabilidad, amigabilidad e interfaces (Siglo21, Informática, 2006)

Según Wiegers, los RNF son difíciles de expresar, los errores en el proceso de identificación y seguimiento de los RNF son muchas veces los más costosos de resolver. Por el grado de dificultad que implica verificar estos requisitos se evalúan subjetivamente. (Wiegers, 2003)

Sommerville (Sommerville, 2004) describe los RNF como restricciones sobre los servicios y funcionalidades ofrecidas por el sistema. Incluyendo restricciones en el tiempo que se debe demorar un proceso, restricciones sobre el proceso de desarrollo y estándares. Los RNF se aplican usualmente sobre el sistema como un todo, no a características o servicios particulares del sistema.

Los RNF no modifican la funcionalidad del producto, y constituyen de manera convincente parte de de la razón fundamental del producto.

1.3.1 Clasificación e importancia.

Se hace complicado determinar una clasificación estándar para los RNF, pues existen diversas clasificaciones dadas por el equipo de trabajo que se enfrenta a resolver los mismos, incluso suele encontrarse el término "tipo de requisito no funcional". Entre estas clasificaciones la correspondiente a los atributos de calidad de software juega un papel fundamental, dichos atributos contribuyen a la elección correcta de la arquitectura del software. Estos se agrupan en dos grupos, el grupo de los atributos importantes para los usuarios y los que son importantes para los desarrolladores.

Karl E. Wiegers presenta una completa descripción de estos atributos, donde destaca que los importantes para los usuarios son (Wiegers, 2003):

- *Disponibilidad (Availability)*: es una medida de tiempo planeado durante el uso del software, en el cual el sistema es disponible y completamente operacional, así como la manera en que podrá recuperarse el sistema ante fallos encontrados. Disponibilidad abarca también confiabilidad, mantenibilidad e integridad.
- *Eficiencia (Efficiency)*: es la medida en que el sistema aprovechara correctamente los recursos que presenta, como la capacidad del procesador, espacio en disco, memoria, y ancho de banda, etc. Si el sistema no distribuye de manera eficiente estos recursos el funcionamiento no será el esperado por el cliente. Está relacionada con el funcionamiento (performance).
- *Flexibilidad (Flexibility)*: también conocida como extensibilidad y expansibilidad, es la medida de cómo se podrán hacer variaciones en el producto final, o sea, de cómo el sistema se adaptará a nuevos cambios funcionales, cambios que se pueden contemplar en el diseño del software. Este atributo se pone de manifiesto principalmente en productos de muchas iteraciones e incrementos en las funcionalidades.
- *Integridad (Integrity)*: muy relacionado con la seguridad del producto, se encarga de bloquear el acceso a personal no autorizado a interactuar con el sistema, tanto

personas como virus, además de mantener la integridad de la información manejada por usuarios del sistema. Es de vital importancia en el software que se encuentra en aplicaciones web.

- *Interoperabilidad (Interoperability)*: indica como el sistema puede intercambiar información o servicios con otros sistemas, datos relacionados y semejantes. Es la capacidad de migrar que tendrá el software a otro sistema.
- *Confiabilidad (Reliability)*: es la probabilidad de que el software se ejecute por un período del tiempo específico, sin fallas. Incluye el porciento de operaciones que se realizarán correctamente antes de que el sistema pueda fallar. La robustez es considerada en ocasiones como un aspecto de la confiabilidad.
- *Robustez (Robustness)*: es el grado en que el sistema continua funcionando ante fallas inesperadas, ya sean de componentes de hardware o software asociadas al sistema, o ante entradas invalidas de datos, es la posibilidad de recuperación de datos que presentara el sistema ante errores de usuarios o fallas de otras índoles.
- *Usabilidad (Usability)*: el software se dice usable o útil, cuando es fácil de manejar por los usuarios sin necesidad de una completa preparación para enfrentarse al sistema, es la ayuda que le brindará el sistema a los usuarios durante su uso, ayudará al esfuerzo que hará el usuario a la hora de entrar datos, y manejar las salidas del sistema.

Entre los más importantes para los desarrolladores se encuentran (Wiegers, 2003):

- *Mantenibilidad (Maintainability)*: indica que tan fácil puede ser modificar o corregir algún defecto en el software, representa como el software puede ser entendido, cambiado o probado de manera fácil. Este atributo está muy relacionado con la flexibilidad y la comprobabilidad (testability).
- *Portabilidad (Portability)*: es el esfuerzo requerido a la hora de migrar una parte del software de un ambiente de trabajo a otro. Es la habilidad de vincular o localizar un producto con características específicas.
- *Reusabilidad (Reusability)*: reusabilidad o reutilidad indica la conversión de un componente de software con funcionalidades determinadas a otros usos, a otras funciones. El software reusable debe estar bien documentado, debe ser modular, independientemente de las aplicaciones específicas o ambiente de funcionamiento.

- *Comprobabilidad (Testability)*: tambien conocido como verificabilidad, representa en qué medida los componentes del software o el producto como tal pueden ser probados con el objetivo de buscarle defectos al funcionamiento del mismo.

El grupo de desarrolladores colombianos "Informática Siglo 21" define las siguientes clasificaciones (Siglo21, Informática, 2006):

- *Rendimiento*: Están relacionados con tiempos de respuesta estimados, requeridos y esperados para la ejecución en línea de procesos del sistema, teniendo como base la plataforma tecnológica y escenarios específicos a los que en teoría el sistema estará expuesto y frente a los que deberá responder.
- *Fiabilidad*: Están interrelacionados con la capacidad del usuario para confiar en las respuestas del sistema, en un sentido técnico, es decir, que la funcionalidad del sistema no se vea afectada por factores ajenos al sistema.
- *Disponibilidad*: Se relacionan con la capacidad del sistema para estar disponible para los usuarios, esto se refleja en el tiempo estimado, esperado y requerido por el usuario para que el sistema esté disponible.
- *Seguridad*: Requisitos relacionados con la confidencialidad de los datos en la transmisión y en el almacenamiento, junto con las necesidades del sistema para evitar intrusiones no autorizadas al mismo y la capacidad para seguir eventos que comprometan esta seguridad a través del tiempo.
- *Portabilidad*: Describen la capacidad del sistema para migrar de una plataforma hardware a otra sin que esto represente mayores traumatismos para el cliente del mismo.
- *Mantenibilidad*: Relacionados con la capacidad para realizar revisiones y cambios sobre la funcionalidad del sistema de manera que no represente una exagerada inversión en recursos el desarrollo del cambio mencionado, estos requisitos están orientados a consideraciones de diseño, de codificación, al uso de modelos de desarrollo, para lograr que el mantenimiento de sistema sea lo más natural posible.
- *Escalabilidad*: Hacen referencia a la capacidad del sistema para acoplar módulos componentes y extensiones, sugiere el empleo de tendencias en tecnologías que puedan permitir al sistema tener vigencia en su diseño por lo menos en un periodo de

tiempo considerable, de tal manera que los componentes de extensión que se propongan puedan ser desarrollados basados en tecnologías conocidas.

- *Reusabilidad*: Consideran la capacidad de los componentes del sistema de prestar servicios a otros sistemas, de tal manera que el componente como valor adicional al cumplimiento de sus funciones pueda prestar sus servicios a otros sistemas sin que esto implique modificaciones o redefiniciones en dicho componente.
- *Interfaces*: Contemplan las características del sistema respecto a intercomunicación con otros sistemas, ya sea a través de servicios o de salidas en archivo, definen los formatos y tecnologías para la exposición o la captura de información desde fuentes externas.
- *Amigabilidad (usabilidad)*: Requisititos que determinan las características generales de la capa de presentación del sistema en cuanto a las características de diseño gráfico de la misma, además de las facilidades para que el uso del sistema por parte del usuario final.
- *Capacidad (consumo de recursos)*: Consideran los requisitos de tecnología que permitirán al sistema ejecutar sus funciones de manera que responda a las expectativas del usuario en términos de eficiencia y eficacia en la respuesta de las operaciones. Se tienen en cuenta también las características, que harán que sea posible que el sistema funcione de manera estable durante un tiempo determinado, planeando crecimiento del mismo.

Hay otras clasificaciones como son las de Robertson & Robertson, agrupan los RNF en requisitos de Apariencia o Interfaz, Usabilidad, Rendimiento, Portabilidad y Mantenibilidad, Seguridad, Culturales y Políticos y finalmente Legales (Robertson, et al., 1999). La clasificación de Brito, Moreira y Araújo, mostrando Requisitos de Interfaz, de Desempeño, requisitos Operacionales, requisitos asociados al Ciclo de Vida del producto, de Seguridad, y requisitos Económicos, ellos destacan que la integración tardía de los requisitos no funcionales puede comprometer algunos principios de la Ingeniería de Software como la modularidad o reusabilidad. (Brito, et al., 2002). Malan y Bredemeyer se refieren a las restricciones y características, clasificándolos en Restricciones contextuales, de acuerdo a Características en Tiempo de Ejecución y durante el periodo de desarrollo. (Malan, 2001). Por

su parte McGraw relaciona los tipos de requisitos con la seguridad del producto, los clasifican en requisitos de Funcionalidad, de Usabilidad, de Eficiencia, de Tiempo de mercadeo, de Simplicidad y de requisitos Económicos. (McGraw, et al., 2004).

Finalmente se encuentra la clasificación hecha por Chung y Nixon, estableciendo una jerarquía de requisitos donde se encuentran como principales los requisitos de usabilidad, de desempeño, de seguridad y de costo. Dividiendo los de desempeño en tiempo de respuesta y de rendimiento; y en espacio de memoria principal y almacenamiento. Los requisitos de seguridad se desglosan en confidencialidad, integridad y disponibilidad. Donde los de integridad se dividen en consistencia interna e externa. (Chung, et al., 1999)

Importancia de los RNF.

Los requisitos de software, como su nombre lo indican son necesidades que presentan los usuarios ante la realización de algún software determinado, estos contribuyen a la aplicación de un correcto proceso de desarrollo sea cual sea la metodología que se utiliza para obtener el producto. Los RNF juegan su papel elemental dentro de dicho proceso, pues así como los requisitos funcionales tributan a funciones futuras que tendrá el producto, los RNF enmarcan el entorno donde se desenvolverá dicho software, ya que proveen las condiciones que deberá tener el mismo para un uso efectivo del software. Los RNF tienen una constante interacción con la propuesta de la arquitectura del sistema, pues estos determinan restricciones en el uso del software, restricciones que la arquitectura aporta en gran escala, determinando una adecuada arquitectura de software, antes que el sistema sea desarrollado, y de esta manera poder evaluarla. En el proceso de desarrollo de software un buen desarrollo de los RNF contribuye a la obtención de una buena calidad de software, pues como se abordó en este epígrafe, los atributos de calidad enmarcan propiedades que el software debe cumplir. Estos atributos de calidad del software interactúan entre sí, presentando mejoras o conflictos, teniendo en cuenta el peso que tienen en el resultado del producto final.

1.3.2 ¿Cómo se equilibran los RNF?

Wiegers señala que la combinación de ciertos atributos de calidad provoca compensaciones ineludibles. Usuarios y desarrolladores deben decidir qué o cuáles atributos son más

importantes que otros, por lo que deben interesar las propiedades de los mismos a la hora de hacer alguna decisión final. (Wiegers, 2003)

En el Anexo 1 se ilustra la interrelación que existe entre atributos de calidad, los que representan RNF del software, a pesar de que haya excepciones en algunos casos.

Donde se aprecia que el aumento en la calidad de un atributo afecta de manera positiva otro atributo, por ejemplo, las condiciones del diseño que hacen más portable al software, también lo hacen más flexible, con mayor facilidad de conexión con otros componentes, así como hacerlo más sencillo a la hora de probar y reutilizar. De forma contraria se muestra como el aumento de un atributo afecta negativamente la calidad de otros. Un aumento en la eficiencia, por ejemplo, influye de forma negativa en la mayoría de los atributos de calidad restantes. Similarmente los sistemas que se crean con ciertas facilidades de uso, siendo los mismos reutilizables, flexibles e interoperables con otros componentes de software, por lo general, presentan problemas de funcionamiento. También se presentan atributos que la contraposición que existe entre estos y otros atributos es mínima, de muy poco impacto, casi nula.

Se debe balancear las condiciones o restricciones del software contra las ventajas que este presentará en un resultado final. De esta manera se garantiza un buen trabajo. Para alcanzar un correcto equilibrio en las características del producto, se debe identificar, especificar y dar prioridad a las cualidades de cada atributo de calidad. Con el uso del Anexo 1 se podrá obtener mejores resultados en la definición de las cualidades que tendrá el software, algunos ejemplos son (Wiegers, 2003):

- Es difícil probar plenamente los requisitos correspondientes a la integridad de sistemas de máxima seguridad. La reutilización de componentes podría comprometer los mecanismos de seguridad.
- Código altamente robusto, decrementa la eficiencia porque las validaciones de datos y chequeo de errores se realizan con mayor frecuencia.

Finalmente, plantea que como es usual, a medida que las restricciones aumenten, así como las expectativas, se hace prácticamente imposible para los desarrolladores satisfacer totalmente los requisitos propuestos por los usuarios, por lo que debe hacerse un correcto análisis de cómo se afectan los requisitos entre sí y garantizar la calidad del producto final. (Wiegers, 2003)

1.4 Sobre el desarrollo de RNF

Las actividades que define la ingeniería de requisitos (IR) determinan una correcta gestión y seguimiento de los mismos, el desarrollo de requisitos, por su parte, comprende solo cuatro actividades. Dichas actividades pueden ser presentadas de varias formas. Pressman define que el proceso de IR puede ser descrito en cinco etapas o pasos, estos son: Levantamiento de Requisitos, Análisis y Negociación de Requisitos, Modelado de Actividades, Especificación de Requisitos, Validación de Requisitos y Gestión de Requisitos. (Pressman, 2002)

Dorfman y Thayer consideran que la ingeniería de requisitos incluye tareas de elicitación, análisis, especificación, validación y administración de requerimientos de software, siendo la "administración de requisitos de software" la planificación y control de todas esas actividades relacionadas. (Thayer, et al., 1997)

Dávila define que se puede dividir las prácticas de la IR en 4 actividades: Extracción, Análisis, Especificación, Validación. Especifica que como toda división de tareas, no es una estricta representación de la realidad, sino que se hace con el fin de sistematizar la realización de la IR. En general la delimitación entre una actividad y la otra no es tan clara, ya que están sumamente interrelacionadas, existiendo un alto grado de iteración y retroalimentación entre una y otra. (Dávila, 2001)

Desarrollo de requisitos, en cambio no comprende el seguimiento de los mismos, ni la gestión una vez que se han elicitado, analizados y evaluados. Este trabajo de diploma se enmarca en el desarrollo de requisitos según lo que plantea CMMI (Capability Maturity Model Integration).

El proceso de desarrollo de requisitos, según fuentes bibliografías consultadas de la Universidad Rey Juan Carlos (Kybele, Grupo de investigación, 2007) se describe como el

objetivo de crear y mantener un “documento” de requisitos del sistema. Además plantea que define el conjunto estructurado de actividades de cuya ejecución se obtiene y mantiene la especificación de los requisitos. Ilustrando entonces como el proceso de desarrollo de requisitos comprende (en general) cuatro etapas: identificación o captura de requisitos, análisis (y negociación) de requisitos, especificación o documentación de requisitos y validación de requisitos.

1.4.1 ¿Qué es el desarrollo de requisitos según CMMI?

CMMI, es un modelo de guía para la mejora o evaluación de los procesos de desarrollo y mantenimiento de sistemas y productos de software, o sea, lograr la optimación permanente en procesos de ingeniería. Es desarrollado por el Instituto de Ingeniería del Software de la Universidad Carnegie Mellon (SEI por sus sigas en inglés). Este modelo se hace efectivo a través de un camino de estados de evolución y aprendizaje llamados niveles de madurez: Inicial, Repetible, Definido, Administrado y Optimizante, organizados de manera ascendente en el orden que fueron mencionados. Estos niveles presentan diferentes aéreas de procesos, CMMI utiliza veintidós áreas de procesos para evaluar y mejorar la calidad del producto final. Una de estas áreas precisamente, se encarga del desarrollo de requisitos (RD, por sus siglas en inglés), correspondiente a la categoría de ingeniería, y localiza en el nivel 2 Repetible.

El equipo de Carnegie Mellon (CMMI Product Team, 2006) determina que el propósito del RD es producir y analizar requisitos del cliente, del producto, y de componentes del producto como tal. Explican que en esta área de proceso se describen tres tipos de requisitos: requisitos del cliente, requisitos del producto, y requisitos del componente del producto; incluyendo además tres metas específicas: el desarrollo de requisitos del cliente dirigido a definir un conjunto de requisitos del usuario, útiles para el desarrollo de los requisitos del producto. El desarrollo de requisitos de productos, dirigido a la definición de un conjunto de componentes de productos o productos como tal y por último el análisis y validación de requisitos, meta dirigida al análisis necesario de clientes, productos y requisitos de productos para definir, derivar, y entender los requisitos.
Plantean también que según CMMI los requisitos, son la base para el diseño, y el desarrollo de requisitos incluye las actividades siguientes (CMMI Product Team, 2006):

- Elicitación, análisis, validación y comunicación de las necesidades del cliente, de sus expectativas, y restricciones que se pueden obtener de los requisitos del cliente que constituyen el entendimiento de que lo que los stakeholders precisan.
- Colección y coordinación de las necesidades de los stakeholders.
- Desarrollo del ciclo de vida de los requisitos del producto.
- Establecimiento de los requisitos del cliente.
- Establecimiento del producto inicial y de los requisitos consistentes del producto con los requisitos del cliente.

Estas actividades se resumen en las etapas de elicitación, análisis, especificación y validación de requisitos.

1.4.2 Elicitación.

Dávila señala que esta etapa representa el comienzo de cada ciclo, afirmando que elicitación es el nombre comúnmente dado a las actividades involucradas en el descubrimiento de los requisitos del sistema. “Aquí, los analistas deben trabajar junto al cliente para descubrir el problema que el sistema debe resolver, los diferentes servicios que el sistema debe prestar, las restricciones que se pueden presentar, etc.” (Dávila, 2001)

Sin embargo, Pressman apunta que esta etapa parece ser fácil, pero, constituye posiblemente una de las más complicadas, pues existen algunos riesgos (Pressman, 2002):

- ***Problemas de alcance***: El límite del sistema está mal definido o los detalles técnicos innecesarios, que han sido aportados por los clientes/usuarios, pueden confundir más que clarificar los objetivos del sistema.
- ***Problemas de comprensión***: Los clientes/usuarios no están completamente seguros de lo que necesitan, tienen una pobre compresión de las capacidades y limitaciones de su entorno de computación, no existe un total entendimiento del dominio del problema, existen dificultades para comunicar las necesidades al ingeniero del sistema, la omisión de información por considerar que es “obvia”, especificación de requisitos que están en conflicto con las necesidades de otros clientes/usuarios, o especificar requisitos ambiguos o poco estables.

- ***Problemas de volatilidad***: Los requisitos cambian con el tiempo.

En definitiva, descubrir los requisitos del sistema no sólo implica preguntar a las personas qué quieren: es un proceso delicado. Se llega a la conclusión de que la elicitación debe ser efectiva, ya que la aceptación del sistema dependerá de cuan bien éste satisfaga las necesidades del cliente y de cuan bien asista a la automatización del trabajo.
La elicitación de los RNF está dada fundamentalmente por la realización de entrevistas a los clientes sobre las condiciones que presentará el sistema para su correcto funcionamiento. Es común dejar las tareas de elicitación de requisitos no funcionales a los arquitectos.

Durán y Bernández proponen en su metodología para elicitar requisitos de software una plantilla basada en patrones lingüísticos (patrones-L). Estos son frases que se utilizarán en algunos campos de la plantilla, frases que su uso es estándar en la especificación de requisitos. Esta plantilla se usa no solo en las reuniones de elicitación de requisitos sino también en el registro de los mismos. Con el uso de estos aspectos, la estructura de la información en forma de plantilla y la propuesta de frases estándar se facilita la redacción de los requisitos, permitiendo a los participantes en las actividades de elicitación centrarse en expresar sus necesidades y no en cómo expresarlas. La plantilla se muestra en el Anexo 2. El único campo específico de la plantilla es la descripción, en la que se usa un patrón–L que debe completarse con la capacidad que deberá presentar el sistema, el significado del resto de los campos es explicado en el Anexo 2. Y en caso de que no tenga sentido poner algún campo de la plantilla este se omite. (Durán Toro, et al., 2000)

1.4.3 Análisis

Luego de realizada la etapa de elicitación, y en la base de la misma comienza esta etapa. Pressman explica que en ella “los requisitos se agrupan por categorías y se organizan en subconjuntos, se estudia cada requisito en relación con el resto, se examinan los requisitos en su consistencia, completitud y ambigüedad, y se clasifican en base a las necesidades de los clientes” (Pressman, 2002)

También define algunas preguntas para el desarrollo exitoso de esta etapa, entre ellas se encuentran:

- ¿Cada requisito es consistente con los objetivos generales del producto?
- ¿Tienen todos especificado el nivel adecuado de abstracción?
- ¿Existen requisitos incompatibles con otros requisitos?
- ¿Se puede probar el requisito una vez implementado?

Arias Chaves (Arias Chaves, 2007) explica que se realiza un análisis luego de haber producido un bosquejo inicial del documento de requisitos; lo que lleva a lectura de los requisitos, conceptualización e investigación de los mismos. Se hace un intercambio de ideas con el resto del equipo, resaltando los problemas encontrados, proponiendo alternativas o soluciones.

Sumaro López (Sumaro López, 2001) plantea que el análisis de requisitos es un proceso de derivación de requisitos del sistema a través de la observación de sistemas existentes, discusión con usuarios y proveedores de información potenciales, análisis de tareas y así sucesivamente. Señala que “esto puede involucrar el desarrollo de uno o más modelos diferentes del sistema, que ayudan al analista a entender el sistema que será especificado. También pueden desarrollarse prototipos para ayudar a entender requerimientos” (Sumaro López, 2001).
Esta etapa se puede presentar compleja si el dominio del negocio es desconocido, pues en ella se descubren problemas con los requisitos del sistema identificados hasta el momento. De acuerdo a la complejidad del sistema a desarrollar y la envergadura de su negocio el analista puede unir la etapa de elicitación junto con la etapa de análisis del problema.

1.4.4 Especificación

En esta etapa se documentan los requisitos acordados con el cliente, en un nivel apropiado de detalle. En la práctica, se va realizando conjuntamente con el análisis, pero se podría decir que la especificación es "pasar en limpio" el análisis realizado previamente aplicando técnicas y/o estándares de documentación.

Para el desarrollo exitoso de la misma se confeccionan plantillas que recojan los datos específicos de los requisitos, en el Anexo 3 se muestra una plantilla propuesta por el grupo de desarrolladores Siglo 21 (Siglo21, Informática, 2006) Todas estas plantillas se recogen en un documento de especificación de requisitos. Un documento bien estructurado, coherente, que define de forma precisa y verificable los requisitos, el diseño el comportamiento u otras características de un sistema o componentes de un sistema. Se puede argumentar con toda seguridad que se conseguirán requisitos que sean presentados de una forma más consistente y más comprensible. No obstante en ocasiones es necesario buscar la flexibilidad cuando una especificación va a ser desarrollada.

En el Anexo 4 se presenta la estructura de una Especificación de Requerimientos de Software presentada por el estándar de documentación de IEEE 830 para la especificación desarrollada por Monferrer. Este estándar enmarca algunos criterios deseables; que se deben llevar a la hora de especificar los mismos (Monferrer Agut, 2001):

- No ambigüedad: cada definición tendría una sola interpretación, los términos serán claros y bien definidos.
- Completitud: se incluyen todos los requisitos significativos. No se produce la extracción de alguno para utilizarlo en definiciones futuras.
- Verificable: todas las características especificadas como parte del proyecto tienen una prueba correspondiente para determinar si han sido entregadas exitosamente.
- Consistencia: se ausenta la terminología conflictiva, las acciones requeridas contradictorias y las combinaciones imposibles.
- Modificable: se puede cambiar en caso de presentar redundancia, además de garantizar que el índice y el contenido estén correctos.
- Correcto: se debe tener cuidado de no incluir requisitos extraños o aquellos que pertenezcan a sistemas completamente diferentes.
- Clasificación: no todos los requisitos tienen la misma importancia, se pueden clasificar en dos criterios; importancia e estabilidad.
- Explorabilidad: el requisito será explorable si el origen de cada requisito es claro tanto hacia atrás (origen que puede ser un documento, una persona etc.) como hacia delante (componentes del sistema que realizan dicho requisito).

- Utilizable durante las tareas de mantenimiento y uso: el personal que no ha intervenido de manera directa en el desarrollo del software debe ser capaz de encargarse de su mantenimiento.

1.4.5 Validación

En esta etapa se ve el resultado del trabajo realizado, el cual es una consecuencia de la especificación del sistema e información relacionada, por lo se evalúa la calidad de la misma. Pressman dice que "la validación de requisitos examina las especificaciones para asegurar que todos los requisitos del sistema han sido establecidos sin ambigüedad, sin inconsistencias, sin omisiones, que los errores detectados hayan sido corregidos, y que el resultado del trabajo se ajusta a los estándares establecidos para el proceso, el proyecto y el producto." (Pressman, 2002)

"La validación representa un punto de control interno y externo; interno, porque se debe verificar internamente lo que se está haciendo, y externo, porque se debe validar con el cliente." (Dávila, 2001)

El equipo de desarrolladores Informática Siglo 21 plantea siete pasos para una correcta recopilación y validación de RNF, estos son (Siglo21, Informática, 2006):

1. Elaboración del guión de entrevistas, se define de forma previa la manera de documentar de los requerimientos no funcionales.
2. Validación interna de las versiones de los requerimientos no funcionales.
3. Desarrollo de las correcciones correspondientes a los documentos de acuerdo con las observaciones generadas.
4. Elaboración de la documentación de los requerimientos no funcionales.
5. Validación de los requerimientos no funcionales con los clientes
6. Entrega del documento de requerimientos no funcionales al cliente
7. Generación del documento de requerimientos no funcionales, con las correcciones de las observaciones y el complemento de los temas relacionados.

La validación de los requisitos también suele hacerse a través de la negociación de los mismos. Grünbacher y Seyff (Grünbacher, et al., 2005) demuestran que el proceso de negociación de requisitos (NR) no es un episodio como tal del proceso de desarrollo, se debe hacer en etapas avanzadas del desarrollo. El resultado de la negociación son las discrepancias que existen entre los stakeholders. Estos desacuerdos una vez identificados representan riesgos. Por lo que el objetivo principal de la NR es identificar y resolver estos conflictos entre los stakeholders. Contribuye además al éxito de definir los requisitos factibles que adaptan todas expectativas de los stakeholders. La NR está compuesta de tres fases, la pre-negociación, la negociación y la post-negociación. (Grünbacher, et al., 2005)

La pre-negociación es la fase que se encarga de definir el problema de la negociación, además de la identificación y solicitación de los stakeholders, así como las metas de ellos, también del análisis de dichas metas para encontrar posibles conflictos, el resultado de esta fase es que todos los conflictos estén encontrados. (Grünbacher, et al., 2005)

Desglosando la pre-negociación quedaría (Grünbacher, et al., 2005):

- Definición del problema: antes de que la negociación comience se debe identificar el problema, analizando la situación y definiendo el propósito de la negociación. Definir el problema de la negociación es esencial para la identificación de los stakeholders y para ajustar el método y técnicas de negociación que se utilizarán.
- Identificar stakeholders: constituye encontrar las personas cuyos intereses sean comunes. Identificando estas personas de manera correcta se acelera el proceso de la negociación.
- Elicitar metas: antes de que los conflictos puedan ser identificados los stakeholders deben presentar sus metas individuales. Estas metas son los objetivos del sistema.
- Analizar metas: las metas elicitadas son examinadas para identificar conflictos, o sea, se analizan las metas y preferencias de los stakeholders. El análisis de la meta revela no sólo conflictos entre metas del stakeholders, sino también revelan inconsistencias y riesgos.

La fase de negociación implica la conducta actual de la negociación y la definición de los acuerdos, basada sobre la elicitación de las metas y conflictos identificados de los stakeholders. Esta actividad estructura ediciones y desarrolla alternativas para resolver problemas. Luego se desarrollan soluciones factibles para los stakeholders, eventualmente de acuerdo con la mejor de ellas. La explicación de posibles soluciones es un pre-requisito ante que los stakeholders puedan coincidir con una decisión y requieran del establecimiento de un criterio unánime, lo que cual constituye un conjunto de reglas donde todos los stakeholders estén de acuerdo (Grünbacher, et al., 2005).

Post-negociación es la última fase. En ella los stakeholders (o herramientas automáticas) analizan y evalúan los resultados de la negociación y sugieren una re-negociación en caso de que fuera necesario. Por ejemplo, puede ser determinado que aunque el acuerdo actual satisfaga las preferencias de stakeholders se puede hacer uno nuevo. En modelos iterativos especialmente, los resultados de la negociación necesitan ser desarrollados como nuevas metas que pueden potenciar siempre la causa de nuevos conflictos. (Grünbacher, y otros, 2005)

1.5 Técnicas para el desarrollo de RNF

Para un correcto desarrollo de RNF se emplean diversas técnicas. A continuación se presentan técnicas utilizadas para ello, en este caso para elicitar los RNF:

1.5.1 Entrevistas y cuestionarios

Dávila explica que las entrevistas y cuestionarios se emplean para reunir información proveniente de personas o grupos, información que se obtiene conversando con el encuestado.

“Para la realización de las entrevistas debemos coordinar previamente la fecha y hora, y se debe realizar un plan de agenda, en el cual se hace en un punteo del objetivo de la entrevista. Por ejemplo, en la primera entrevista se establecerá un plan de comunicación, en el cual se intercambiarán los teléfonos, celulares, direcciones de e-mail y horarios de contacto.” (Dávila, 2001)

Es por ello que para realizar las entrevistas, conviene llevar preparado un cuestionario. En términos generales, un cuestionario consiste en un conjunto de preguntas presentadas a una persona para su respuesta. La forma de la pregunta puede influir en las respuestas, por lo que hay que planearlas cuidadosamente.

"Las preguntas suelen distinguirse en dos categorías: abiertas y cerradas. Las preguntas abiertas permiten que los encuestados respondan con su propia terminología. Generalmente estas son más reveladoras, ya que los interrogados no están limitados en sus respuestas. Son especialmente útiles en la etapa exploratoria de la investigación, cuando el analista busca penetrar en el pensamiento del encuestado.". "Por su parte, las preguntas cerradas predeterminan todas las posibles respuestas y el interrogado elige entre las opciones presentadas." (Dávila, 2001)

Esta constituye una de las técnicas de elicitación más usada, pues se establece una comunicación directa entre las personas interesadas y el equipo de desarrollo.

1.5.2 Sistemas Existentes

Dávila plantea que "esta técnica consiste en analizar distintos sistemas ya desarrollados que estén relacionados con el sistema a ser construido". Entre algunas de las ventajas de esta técnica se destacan:

- Poder analizar las interfaces de usuario, observando el tipo de información que se maneja y como es manejada. Por este medio se puede descubrir información importante que el usuario pudo haber omitido.
- Se analizan las distintas salidas que los sistemas producen y de esta manera se determinan posibles condiciones que el sistema debe cumplir.
- Estos sistemas ya están en producción, ya han pasado por la curva de aprendizaje del dominio del problema. Entonces, cuando se analizan, se tiene que tratar de pensar, por ejemplo, por qué manejan cierta información y para qué sirve, lo que resultará de suma utilidad para el cliente.
- Aunque esto requiere de cierto grado de trabajo (investigación y análisis). Se puede realizar a priori sin que intervenga el cliente/usuario para ello, existen en internet cantidad de demos de productos que pueden resultar similares y, también, se puede

establecer contactos con profesionales que desarrollan sistemas de características comparables.

También es recomendable que luego de haber analizado el sistema, se le muestre al cliente/usuario, ya que por su experiencia puede sugerir importantes ideas nuevas. (Dávila, 2001)

1.5.3 **Comparación de terminología**

Escalona y Koch en su estudio comparativo señalan que "uno de los problemas que surge durante la elicitación de requisitos es que usuarios y expertos no llegan a entenderse debido a problemas de terminología. Esta técnica es utilizada en forma complementaria a otras técnicas para obtener consenso respecto de la terminología a ser usada en el proyecto de desarrollo. Para ello es necesario identificar el uso de términos diferentes para los mismos conceptos (correspondencia), misma terminología para diferentes conceptos (conflictos) o cuando no hay concordancia exacta ni en el vocabulario ni en los conceptos (contraste)". (Escalona, y otros, 2002)

1.5.4 Grabaciones de video y audio

Dávila se refiere a que existen dos formas de utilizar las grabaciones, la primera como registro y apoyo de las entrevistas, y la otra para analizar algún proceso en particular.
De acuerdo a su función de apoyo, se puede centrar la atención en la entrevista en sí, en vez de tener en cuenta las notas de todo lo que se esta diciendo. Grabándose la conversación, basta con tomar algunas notas de los temas tratados para después tener una guía básica de los mismos, y saber en qué lugar de la grabación buscar. El analista puede analizar los temas con más detenimiento y con una visión más global.
Señala que "cuando se trata de analizar algún proceso en particular, su ayuda es inestimable (sobre todo las filmaciones de video) ya que permite ver y analizar en detalle ese proceso la cantidad de veces que sea necesario." Además que "filmando el lugar de trabajo se está

capturando el proceso de trabajo, lo que evita que impongamos nuestras expectativas y preferencias." (Dávila, 2001)

Siempre es conveniente comenzar las grabaciones con preguntas poco importantes dándole comodidad al entrevistado, este podría ponerse nervioso durante los primeros minutos de grabación. Por último, los entrevistados necesitan de tiempo para sentirse relajados a la hora de la grabación o la filmación.

1.5.5 Tormenta de ideas (Brainstorming)

Arango determina que esta es una técnica que se usa para generar ideas, pues estimula el pensamiento creativo. La intención en su aplicación es la de generar la máxima cantidad posible de requisitos para el sistema. No hay que detenerse en pensar si la idea es o no del todo utilizable. La intención de este ejercicio es generar, en una primera instancia, muchas ideas. Luego, se irán eliminando en base a distintos criterios como, por ejemplo, "caro", "impracticable", "imposible", etc. Las reglas básicas a seguir son, inicialmente, los participantes deben pertenecer a distintas disciplinas y, preferentemente, deben tener mucha experiencia. Esto trae aparejado la obtención de una cantidad mayor de ideas creativas. También conviene suspender el juicio crítico y se debe permitir la evolución de cada una de las ideas, y por más locas o salvajes que parezcan algunas ideas, no se las debe descartar, porque luego de maduradas probablemente se tornen en un requerimiento sumamente útil. Esto conlleva a que a veces ocurra que una idea resulta en otra idea, y otras veces se pueden relacionar varias ideas para generar una nueva. (Arango, 2002)

También son usadas las técnicas para describir los requisitos como tal, técnicas que especifican el estado de cada requisito. Las más reconocidas son la técnica del lenguaje natural y la del lenguaje formal. Escalona y Koch especifican que el lenguaje natural: es una técnica muy ambigua para la definición de los requisitos. Se definen los requisitos en lenguaje natural, sin el uso de alguna regla para ello. Y, aunque se critique su uso, a nivel práctico se llega a un mejor entendimiento con el cliente por lo que se sigue utilizando. Sin embargo el lenguaje formal: es el extremo opuesto del lenguaje natural. Se utiliza lenguaje o lenguajes formales para describir los requisitos de un sistema, para la descripción formal se usa hace años las especificaciones algebraicas, pese a que son difícil de entender para el cliente, pues

estas no favorecen a la comunicación entre el cliente y el analista. Aunque, es la representación menos ambigua y la que facilita a técnicas de verificación automatizadas. (Escalona, y otros, 2002)

1.6 Herramientas de modelado

Las herramientas que se emplean en el desarrollo de software son conocidas como herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador), estas constituyen un conjunto de aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero. Para el desarrollo de RNF se describe a continuación:

1.6.1 NFR Framework de Chung

El framework de NFR creado por Lawrence Chung (Chung, y otros, 1999) es uno de los trabajos más completos en lo referente a requisitos no funcionales (RNF) y en el propone una estructura que usa los RNF para dirigir el proceso de diseño. Uno de los conceptos principales son los “softgoal” que representa un objetivo que no tiene definidas claramente los criterios para definir si es satisfecho o no (estos suelen ser subjetivos). En el modelo recogido por Díez González en su trabajo sobre RNF, ofrece una estructura para representar y guardar los procesos de diseño y razonamiento en gráficos, llamados Softgoal Interdependency Graph (SIG). Estos gráficos recogen las consideraciones del desarrollador sobre los softgoals, y muestran la interdependencia entre estos. Para ello muestra los softgoals como nubes representando los requisitos principales en la parte superior del gráfico, mostrando con líneas los enlaces de interdependencia, y utilizando etiquetas para representar el grado en el que el softgoal es conseguido.

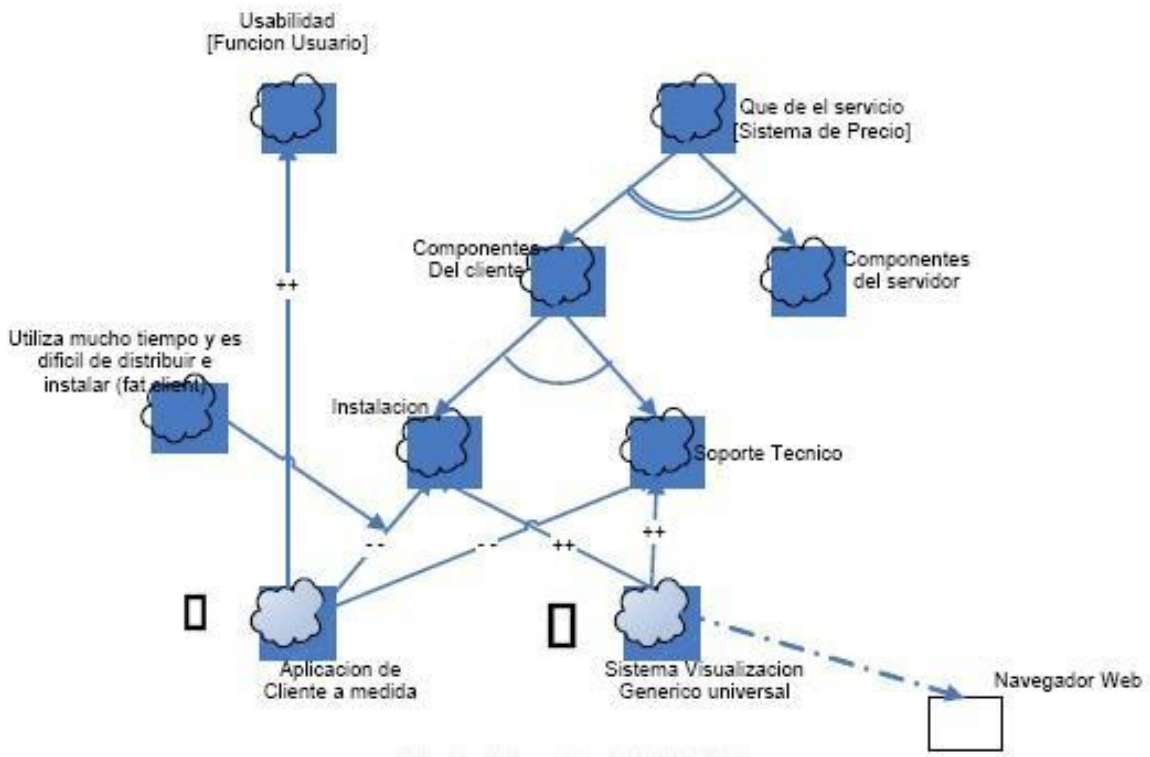

Figura 1: Ejemplo de SIG (Chung, et al., 1999)

El modelo ofrece además la catalogación de conocimiento sobre los RNF y diseño, incluyendo conocimiento sobre técnicas de desarrollo.

Los catálogos de conocimiento plantea Díez González pueden ser utilizados por los desarrolladores como una base de conocimiento de previos proyectos organizados en tres tipos de catálogos:

- De conocimiento sobre determinados tipos de RNF, junto con sus conceptos y terminología.
- Usado para organizar las técnicas de desarrollo que ayudan a cumplir los requisitos.
- Mostrando las interdependencias implícitas entre los distintos softgoals.

El conocimiento recogido en estos catálogos puede venir de distintas fuentes; libros, experiencia propia, o especialistas en la industria para conocimiento específico en un determinado dominio. El catálogo sobre los determinados tipos de RNF se usará para ir recogiendo estos requisitos en una jerarquía y se proseguirá a la construcción de un SIG en el que se plasman los principales RNF que el sistema a desarrollar debe cumplir. Estos

requisitos son softgoals, que deberán ser priorizados, clarificados, elaborados, y descompuestos en otros softgoals en caso de que fuera necesario. “Al final de todo este proceso se obtienen soluciones que pueden ser aplicadas al sistema a desarrollar, presentando posibles alternativas, y escogiendo las que más se adecuen al sistema a desarrollar”. (Díez González, 2006)

1.7 Lenguajes de modelado

El lenguaje de modelado es un conjunto estandarizado de símbolos y de modos de disponerlos para modelar un diseño de software. Se hace necesario el uso de lenguajes que modelen de manera precisa las necesidades del usuario para que el producto final tenga aceptación por los mismos. Modelar RNF no es tarea fácil debido a la abstracción que en ocasiones estos presentan.

1.7.1 Léxico Extendido del Lenguaje

Para evitar requisitos que no sean detectados o entendidos el uso de escenarios es una propuesta bastante potente. En ellos el uso de glosarios constituye una parte importante en el proceso de construcción de escenarios, este glosario se va refinando en sucesivas etapas del desarrollo y se lo utiliza para describir características del sistema.
Kaplan, Hadad, Doorn y Leite (Kaplan, et al., 2001) proponen para modelar RNF el uso del Léxico Extendido del Lenguaje (LEL), el cual es un glosario con roles y una estructura más amplia que lo habitual. Está escrito en lenguaje natural y compuesto por símbolos que son palabras o frases repetidas o relevantes del universo de discurso (UdeD). El objetivo del LEL es registrar el vocabulario de un UdeD dado por el problema, es registrar signos, palabras o frases, desde lo peculiar hasta campos específicos de una aplicación.

El LEL es un meta-modelo (entiéndase meta-modelo como un modelo que contiene varios modelos) designado a ayudar a la elicitación y representación del lenguaje usado en el macrosistema. Está centrado en una descripción de los términos del lenguaje para mejorar la comprensión del UdeD. “Para generar el LEL se registran símbolos (palabras o frases) peculiares o relevantes del dominio. Cada entrada del léxico se identifica con un nombre (o más de uno en caso de sinónimos). Tiene dos tipos de descripciones, una llamada noción que

describe la denotación del símbolo y el otro impacto que describe la connotación del mismo. Las entradas se clasifican en cuatro tipos de acuerdo a su uso general en el UdeD. Estos tipos son: Sujeto, Objeto, Verbo y Estado." (Kaplan, et al., 2001)

Señalan que para obtener el LEL de un UdeD se realizan actividades que pueden ocurrir de manera simultánea: se identifican las fuentes de información, se identifican, clasifican y describen símbolos, finalmente se verifica y valida el LEL. Concluyen demostrando que el LEL como todo lenguaje de modelado es vulnerable a defectos, durante la identificación de símbolos candidatos, pueden no aparecer claros en su primera observación, por lo que se debe prestar especial atención a esta actividad. (Kaplan, et al., 2001)

Por su parte, Sampaio do Prado Leite y Cysneiros (Sampaio do Prado Leite, et al., 2001) abordan que aunque el LEL puede manejar aspectos no funcionales del dominio a pesar del nivel de abstracción que los RNF pueden tener, la primera generación del lenguaje estaba estrechamente relacionada con símbolos que responden a RF. Esto no significa que el ingeniero de software no pueda registrar información referente a un RNF. Un conjunto bien definido de símbolos representando el vocabulario a usar en el UdeD es la clave del éxito en la estrategia de captura. La construcción de la vista no funcional se extrae del uso de un LEL existente. Por lo que se amplia el LEL para ayudar en este proceso de la elicitación de los RNF. (Sampaio do Prado Leite, et al., 2001)

1.7.2 NFR con UML

Díez González (Díez González, 2006) puntualiza que a pesar de que el modelo del Framework de Chung es uno de los trabajos más completos en lo referente a RNF, no se trata en detalle la integración de estos RNF con los RF. Por lo que existen varias propuestas para lograr dicha integración. Una de ellas es la que se realiza utilizando el lenguaje UML y sus respectivas representaciones.

Señala que mediante los puntos de convergencia que se establecen en los ciclos independientes del proceso de desarrollo, se añaden a la vista funcional las acciones y datos necesarios para satisfacer la vista no funcional. Para ello se utilizarán diagramas de clase de clases UML que expresarán los RNF.

Cysneiros (Cysneiros, et al., 2001) hace alusión a que este modelo utiliza un Léxico Extendido del Lenguaje (LEL) en el cual se registra todo el vocabulario del dominio del sistema. Y recogerá también información relacionada con RF y RNF, así como sus interdependencias. No obstante, para la representación de los RNF se usa la misma notación y los mismos diagramas SIG que Chung (Chung, et al., 1999) propone, representado los softgoals, estos se priorizan, se descomponen y al final se "operazionalizan", o sea, que se obtienen RF para satisfacer los RNF. Estas "operazionalizaciones" pueden ser dinámicas (que se refieren a conceptos abstractos y requieren más análisis) y las estáticas (que implican la necesidad de utilizar algún dato en el diseño para guardar información que es necesaria para satisfacer el RNF) (Cysneiros, et al., 2001) En la figura 2 se aprecia un ejemplo de un gráfico de RNF en el que se detalla la seguridad en un sistema de luces de una habitación.

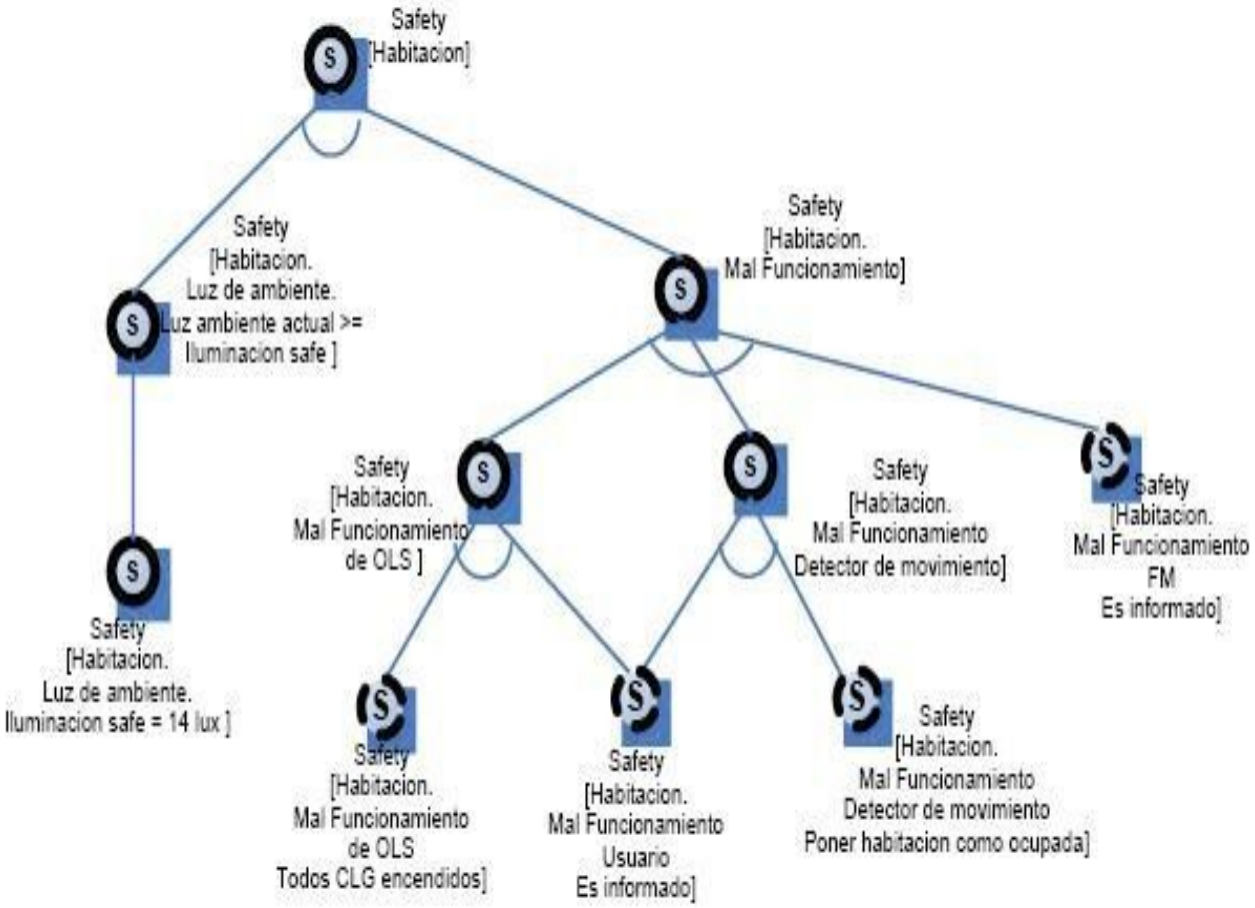

Figura 2: Ejemplo de un gráfico de RNF. (Cysneiros, et al., 2001)

Refiriéndose a la integración de requisitos antes mencionada propuesta por el modelo de UML con RNF González (Díez González, 2006) especifica que "para hacer la integración se utilizará el LEL como punto común entre los gráficos de RNFs y los diagramas de clases. Para ello, toda raíz del gráfico de RNF debe referir a un símbolo del LEL y toda clase del diagrama de clases tiene que ser nombrada utilizando un símbolo del LEL. De esta forma no solo se facilita la integración sino que también se validan ambos modelos. El modelo de integración se centra en la búsqueda de un símbolo que aparezca en los dos modelos y en evaluar el impacto de añadir las operalizaciones de RNF al diagrama de clases." (Díez González, 2006) Explica además que el proceso sería la selección de una clase del diagrama de clases, y luego se realiza la búsqueda por la ocurrencia de esta clase en los diagramas de RNF, en cada gráfico donde se encuentre el nombre de la clase buscada, se identifican las operalizaciones dinámicas y estáticas. Se debe comprobar si las operaciones y atributos que pertenecen a la clase cumplen con las necesidades expresadas en el gráfico de RNF. Si no lo cumplen, se añaden atributos y operaciones que lo hagan. En algunos casos, se llega a la introducción de nuevas clases para satisfacer los RNFs. Una vez incorporadas las nuevas clases con sus operaciones y atributos, se procede según la metodología UML para el resto del proceso. (Díez González, 2006)

1.8 Estrategias y modelos existentes.

Los modelos conceptuales que representan RNF permiten mostrar la situación real de manera más eficiente. Uno de estos modelos es el que se lleva a cabo en la estrategia de integración que aporta como resultado un framework para integrar RNF dentro de modelos conceptuales propuesto por Luiz Marcio Cysneiros, Julio Cesar Sampaio do Prado Leite y Jaime de Melo Sabat Neto (Sabat Neto, et al., 2001) con la fuerte influencia de Chung (Chung, et al., 1999), adaptando levemente los gráficos de su framework, ya que son utilizados en los inicios del desarrollo del software. Aunque, el trabajo de Chung apunta a representar RNFs y sus conflictos, conflictos detectados principalmente con los requisitos funcionales; la integración sistemática de NFRs en modelos conceptuales puede ser provechosa en identificar tales conflictos. Dicha estrategia de integración puede clasificarse como fácil de usar, simple y no requiere de mucha investigación para ponerla en práctica. Se intenta mostrar que tratando

debidamente los RNFs desde inicios del proceso de desarrollo de software e integrando estos conocimientos con el modelo conceptual se obtengan menos costos y mayor satisfacción en el cliente. Inicialmente se trata el modelo conceptual entidad-relación como el modelo que será usado, y luego el framework se extiende para ser usado con modelos orientados a objetos.

La estrategia propuesta no intenta cubrir ningún RNFs en específico. Es un acercamiento más amplio de la identificación y representación del proceso orientado a los RNF. La idea de integrar el modelo conceptual surge debido al problema de representación de los RNFs para así entender el impacto del diseño del modelo conceptual, esta representación también permite tomar decisiones en el diseño más sistemáticas. La estrategia propone usar el Léxico Extendido del Lenguaje (LEL) como pieza clave para la integración de los RNFs con el modelo entidad-relación y el modelo orientado a objetos. Una vez que todos los RNFs son representados en el modelo conceptual, se realiza un proceso de verificación. Esta verificación impone revisar que todo lo representado en el gráfico de RNF esté en el modelo conceptual, por lo que cada RNF que exista en uno estará en el otro. Otro punto importante es ver si el modelo contiene todos los atributos necesarios para satisfacer RNFs existentes. Satisfacer un RNF conlleva a cambios o inclusiones de atributos en las entidades existentes o clases o incluso la creación de nuevas entidades, relaciones o clases. (Sabat Neto, et al., 2001). En el Anexo 6 se presenta una descripción gráfica de la estrategia mencionada.

1.9 Conclusiones del capítulo.

Mediante la realización del presente capítulo se puede concluir, que el desarrollo de los RNF de software, en de la ingeriría de requisitos, determina aspectos importantes dentro del proceso de desarrollo del software. Además, teniendo en cuenta que las técnicas tratadas en el capítulo de acuerdo a lo buscado y la bibliografía consultada; el lenguaje que se usa para el modelado de los RNF y la estrategia de integración de los RNF con modelos conceptuales no resuelve en ocasiones de manera satisfactoria y esperada por los clientes el desarrollo de los RNF. Por lo que se propone una estrategia que potencie el proceso de desarrollo de estos requisitos, que consolide pasos en busca de la calidad de los mismos, que involucre buenas prácticas en torno a la gestión del conocimiento y que la misma sea pueda ser consultada en proyectos de la universidad.

CAPÍTULO 2: DIAGNÓSTICO DEL ESTADO ACTUAL DEL PROBLEMA Y SOLUCIÓN PROPUESTA.

2.1 Introducción.

En este capítulo se presentan aspectos relacionados al estudio de la situación actual del proceso de desarrollo (elicitación, análisis, especificación, y validación) de los requisitos no funcionales en proyectos productivos de la UCI, y se especifica y argumenta la propuesta de una estrategia que contribuya al éxito de estos procesos.

2.2 Situación actual de la problemática analizada.

Se hizo un estudio de la situación actual de cómo se realiza este proceso de desarrollo de los requisitos no funcionales de software en los proyectos productivos de la UCI, y se diagnosticaron a través de los instrumentos utilizados para ello, aspectos relevantes para el resultado de la investigación. Estos instrumentos fueron seleccionados de las propuestas que brinda la metodología de la investigación científica y fueron las entrevistas y encuestas.

2.3 Población y muestra de la investigación.

La población que se tomó para el estudio de la situación actual de cómo se realiza el desarrollo de los requisitos no funcionales de software en los proyectos productivos de la UCI fue conformada con integrantes de proyectos productivos de cada una de las diez facultades, miembros de la infraestructura productiva y especialistas de la Dirección de Calidad de Software de la universidad.

En la población mencionada se tomó un muestreo intencional siendo la muestra tres analistas principales en cada facultad, teniendo en cuenta que por su desempeño dentro del proyecto y el rol asumido en el proceso de desarrollo son personas que pueden con gran probabilidad aportar información muy valiosa para la investigación.

2.4 Propósito y resultados de los instrumentos utilizados.

Se realizó una encuesta y una entrevista (Anexos 7 y 8 respectivamente), donde se realizan algunas preguntas con el propósito de diagnosticar aspectos relevantes para nuestra investigación.

Se efectuó una entrevista a la Directora de Calidad de Software de la universidad, Aylin Febles, que dio paso a la identificación de aspectos a considerar para la confección de la encuesta que se aplicó. (Ver anexo 7)

Estos aspectos se identifican dentro de las cuatro etapas del desarrollo de requisitos, principalmente los no funcionales para los cuales está dirigida esta investigación.

Como se puedo ver en el capítulo 1 esas cuatro etapas serían: elicitación, análisis, especificación, y validación de los requisitos no funcionales. A continuación se muestran las cuestiones que fueron propósito de diagnosticar y los resultados en las mismas.

Análisis de los resultados arrojados por la encuesta.

Mediante las técnicas utilizadas para la recopilación de información se pudo arribar a un grupo de conclusiones y reflexiones importantes en torno al proceso de desarrollo de los RNF. Estas arrojaron los siguientes resultados que se exponen a continuación.

a) **Sobre las actividades que se considera deben realizarse con alto grado de importancia durante la fase de inicio del proceso de desarrollo de software, concretamente en flujos de trabajo de requisitos.**

Para identificar estas actividades se partió de una propuesta enfocada generalmente a la metodología RUP, una de las más usadas en los proyectos de la universidad, pero teniendo también en cuenta la posibilidad de enfocar la respuesta al uso de cualquier otra metodología que se usase en determinado proyecto.

Los resultados de esta pregunta fueron los siguientes que se visualizan a continuación auxiliados de un gráfico que enuncia las actividades más ratificadas con alto grado de importancia.

Se habla de la actividad de determinar las necesidades (requisitos) del cliente, donde veintiocho de los encuestados seleccionan esta actividad siendo así una estadística alta que

arroja al 93, 3 % de la muestra. Desarrollar el plan de gestión también se encuentra altamente valorado con un número de veintitrés encuestados que optan por su selección, así como encontrar actores y casos de uso siendo en este caso veintidós los que la seleccionan. Veinte seleccionan la captura de un vocabulario común, diecinueve estructurar el modelo de caso de uso. Dieciséis marcaron desarrollar el documento visión y ocho identifican administrar las dependencias.

Además de estos datos se recogieron unos cuantos criterios que también tributan al diagnóstico que se quiere recoger con esta pregunta, donde algunos refieren añadir con alto grado de importancia la descripción textual de los casos de uso, hacer modelos conceptuales y diccionarios de datos. Modelar el Negocio (Entender perfectamente como funciona la organización y determinar cuáles son las políticas y restricciones que tienen). Evaluar la organización del negocio, la existencia de documentación referida a los procesos, si se necesita o no reingeniería de procesos.

En cuanto a requisitos es abarcada la actividad de especificarlos. Validar cada uno de ellos de acuerdo a criterios de clientes, especialistas y analistas que incluso no hayan participado en la captura. Aplicación de métricas que permitan medir consistencia, completitud, y complejidad. Hacer una maqueta o prototipo no funcional que soporte los CU críticos para la arquitectura. (Se le presenta al cliente).

En algunos casos se hablaba del hecho de la comunicación con el cliente lo que constituye un aspecto de gran importancia para lograr identificar sus expectativas, en el marco de lo que constituyen los requisitos del sistema que se desea construir.

Con ese propósito se hablaba de cómo sería interesante por parte de los clientes la presencia de un informático en el equipo de ellos para facilitar la comprensión entre ambos (cliente/usuarios con los desarrolladores), aspecto que en caso de suceder sería muy útil.

La solución está en buscar alternativas de comunicación con el cliente, que logre en el marco normal del desarrollo del negocio la eficiencia de la comunicación de lo que solicita, idea que también se ve fortalecida en el criterio de otros analistas de realizar los prototipos iniciales de la interfaz de usuario.

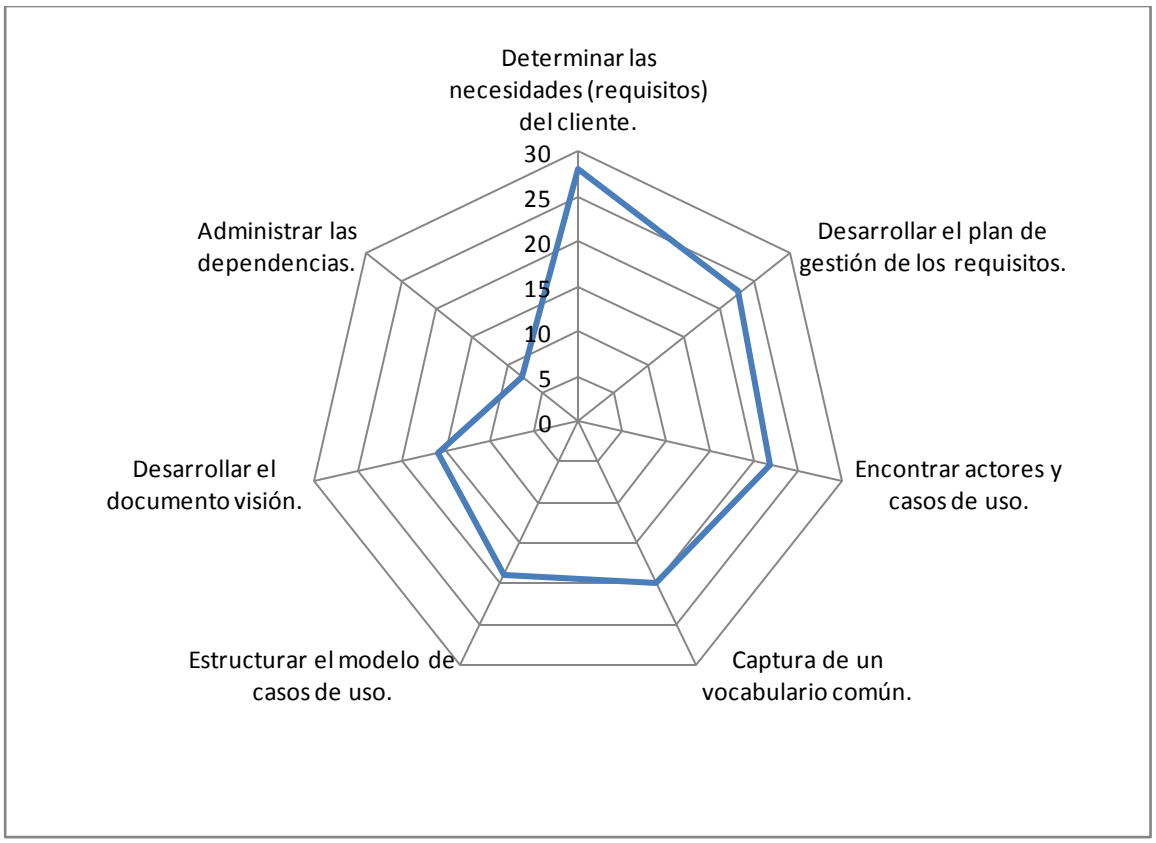

Figura 3: Gráfico que visualiza el resultado de la pregunta de la encuesta con propósito de identificar actividades más consideradas con alto grado de importancia en el proceso de desarrollo. (Fase de inicial del proceso de desarrollo)

Por lo que se puede identificar que la actividad de determinar las necesidades (requisitos) de los clientes es de alto grado de importancia determinado por un significativo número de criterios dentro de la muestra estudiada. Seguida de otro grupo de actividades que se relacionan a la misma. Teniendo en cuenta el porciento es evidente que se considera así en todas las facultades.

b) Sobre si se utiliza un modelo para capturar los requisitos del cliente.

En cuanto a si se utiliza un modelo para la captura de los requisitos del cliente se identifican mayormente los que propone RUP, determinándose en sentido prioritario a los requisitos funcionales.

"RUP dirige la descomposición del problema hacia la funcionalidad (casos de uso) que debe ser entregada como producto de software. Por la orientación de los casos de uso, los analistas limitan su especificación a la descripción de requisitos funcionales. Comúnmente atributos de calidad, (conocidos también como requisitos no funcionales) y reglas del negocio que han sido elicitados y que podrían ser transversales a muchas funcionalidades, son tratados por los arquitectos en fases avanzadas del ciclo de vida." (Tabares, y otros, 2007)

Muchos de estos proyectos identifican modelos adaptados al trabajo en el proyecto, a estrategias confeccionadas a través de la experiencia de equipos de desarrollo, de trabajos presentados en eventos y tesis confeccionadas en el marco del desarrollo del proyecto por roles de sus integrantes. Pero no se identifica un modelo asumido para el caso de los requisitos no funcionales que se adopte en las estrategias para el desarrollo de forma unificada o al menos mayormente empleados por los mismos.

c) Sobre si se tiene identificado los pasos a seguir para realizar la captura de los requisitos de software.

Se evidencia en los resultados como la tendencia es adecuarlo al proyecto, identificándose trece casos que especifican que depende del proyecto. Dependiendo de la metodología empleada estos pasos se reajustan a las estrategias concebidas por el equipo de desarrollo, las facultades asumen estos pasos teniendo en cuenta las buenas prácticas y experiencias en este sentido. Se identifican seis casos que no tienen identificado los pasos para esta actividad.

A modo de resumen generalmente sí se tienen identificados estos pasos, aunque en la mayoría dependen del proyecto que se va asumir.

Es recomendable se trabaje en la identificación de pasos comúnmente a enfrentar en este proceso, pues es importante que los analistas tengan definidos un grupo de pasos que aunque después sean moldeados a razón de las necesidades del proyecto, contribuyan a la estrategia a utilizar principalmente en caso de los RNF.

d) Sobre los aspectos que piensan pueden influir en la calidad de la captura de los requisitos no funcionales.

Sobre esta pregunta se pusieron a consideración de los encuestados un grupo de posibles aspectos que pudiesen ser los candidatos a identificarse y la posibilidad de sumar otros como respuesta a la cuestión planteada. Siendo así el resultado de la misma el que se presenta a continuación. Veintiséis de los encuestados seleccionan el adecuado lenguaje de comunicación con el usuario y esa misma cantidad considera el aspecto de conocer buenas prácticas u experiencias vividas durante esta actividad en otros proyectos productivos de la UCI. Un número de veintidós personas seleccionan que influye la firma con el cliente de los requisitos no funcionales que se identifican y ese mismo número de selecciones corresponde a no clasificar estos requisitos o clasificarlos incorrectamente. Quince piensan que influye el grado de importancia con que se valore establecer previamente durante la entrevista con el cliente estos requisitos, es decir, considerar la necesidad de establecerlos con el cliente durante ese tiempo. El entorno de desarrollo de la entrevista es significativo para nueve de los encuestados así como siete piensan en que repercute el hecho de no adoptar un modelo para esta actividad. El tiempo de duración de la entrevista, lo consideran que influye solo cinco de los treinta.
Además de estos aspectos reflejados anteriormente se añadieron otros que los analistas encuestados determinaron son aspectos que pueden influir en la calidad de la captura de los requisitos no funcionales.

Dentro de ellas se encuentran la falta de conocimiento, de buenas prácticas en el mundo de la captura de requisitos no funcionales. El mal modelamiento del negocio producido también por una mala captura de requisitos y mal modelamiento del sistema. Además la habilidad para identificar RNF a partir del lenguaje natural de expresión de los clientes y la utilización de varias técnicas de obtención de requisitos (no solo entrevista).

Se identifica que los RNF generalmente están fuertemente asociados a decisiones técnicas, sería viable que los analistas estén bien preparados en este sentido. También añaden que deberían participar en su identificación y negociación el arquitecto y los líderes técnicos del equipo. Sobre los RNF recomiendan escribirlos en algún lugar(o en un documento común en

la descripción textual del CU). Y consideran también que no manejar bien la disponibilidad de tiempo del cliente podría incluso ser un factor que atrasaría el proceso.

Figura 4: Gráfica que visualiza el estado de opinión de los encuestados respecto a los aspectos que piensan influye en la calidad de la captura de los RNF.

Por lo que se puede concluir se prioriza el hecho de lograr establecer una buena comunicación con el cliente, dada la significación que esto tiene para lograr identificar correctamente los RNF, en lo que de alguna forma se debe tener en cuenta los aspectos del tiempo de las entrevistas con el cliente, el entorno donde se desarrolla, y la firma de estos requisitos. Así como se pudo analizar la necesidad de un modelo, que también pudiera contribuir al adecuado proceso de clasificación de los mismos.

Es muy satisfactorio el hecho de que se identifique la importancia de establecer estos requisitos durante la entrevista con el cliente así como conocer buenas prácticas u experiencias que tributen a consolidar estrategias cada vez más eficientes.

e) Sobre si se piensa que pueda ser obviado el desarrollo de los RNF durante la fase de inicio en dependencia del tamaño o complejidad del proyecto.

Veinticinco de los encuestados piensan que no puede ser obviado el desarrollo de los RNF durante la fase de inicio, argumentando que:

- Son el complemento de los RF para lograr la calidad del producto final.
- Es importante hacer esta actividad durante todo el desarrollo del proyecto de software, para hacer un producto con la calidad y la eficiencia requerida.
- Los RNF están ligados estrechamente a los RF y a la implementación de los mismos.
- La captura de los RNF no depende de la complejidad del proyecto.
- Muchos de estos RNF pueden definir cambios en una arquitectura del software que se diseña.
- Estos deben estar reflejados en el documento visión y deben ser tomados en cuenta para la selección de herramientas a emplear durante el desarrollo, influyen en los RF, así como en la distribución o concepción de la arquitectura del software.
- Independientemente del tamaño o complejidad son claves para tomar decisiones respecto a la arquitectura.
- Durante la fase de inicio es donde se establece una visión del proyecto y se define el alcance del mismo y en todo esto influyen los RNF.
- De estos RNF depende mucho la calidad del producto.
- De ellos depende en gran medida la arquitectura.
- El desarrollo de los RNF constituye en parte, la base para la posterior arquitectura a formar.
- Al terminar la fase de inicio deben estar estables los requisitos, conocido como línea base de requisitos. Los RNF influyen con peso en la arquitectura de la aplicación, cuando se incluye elaboración ya es hora de comenzar a implementar la misma.
- Desde el inicio te da una medida de las características del sistema.
- Independientemente del tamaño del proyecto se deben conocer cuáles, sobre qué desarrollar, sobre qué debe correr una vez implementado, etc.
- No importa si un software es grande o pequeño, siempre tiene RNF.

- Es importante tener claros los RNF independientemente del tamaño o complejidad del proyecto.
- Define funcionalidades y asegura las bases de la programación.
- Estos requisitos son los que va a dictar las propiedades que debe cumplir el sistema. Si no se hace al inicio se puede caer en problemas que son más costosos en etapas futuras.
- No tienen que ver nada la complejidad y tamaño del proyecto sino que ayude a que la calidad del producto de software. Afirmando que es fundamental hacerla en la fase de inicio.
- El correcto funcionamiento del software depende también el entorno donde se despliegue.
- Siempre ayuda a establecer el comportamiento del sistema.
- En la fase de inicio, considero que es fundamental tener una visión general del proyecto. Por lo que una cosa tan importante como la obtención de los RNF no debe dejar de hacerse.

Sólo una de las personas encuestadas piensa que puede ser obviado el desarrollo de los requisitos no funcionales durante la fase de inicio en dependencia del tamaño o complejidad del proyecto, explicando que "los requisitos funcionales son los fundamentales en ese momento aunque los RNF de software hay que definirlos", lo que da una medida que aun así se considera la necesidad de hacer énfasis en la captura de los RNF.

f) Sobre cómo se valora la importancia de la actividad de especificar los requisitos de software y cómo se valora para el caso de los RNF.

Mediante esta pregunta se pudo llegar a analizar el alto grado de importancia con que se valora la actividad de especificar los requisitos de software, y dentro de estos también se identifican con alto grado de importancia para el caso de los RNF.

Veinticinco de los encuestados consideran alto grado de importancia para especificar los requisitos del software en general, solo dos de ellos le atribuyen una importancia media. Para el caso de los requisitos no funcionales (RNF), veintiuno consideran una importancia alta, seis

le consideran de forma media la importancia y solo dos de importancia baja. En ninguno de los dos casos la importancia es identificada nula.

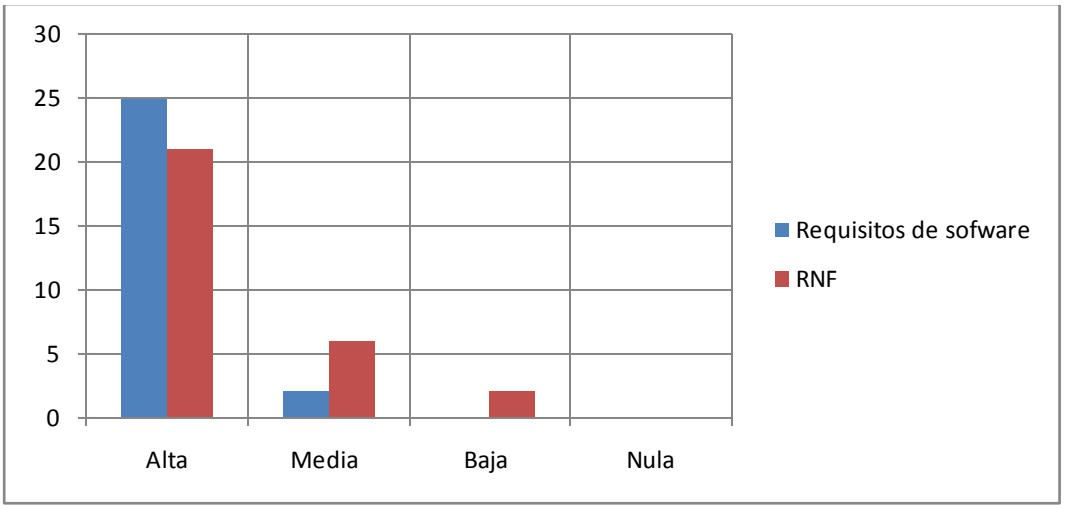

Figura 5: Gráfico para visualizar resultado de la encuesta como se valora la importancia de la actividad de especificar los requisitos de software y como se valora para el caso de los RNF.

g) Sobre los tipos de RNF más comúnmente utilizados en los proyectos productivos.

Cantidad de personas	Requisitos identificados en las encuestas.
26	Requisitos de Rendimiento
21	Requisitos de Fiabilidad
23	Requisitos de Disponibilidad
29	Requisitos de Seguridad
22	Requisitos de Portabilidad
10	Requisitos de Mantenibilidad
7	Requisitos de Escalabilidad
14	Requisitos de Reusabilidad
27	Requisitos de Interfaz Externa
21	Requisitos de Amigabilidad (Usabilidad)
8	Requisitos de Capacidad (Consumo de recursos)

7	Requisitos Culturales y Políticos (Robertson & Robertson)
16	Requisitos Legales (Robertson & Robertson)
1	Requisitos de Desempeño (Brito, Moreira y Araújo)
2	Requisitos Operacionales (Brito, Moreira y Araújo)
2	Requisitos Económicos (Brito, Moreira y Araújo)
7	Requisitos de Características en Tiempo de Ejecución (Malan y Bredemeyer)
1	Requisitos de Tiempo de Mercadeo (Viega y McGraw)
0	Requisitos de Simplicidad (Viega y McGraw)
4	Requisitos de Precisión (Consistencia Interna – Chung y Nixon)
0	Requisitos de Precisión (Consistencia Externa – Chung y Nixon)

Tabla 1: Relación de cantidad de personas que seleccionaron la clasificación de RNF puesta en la encuesta.

Se identificaron además:

- Requisito de software, de hardware.
- Adicionales (Cuando es difícil ponerle una categoría porque es una característica necesaria para cumplir por ejemplo con una regla de negocio)
- Interfaces de usuario.

h) Sobre qué métodos se utilizan para identificar RNF.

Se identificaron como los métodos más usados las entrevistas y cuestionarios, los sistemas existentes y las tormentas de ideas. En el orden de las veintiocho, dieciocho y trece opiniones respectivamente. Luego les siguen las grabaciones de audio y video con seis que la identifican y cuatro en el caso de la comparación de terminología.
También se recogieron otro grupo de métodos utilizados como son:

- Revisión de documentos.
- Estándares, buenas prácticas y referencias sobre arquitectura de información. (Internet y especialistas)
- Documentación existente sobre las tecnologías reuniones de intercambio sobre estas tecnologías.

i) Sobre el lenguaje que consideran más apropiado y eficiente para la especificación de los RNF.

Veintitrés seleccionan el lenguaje natural. Cuatro seleccionan el lenguaje de modelado siendo de los mismos especificados tres como UML. Cinco hablan de un lenguaje formal, sin especificar cual sería este lenguaje. Solo uno refiriere que sería con los elementos técnicos necesarios.

j) Sobre la medida de la suficiencia alcanzada durante la especificación en el proceso de desarrollo. ¿De qué vías de comprobación se apoya para esto?

Quince señalan utilizar las métricas orientadas al cliente, y trece las métricas orientadas a la arquitectura muchos de los casos señalan utilizar las dos.

Como otros de los aspectos o posibles vías de comprobación también especifican:

- Se documentan como parte de algunos artefactos del proyecto.
- Pruebas de aceptación y revisiones internas del software.

k) ¿Qué herramientas se utilizan para el modelado de los RNF?

Ninguno de los encuestados seleccionó del cuestionario: NFR Framework de Chung. Solo ocho se refieren a NFR con UML y en un caso especifican que no los representan, solamente se listan en un documento.

l) Sobre si se intercambia experiencia del trabajo realizado relacionado con esta actividad con especialistas de otros proyectos que realicen funciones similares en el mismo o estén relacionados activamente con el tema.

Solo dos señalan realizarlo con mucha frecuencia, seis señalan que si, doce muy pocas veces y nueve responden que no.
En los casos que responden afirmativamente y de algún modo realizan ese intercambio de experiencias argumentan causas como:

- “En el proyecto productivo al que pertenezco donde hay profesores y estudiantes que les interesa el tema, en la maestría de GP a la que pertenezco y en el grupo de investigación de requisitos. “
- “Con mucha frecuencia por lo general lo que hacemos es conversar acerca de cómo cada uno ha resuelto los problemas encontrados y así siempre encontramos una manera más eficiente de tratarlos.”
- “Sí, hemos realizado encuentros informales donde conversamos sobre el tema de acuerdo a las características de nuestros proyectos, lenguajes de desarrollo, solicitudes de los clientes, disposición regional, etc.”

2.5 Propuesta de estrategia para el trabajo con los requisitos no funcionales de software en proyectos productivos de la UCI.

2.5.1 Antecedentes que justifican la propuesta

La UCI, es una universidad que “tiene como misión dirigir la formación del Ingeniero en Ciencias Informáticas con conocimientos, habilidades y valores sólidos, sustentados en una concepción científica y dialéctico-materialista del mundo, que estén comprometidos con su Patria y que actúen como profesionales responsables, honestos, honrados, creativos, modestos, solidarios y con ética revolucionaria en el campo de las Ciencias Informáticas, poseedores además de una cultura general integral.

La UCI, es una universidad productiva, que tiene la misión de producir software y servicios informáticos a partir de la vinculación estudio-trabajo como modelo de formación.

La Producción de Software y Servicios Informáticos se basa en la integración de los procesos de formación, investigación y producción en torno a una temática para convertirla en una rama productiva.

Este espacio de integración temática es denominado Polo Productivo y se promueve la formación de pregrado, postgrado, la colaboración nacional e internacional, el fomento de líneas de investigación y desarrollo y la ejecución de proyectos en el marco de acuerdos de trabajo.

Esta integración garantiza la innovación continua que genera y aporta valor a los productos y servicios, promueve la gestión del conocimiento garantizando un mayor rendimiento, logra una mejor utilización y aprovechamiento de los recursos humanos y materiales, generando alta especialización y colaboración.

La producción se concentra en el desarrollo de proyectos de más de 30 Polo Productivos y se destacan resultados en las esferas de salud, educación, software libre, teleformación, sistemas legales, realidad virtual, automatización, bioinformática, procesamiento de imágenes y señales, entre otras.

Promueve el desarrollo de productos y servicios informáticos en aquellas ramas donde Cuba tiene un reconocido prestigio en el mundo a través del concurso de los mejores especialistas del país para lograr una solución con calidad y de impacto internacional." (UCI, 2008)

Para la producción se cuenta con una fuerza admirable compuesta en gran medida por estudiantes y profesores de la universidad. Estos distribuidos por los proyectos que integran asumiendo determinados roles en los mismos. Muchos de estos casos analistas involucrados en el proceso de desarrollo de software que tienen que desempeñar actividades para fases iniciales del proceso, donde por ejemplo se encuentran las relacionadas a la ingeniería de requisitos.

Para la producción de software y servicios informáticos la ingeniería de requisitos juega un papel fundamental en los proyectos productivos, destacándose el desarrollo de modelos que potencian el coherente y eficaz momento que esta abarca. Se asume de este modo alternativas para el desarrollo de los requisitos de software, desarrollo que se refiere como se explica en el capítulo uno a la obtención, análisis, especificación y validación de los mismos.

En el caso de los requisitos funcionales generalmente se sigue un proceso con aspectos consistentemente comunes. Se siguen modelos que tributan a un estándar estratégico potenciada por la metodología escogida para trabajar en el proyecto y asumir determinado proceso de desarrollo de software. Pero en el caso de los requisitos no funcionales, no sucede de la misma manera. Se siguen alternativas eficientes, pero no un modelo o estrategia que asuma un concepto común de buenas prácticas identificadas en los excelentes profesionales

y estudiantes generalmente vinculados a estos proyectos, ni se consolidan elementos del lenguaje y/o herramientas a utilizar de forma sistemática en torno a esta actividad.

Se realiza un proceso de obtención, análisis, especificación y validación sin seguir la misma estrategia, lo que lleva a considerar que no siempre se toman estándares que sean identificables en procesos equivalentes de otros analistas o proyectos.

Sobre las herramientas que pudieran identificarse en este proceso, solo se identifican pocos casos que se refieren a NFR con UML y en un caso especifican que no los representan, solamente se listan en un documento.

Muchas propuestas encuentran limitaciones en el trazado de los RNF, ya que muchas metodologías como UML no presentan elementos de modelado para este tipo de requisito.

Un buen paso para contribuir a elevar el buen funcionamiento de estos sistemas es "cerciorarse que los requisitos del sistema, y en particular los requisitos no funcionales que afectan al "safety" en el sistema se recojan y se plasmen el los requisitos del sistema y se incorporen al diseño." (Díez González, 2006)

Estos factores pueden conllevar en ocasiones a determinados problemas entre los cuales pueden encontrarse que:

- No se identifique una alternativa común que permita un flujo estandarizado de este trabajo dentro de un mismo proyecto, pudiendo ser continuado por nuevos recursos humanos que pudieran asumir un rol asociado a esto, y que fluya de diferentes formas.
- Ocurran fallas en el flujo de requisitos dentro de la fase inicial del proyecto, o en fases más avanzadas repercutiendo en la arquitectura del sistema como uno de los factores más críticos del proceso de desarrollo del software, y repercutiendo en gran medida en el costo del mismo.
- No se potencie al intercambio de experiencias entre los profesionales y estudiantes vinculados a esta labor, que de una u otra forma también se identifican dentro del proceso de gestión del conocimiento.

- Ocurran problemas que repercutan anexados a determinados riesgos no gestionados por un mal manejo de los requisitos no funcionales.
- Se invierta más tiempo del estimado en actividades planificadas en el proyecto dentro del proceso de desarrollo.
- Se cometan fallas durante la continuidad de un proyecto involucrando nuevo personal relacionado al proceso de desarrollo de los requisitos no funcionales.

Estos factores ejemplifican la importancia de una adecuada estrategia para el desarrollo de los requisitos no funcionales de software dentro de los proyectos productivos, lograr de este modo que los software y productos informáticos que se realicen se vean potenciados por esas buenas prácticas que ejercitan los profesionales y estudiantes además de que estas también potencien en gran medida la experiencia y estrategias de los recursos humanos en su admirable desempeño productivo.

2.5.2 Alcance y misión de la estrategia para el desarrollo de los requisitos no funcionales de software en los proyectos productivos de la UCI

Esta estrategia se realiza aplicable a los proyectos productivos de la universidad de las ciencias informáticas, en cualquiera de sus facultades ya sean incluso facultades regionales de la misma, siempre que la considere a utilización los analistas involucrados a fases iniciales del proceso de desarrollo.

La misión de esta estrategia es dotar a los recursos humanos vinculados a los proyectos productivos, principalmente analistas, de una alternativa eficiente que pueda ser utilizada por los proyectos de la universidad de forma tal que contribuya al proceso de elicitación, análisis, especificación y validación de los requisitos no funcionales, y también a la comunicación entre los mismos tributando a la gestión del conocimiento.

2.5.3 Visión

Los proyectos productivos cuentan con una estrategia que contribuye a su fortalecimiento en sentido al desarrollo de los requisitos, para el caso de los requisitos no funcionales, que tributa a la calidad y eficiencia del proceso de desarrollo que ejercen. Se ve materializada una idea

que apoya a los proyectos a alcanzar mayor entendimiento y satisfacción en respuesta a las solicitudes de los clientes y en este aspecto a la eficiencia que se integra finalmente en el producto de software que determinan. Permite desarrollar una comunicación en términos más comunes y menos ambiguos, enfrentándose a los riesgos que pueden traer implícitas estas etapas y de algún modo constituye un avance más en el reconocido proceso de gestión del conocimiento

2.5.4 Objetivos

Objetivos General:

Crear y potenciar acciones para la elicitación, análisis, especificación y validación de los requisitos no funcionales de software sustentada en la unificación de experiencias de buenas prácticas realizadas en este sentido y dotar estas acciones de plantillas que contribuyan a la organización de las acciones concebidas, para que de esta forma se contribuya al éxito de este proceso de desarrollo de requisitos no funcionales.

Objetivos específicos:

- Diagnosticar el estado del desarrollo de los requisitos no funcionales en proyectos productivos de la UCI.
- Definir los flujos de las técnicas, lenguajes y herramientas para las metodologías de desarrollo en proyectos productivos.
- Unificar y potenciar las acciones dirigidas al desarrollo (elicitación, análisis, especificación y validación) de los requisitos no funcionales.
- Conocer y analizar experiencia de especialistas vinculados al proceso de desarrollo de requisitos no funcionales en proyectos de la UCI.
- Contribuir a la ingeniería de requisitos en los equipos de desarrollo en proyectos de la UCI.
- Proveer al equipo de desarrollo con un conjunto de artefactos para el desarrollo de requisitos no funcionales.

2.5.5 Acciones que propone la estrategia.

La ingeniería de requisitos y por tanto el desarrollo de requisitos funcionales y no funcionales se realiza en la universidad con alto grado de importancia y seguimiento, es un factor directamente vinculado a los polos productivos y es evidenciado en los proyectos productivos de la universidad. Es un elemento que también se encuentra contenido en la integración de los procesos académicos, de producción, e investigación, que tiene como esencia la pirámide del éxito para el cumplimiento de los admirables objetivos de la universidad. (Ver figura 8)

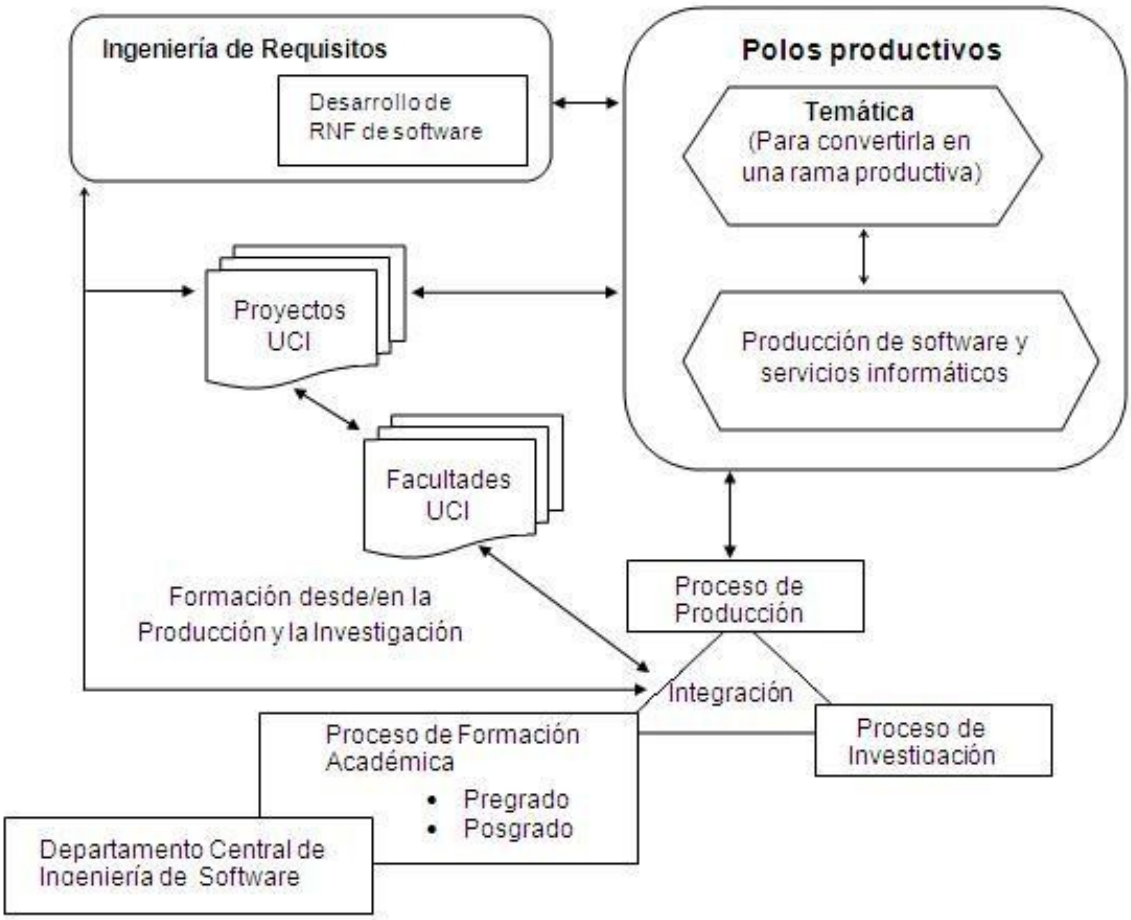

Figura 6: Esquema de integración de factores de la UCI involucrados en la estrategia.

Teniendo en cuenta estos factores, la integración y relación de los mismos se proponen las estrategias especificadas a continuación.

Se recuerda como, según CMMI los requisitos son la base para el diseño, y el desarrollo de requisitos incluye las actividades siguientes:

- Elicitación, análisis, validación y comunicación de las necesidades del cliente, de sus expectativas, y restricciones que se pueden obtener de los requisitos del cliente que constituyen el entendimiento de que lo que los stakeholders precisan.
- Colección y coordinación de las necesidades de los stakeholders.
- Desarrollo del ciclo de vida de los requisitos del producto.
- Establecimiento de los requisitos del cliente.
- Establecimiento del producto inicial y de los requisitos consistentes del producto con los requisitos del cliente.

Estas actividades se resumen en las etapas de elicitación, análisis, especificación y validación de requisitos. (CMMI Product Team, 2006) Por tanto se está trabajando sobre el marco de esas actividades.

a) Personal a involucrarse directamente en el uso de la estrategia.

La elicitación identifica y propone actividades involucradas en el descubrimiento de los requisitos del sistema. Los analistas deben trabajar junto al cliente para descubrir el problema que el sistema debe resolver, los diferentes servicios que el sistema debe prestar, las restricciones que se pueden presentar, etc. (Dávila, 2001)

Del mismo modo que lo propone el proceso unificado del desarrollo de software (RUP) entre los roles más significativos en esta etapa está: el arquitecto, el especificador de requisitos y el revisor técnico y el analista del sistema.

Esta estrategia valora muy positivamente el uso de la técnica de trabajo con las “Tormentas de ideas” (“Brainstorming” en ingles) técnica que se usa para generar ideas, con el propósito de generar la máxima cantidad posible de requisitos para el sistema (Arango, 2002). Es positivo que los participantes pertenezcan a distintas disciplinas y, preferentemente, deben tener experiencia, esto enfocado al equipo de desarrollo. De esta forma se contribuye a la obtención de una cantidad mayor de ideas creativas.

Acciones dirigidas a la elicitación de RNF.

a) Actividades dirigidas a minimizar riesgos de esta etapa:

- Revisión de los detalles técnicos que han sido aportados por el cliente.
- Determinar detalles técnicos innecesarios que pueden confundir más que clarificar los objetivos del sistema.
- Revisión del alcance del sistema.
- Identificar características significativas para el proceso referente a los clientes/usuarios.
  - Si no están completamente seguros de lo que necesitan.
  - Si tienen una pobre compresión de las capacidades y limitaciones de su entorno de computación.
  - Si no existe un total entendimiento del dominio del problema.
  - Si existen dificultades para comunicar las necesidades al ingeniero del sistema.
  - Si se identifica que ocurre la omisión de información por considerar que es "obvia".
  - Si la especificación de requisitos están en conflicto con las necesidades de otros clientes/usuarios,
  - Si los requisitos especificados son ambiguos o poco estables.

Estas actividades se consideran dado un análisis de la fuente bibliográfica consultada: (Pressman, 2002)

b) Acciones dirigidas a la comprensión.

- Realizar revisión interna sobre la comprensión del dominio de la aplicación, y hacer un bojeo de ideas referentes al área general de aplicación del sistema.
- Definición del problema en sí, extendiendo y especializando el conocimiento en las áreas críticas que identifican realmente el problema sobre el cual trabajar y elevar esa visión hasta la comprensión del negocio. ¿Cómo? uno de los modos sugeridos es hacer un esquema analítico de cómo el sistema puede contribuir a lograr las metas referidas por el cliente.

- Identificar inicialmente las principales restricciones de los usuarios del sistema, y verificar con el cliente el rol de cualquier otro sistema que actualmente se involucre en dichos procesos.
- Se propone y recomienda el trabajo con las “Tormentas de ideas” (“Brainstorming” en ingles)

Utilización de plantillas para elicitar requisitos de software.

Se proponen varias plantillas en dependencia del método utilizado para la obtención de estos requisitos con el cliente. Configuradas con el propósito de utilizarlas en las reuniones de elicitación de requisitos y en el registro de los mismos. Con el uso de estos aspectos, la estructura de la información en forma de plantilla y la propuesta de frases estándar se facilita la redacción de los requisitos,

Se tuvo en cuenta para ello trabajar sobre la tendencia de los métodos más utilizados que arrojaron como resultado las encuestas realizadas. Estas fueron las entrevistas y cuestionarios, la comparación con sistemas existentes, las tormentas de ideas, la comparación terminológica y las grabaciones de audio y video.

- Definir método a utilizar y adjuntado a este la selección de la plantilla diseñada para la utilización del mismo.

Para el caso del método entrevistas y cuestionarios

Esta constituye una de las técnicas de elicitación más utilizada y así se confirmó en los resultados de las encuestas a los analistas de los proyectos productivos de la muestra estudiada, pues se establece una comunicación directa entre las personas interesadas y el equipo de desarrollo.

Fecha	**Hora**	**Lugar**
Identificación del Entrevistado:		
Cuestionario:		
Identificador del requisito		
Nombre del requisito		
Descripción		
Clasificación	**__ Usabilidad** **__ Fiabilidad** **__ Eficiencia** **__ Soporte.** **__ Restricciones del diseño** **__ Requisito para la documentación** **__ Componentes comprados** **__ Interfaz** **__ Requisito de licencia** **__ Requisitos legales** **__ Estándares aplicables** **__ Seguridad** **__ Escalabilidad** **__ Reusabilidad** **__ Culturales y Políticos.** **__ Características en tiempo de ejecución** **__ Interoperabilidad**	
Observaciones de la clasificación		
Observaciones generales		
Comparación de terminología		
Grabaciones de video y audio		
Datos y firma del cliente	**Datos y firma del entrevistador**	

Figura 7: Plantilla para la elicitación de los RNF en los proyectos productivos.

Se propone el uso de una plantilla que posee la recogida de los datos referentes a los resultados que puede arrojar este método utilizado (Figura 7). En la cual se puede apreciar, deben siempre tenerse en cuenta la fecha, hora y lugar de la entrevista o cuestionario. Así como la identificación del entrevistado, que pudiera ser el cliente del sistema u otra persona relacionada al mismo, que el cliente determine realiza funciones que pueden aportar información valiosa para el sistema que se solicita.

Por tanto los pasos que propone el uso de esta plantilla son:

- Identificar fecha, hora y lugar de la aplicación de la misma.
- Identificación del entrevistado, con el nombre y de ser posible la responsabilidad que lo asocia a este proceso.
- Llenar el cuestionario, este paso es opcional, depende por ejemplo de la experiencia de encuestador, pues consiste en un conjunto de preguntas presentadas a una persona para su respuesta. La forma de la pregunta puede influir en las respuestas, por lo que hay que planearlas cuidadosamente.
- Identificar el requisito que se elicita con su identificador y con su nombre.
- Escribir la descripción dada de ese requisito.
- Argumentar la clasificación del mismo dentro de las especificadas en la plantilla o pudiendo añadir una nueva clasificación en caso de que existiera.

  Observación: Las clasificaciones que aparecen en la plantilla se determinaron primeramente por ser entre las más conocidas a nivel mundial en la ingeniería de requisitos para el caso de los no funcionales y además identificados según el resultado de las encuestas a la muestra intencional tomada en esta investigación, donde quedó identificada una lista de las clasificaciones de los RNF más utilizadas.
- Argumentar observaciones referentes a la clasificación u observaciones generales de esa elicitación.
- Argumentar una posible comparación terminológica o grabación de audio y/o video, esos campos fueron incluido en la planilla dado que también en el estudio mediante la muestra intencional tomada de la población estudiada se identificaron algunos casos

que usaban estas técnicas para la elicitación, pero no se ponen en una plantilla a parte porque estas se consideran más eficiente como complemento a otras técnicas más recomendadas en esta etapa como es el caso de las entrevistas y encuestas y la comparación con sistemas existentes.

- Firmar la planilla una vez realizada.

Acciones para la realización del cuestionario:

- Definir las preguntas según las categorías en que suelen distinguirse. Por ejemplo suelen distinguirse en dos categorías: abiertas y cerradas (Dávila, 2001).
  - ✓ Las preguntas abiertas permiten que los encuestados respondan con su propia terminología. Generalmente estas son más reveladoras, ya que los interrogados no están limitados en sus respuestas. Son especialmente útiles en la etapa exploratoria de la investigación, cuando el analista busca penetrar en el pensamiento del encuestado.
  - ✓ Por su parte, las preguntas cerradas predeterminan todas las posibles respuestas y el interrogado elige entre las opciones presentadas.

Para el caso del método Sistemas Existentes

Esta técnica consiste en analizar distintos sistemas ya desarrollados que estén relacionados con el sistema a ser construido.
Para esta se proponen pasos que coinciden con los especificados para las entrevistas y cuestionarios, pero adicionándole campos para esclarecer la información devenida del análisis de las interfaces de usuario y el análisis de las salidas propios de este método.

Es recomendable que luego de haber analizado el sistema, se le muestre al cliente/usuario, ya que por su experiencia puede sugerir importantes ideas nuevas. (Dávila, 2001)

Fecha	**Hora**	**Lugar**
Datos del Cliente:		
Identificación del sistema existente		
Cuestionario:		
Identificador del requisito		
Nombre del requisito		
Descripción		
Clasificación	**__ Usabilidad** **__ Fiabilidad** **__ Eficiencia** **__ Soporte.** **__ Restricciones del diseño** **__ Requisito para la documentación** **__ Componentes comprados** **__ Interfaz** **__ Requisito de licencia**	**__ Requisitos legales** **__ Estándares aplicables** **__ Seguridad** **__ Escalabilidad** **__ Reusabilidad** **__ Culturales y Políticos.** **__ Características en tiempo de ejecución** **__ Interoperabilidad**
Observaciones de la clasificación:		
Información devenida del análisis de las interfaces de usuario		
Análisis de las salidas		
Comparación de terminología		
Datos y firma del cliente	**Datos y firma del entrevistador**	

Figura 8: Plantilla para elicitación de los RNF por *Comparación con sistemas existentes.*

Análisis

Mediante esta etapa, se pretende en esencia llevar a un nivel superior de detalle los requisitos del sistema (en este caso dirigido a los RNF).

Van a centrarse en tres momentos principales: definir las prioridades, identificar cuáles aspectos son esenciales y en qué momento se requieren. Este proceso tiene mucha relación con otros procesos de la ingeniería, por ejemplo la relación con la gestión de riesgo ya que se analizan los riesgos vinculados a los requisitos. Esto también repercute en la planificación del proyecto ya que se tienen en cuenta para la estimación de esfuerzos requeridos para el desarrollo del software.

Esta etapa es muestra de la interesante propuesta de realizarse de forma iterativa e incremental, donde los requisitos se eliminan, se combinan o modifican de forma de alcanzar por cada parte cierto grado de satisfacción. Realizándose con el objetivo de convertir los requisitos obtenidos de los clientes o usuarios en requisitos reales.

a) Acciones dirigidas a la realización de este procedimiento.

- Obtener las entradas determinadas por los documentos de la elicitación de los requisitos y el listado de los involucrados en el proyecto.
- Definir el equipo involucrado directamente al análisis.
- Trabajar en conjunto (propuesta de utilización de las Tormentas de Ideas), para determinar que los requisitos realmente representen y reflejen las necesidades comunicadas por el cliente y asegurar la correctitud de los mismos (RNF).
- Trabajar en base a los atributos de los RNF, documentar estos atributos.
- Revisar y redefinir la priorización de los RNF.
- Validar con los clientes los requisitos reales obtenidos.

  Observación: Como parte de una propuesta iterativa, si los requisitos no fueron validados (en la etapa de análisis) en una determinada iteración debe regresarse al tercer paso de las actividades dirigidas a la realización de esta etapa.

Para esta etapa del desarrollo de propone la utilización de una plantilla (Figura 9) que se diseñó teniendo en cuenta el estudio de la estructura de una plantilla para la elicitación (Ver Anexo 2) pero incorporándole elementos propios de la etapa de análisis.

Identificación del RNF	
Autor(es)	<autor de la versión actual> (<organización del autor>) **……**
Fuentes	<fuente de la versión actual>
Objetivos asociados	OBJ-x <nombre del objetivo> **……**
Requisitos asociados	<nombre del requisito> **……**
Estado	__ Propuesto __ Aceptado __ Rechazado
Prioridad	__ Critico __ Importante __ Útil
Riesgo	__ Bajo (< 10%) __ Medio (10-50%) __ Alto (> 50%)
Impacto en la arquitectura	__ Ninguno __ Extensión/Modificación
Estabilidad % de cambio esperado	__ Bajo (< 10%) __ Medio (10-50%) __ Alto (>50%)
Nivel de peligro/ criticidad	__ Nula (no hay daño personal o material) __ Baja (puede ser controlada sin lesión personal o significativos daños al sistema) __ Crítica (puede provocar lesiones personales o grandes daños al sistema o requerir acciones correctivas inmediatas para supervivencia de personas y sistemas) __ Catastrófica (puede causar serias lesiones, muerte o pérdida completa del sistema)

Figura 9: Plantilla para el análisis de RNF en el proceso de desarrollo de los mismos.

Los requisitos asociados pueden ser tanto requisitos funcionales como no funciones y se sugiere se analice también teniendo en cuenta cómo puede repercutir un RNF en otro RNF, apoyándose en la tabla (Anexo 1) que visualiza información referente a como se compensan los RNF. Y atendiendo a esto también se identifican riesgos asociados.

Como se puede ver, se incorpora al análisis un grupo de atributos donde se identifican por ejemplo:

- ✓ la fuente, que es el involucrado que originó el requisito.
- ✓ el estado, definido como propuesto, aceptado o rechazado.
- ✓ la prioridad definida como crítica, importante o útil.
- ✓ el riesgo interpretado como bajo en menos del 10%, medio cuando oscila entre el 10% y el 50%, y alto cuando es mayor que el 50 %.
- ✓ La visión del impacto en la arquitectura pudiendo ser ninguna o extensión/modificación.
- ✓ La estabilidad dada en niveles de porciento de cambio esperado y de esta forma definida en niveles bajos, medios y altos.
- ✓ El nivel de peligro/criticidad, en categorías que definen el nivel de daño personal y/o material.

El análisis por su parte puede identificarse en la presente estrategia unido a la elicitación de los requisitos, puesto que estos aspectos pueden trabajarse en común, se puede identificar la lectura de los requisitos, conceptualización e investigación de los mismos, incluso en el estudio previo que pueda generar una tormenta de ideas en preparación de una entrevista u otro encuentro con el cliente.

Se realiza en ambos casos por tanto el intercambio de ideas con el resto del equipo, para resaltar los problemas encontrados y la búsqueda de alternativas y soluciones. Es importante que exista como actividad esencial en la cumbre de esta etapa la planificación de reuniones con el cliente para discutir los requisitos.

En caso de la complejidad del proyecto, el tamaño del mismo, pueden decidirse fusionar o no estas etapas en una sola, pero siempre se sugiere se desenvuelvan incrementalmente como etapas consecuentes.

Especificación

Acciones dirigidas a la especificación de los RNF.

- Especificar el alcance, determinado por los proyectos con los que se involucra la especificación.
- Establecer definiciones, acrónimos y abreviaturas.
- Dejar especificadas las referencias, pudiendo ser documentos a los que se hacer referencia en la especificación.

Se trabajará con la siguiente clasificación identificada para el uso de esta estrategia, teniendo en cuenta cuáles son las consideraciones frecuentemente aceptadas a nivel mundial, y cuáles son las más utilizadas en nuestra universidad. Para lo cual se tuvieron en cuenta los resultados arrojados por la encuesta y el estudio de la planilla de la especificación de requisitos perteneciente al expediente de proyecto aprobado por la dirección de calidad de la UCI.

Determinada entonces la siguiente clasificación de RNF, para establecerse descriptivamente los mismos en la planilla de especificación que propone la presente estrategia.

- **Usabilidad** [Esta sección incluye todos lo requisitos que afectan la usabilidad. Ejemplos:
  - ✓ Especificar el tiempo de entrenamiento requerido para que usuarios normales y avanzados sean productivos operando el sistema.
  - ✓ Especificar requisitos acordes con estándares de usabilidad establecidos]
- **Fiabilidad** [En esta sección se especifican los requisitos relacionados con la Fiabilidad. Ejemplos:
  - ✓ Disponibilidad – especificar porciento de tiempo disponible (xx.xx%), horas de uso, acceso para mantenimiento, modo de funcionamiento degradado etc.
  - ✓ Tiempo medio entre fallos – usualmente se especifica en horas pero puede también especificarse en términos de días, meses o años.
  - ✓ Tiempo medio de reparación – Cuanto tiempo está permitido que el sistema quede fuera de operación luego de haber fallado?

- ✓ Exactitud – especificar la precisión y exactitud requerida en las salidas del sistema.
- ✓ Máximo de errores – usualmente es expresado en términos de errores/MLC (miles de líneas de código) o errores/puntos de función.
  - ✓ Errores – categorizar los errores en términos de menores, significativos y críticos: los requisitos deben definir que se entiende por error crítico (ej. Pérdida total de los datos o inhabilitadas para el uso ciertas partes del funcionamiento del sistema).]

- **Eficiencia** [Deben perfilarse en esta sección las características de la eficiencia del sistema. Incluir los tiempos de respuesta específicos. Donde sea aplicable, hacer referencia a los Casos de Uso por el nombre.
  - ✓ Tiempo de respuesta por transacción (promedio, máximo).
  - ✓ Rendimiento (ej. transacciones por segundo, cantidad de datos que pueden ser transferidos en un segundo).
  - ✓ Capacidad (ej. número de clientes o transacciones que el sistema puede alojar).
  - ✓ Modos de degradación (cual es el modo de operación aceptable cuando el sistema de alguna forma ha sido degradado).
  - ✓ Utilización de recursos (memoria, disco, comunicaciones, etc.]
- **Soporte** [Esta sección indica cualquier requisito que refuerce el soporte o mantenimiento del sistema a construir, incluyendo normas de codificación, convenciones para nombrado, bibliotecas de clase, el acceso y utilidades de mantenimiento.]
- **Restricciones de diseño** [Esta sección debe indicar cualquier restricción de diseño en el sistema a construir. Las restricciones representan decisiones de diseño que se han tomado y a las cuales es necesario adherirse.
  - ✓ (Ej. lenguajes de programación, requisitos de proceso de software, el uso prescrito de herramientas de desarrollo, restricciones de arquitectura y diseño, componentes comprados, las bibliotecas de la clase, etc.]
- **Requisitos para la documentación de usuarios en línea y ayuda del sistema.** [Describe los requisitos para la documentación de usuarios en línea, la ayuda del sistema, ayuda relacionada con avisos, etc.].

- **Componentes Comprados** [Esta sección describe cualquier componente comprado y a ser usado en el sistema, cualquier licencia aplicable o restricciones del uso, y cualquier compatibilidad/interoperabilidad asociada o estándares de interfaz.]
- **Interfaz** [Esta sección define las interfaces que deben soportadas por la aplicación. Debe contener la especificidad adecuada, protocolos, puertos y direcciones lógicas, etc., para que el software pueda desarrollarse y verificarse contra los requisitos de la interfaz.]
  - ✓ **Interfaces de usuario** [Describe las interfaces de usuario que deben ser implementadas por el software.]
  - ✓ **Interfaces Hardware** [Esta sección define cualquier interfaz del hardware que será soportada por el software, incluyendo la estructura lógica, direcciones físicas, el comportamiento esperado, etc.]
  - ✓ **Interfaces Software** [Esta sección describe las interfaces del software a otros componentes del sistema del software. Éstos pueden ser componentes comprados, componentes reutilizados de otra aplicación o componentes que se desarrollan para subsistemas fuera del alcance de este documento, pero con esta aplicación debe actuar recíprocamente.]
  - ✓ **Interfaces de Comunicación** [Describe cualquier interfaz de comunicaciones a otros sistemas o dispositivos como las redes de área locales, los dispositivos remotos, etc.]
- **Requisitos de Licencia** [Define cualquier requisito de licencia o restricción de uso que serán seguidos por el software.]
- **Requisitos Legales, de Derecho de Autor y otros.** [Esta sección describe cualquier denegación legal necesaria, garantías, notificaciones de derecho de autor, patentes, marca comercial o complacencia con logotipo para el software.]
- **Estándares Aplicables** [Esta sección describe por referencia cualquier norma o estándar aplicable y las secciones específicas que aplicadas al sistema. Por ejemplo, podría incluir estándares legales, de calidad, regulatorios, normas de la industria para la usabilidad, el interoperabilidad, internacionalización, integración con el sistema operativo, etc.]

- ***Seguridad*:** requisitos relacionados con la confidencialidad de los datos en la transmisión y en el almacenamiento, junto con las necesidades del sistema para evitar intrusiones no autorizadas al mismo y la capacidad para seguir eventos que comprometan esta seguridad a través del tiempo.
- ***Portabilidad:*** es el esfuerzo requerido a la hora de migrar una parte del software de un ambiente de trabajo a otro. Es la habilidad de vincular o localizar un producto con características específicas.
- ***Reusabilidad:*** reusabilidad o reutilidad indica la conversión de un componente de software con funcionalidades determinadas a otros usos, a otras funciones. El software reusable debe estar bien documentado, debe ser modular, independientemente de las aplicaciones específicas o ambiente de funcionamiento.
- ***Escalabilidad:*** hacen referencia a la capacidad del sistema para acoplar módulos componentes y extensiones, presentan directrices en el diseño y la arquitectura, sugiere el empleo de tendencias en tecnologías que puedan permitir al sistema tener vigencia en su diseño por lo menos en un periodo de tiempo considerable, de tal manera que los componentes de extensión que se propongan puedan ser desarrollados basados en tecnologías conocidas.

Validación

Esta etapa se encuentra en el mismo camino del análisis destinada a encontrar problemas referidos a los requisitos. En similitud a esto, se tiene una entrada, en el caso de análisis se tenía la entrada de un conjunto incompleto de requisitos, ahora en la validación se cuenta como entrada con un conjunto acordado de requisitos.

a) **Acciones dirigidas a la verificación de la especificación de RNF.**
   - Verificaciones de validez.
   - Verificaciones de consistencia y detención de ambigüedades.
   - Verificaciones sobre la completitud.
   - Verificación y chequeo de riesgos (Si se tuvieron en cuenta los RNF que se asocian a cada uno y como puede afectar esa condición al sistema).

Estas cuatro acciones se confeccionaron teniendo en consideración el análisis de la fuente bibliográfica (Kybele, Grupo de investigación, 2007)

Esta verificación incluye la validación interna de las versiones de los requisitos no funcionales.

En estas acciones debe incluirse la rectificación de ideas incluidas en los artefactos que pudieron generarse mediante herramientas de modelado. También se realizan las correcciones correspondientes a los documentos de acuerdo con las observaciones generadas.

Realizar proceso de negociación de los RNF infiere una pre-negociación y una-pos negociación para su realización.

b) Acciones dirigidas a la pre-negociación, negociación y post-negociación.

Mediante una pre-negociación:

- Definición del problema: Identificar problema por el análisis de la situación y definir el propósito de la negociación.
- Identificación de los “stakeholders“(personas vinculadas directamente al sistema que se desea producir): encontrar las personas representativas para la negociación, desde el punto de vista del manejo de los acuerdos y desacuerdos de los requisitos.
- Revisar una posible cumbre iterativa de la elicitación, significa realizar el análisis de la elicitación, y determinar si realmente la más actual de las iteraciones realizadas en ella fue suficiente para el proceso.
- Revisar una posible cumbre iterativa del análisis, determinar si realmente la más actual de las iteraciones realizadas en el análisis arrojó a consideraciones generalizadas de suficiencia y satisfacción. (Ver figura 10)

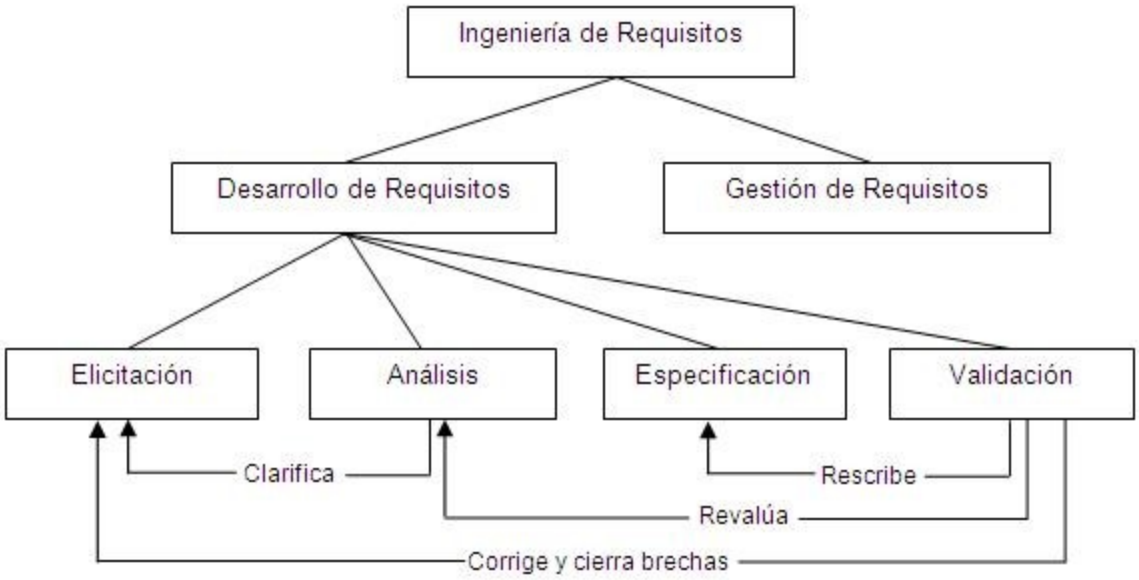

Figura 10: Etapas del desarrollo dentro de la IR.
Consultado de la fuente: (Wiegers, 2006)

(Si fueron identificados aspectos a cambiar como resultado de esas revisiones, se desarrollan soluciones factibles para los stakeholders. Se procede a la explicación de posibles soluciones, es un pre-requisito ante que los stakeholders puedan coincidir con una decisión y requieran del establecimiento de un criterio unánime.)

La negociación por tanto determinará como salidas la elaboración de la documentación de los requisitos no funcionales y como paso esencial la validación de los requisitos no funcionales con los clientes.

Post-negociación es la fase que se analizan y evalúan los resultados de la negociación y pasa a una pre-negociación en caso de que fuera necesario. Es resultado de una propuesta iterativa, orientado a la calidad del proceso de desarrollo, y la satisfacción del/ los clientes.

2.6 Validación por criterio de especialistas

La propuesta “Estrategia para el desarrollo de requisitos no funcionales de software en proyectos productivos de la UCI” fue sometida a criterio de especialistas para determinar su probidad y de este modo su validación. Así mismo se sometió a la consideración de cinco

ingenieros, de ellos 4 pertenecientes al proyecto investigativo de ingeniería de requisitos coordinado por la Dirección de Calidad de la universidad y un profesor del Departamento de Ingeniería de Software de la facultad 5 que por su desempeño se le otorgó el reconocimiento "Premio del Rector" como analista de proyectos productivos.

Se valoró la investigación como una solución novedosa basada en la necesidad del proceso de desarrollo de software que se lleva a cabo en la universidad, considerándose como un tema muy interesante que pudiera ayudar a una mejor organización, estandarización y validación de los requisitos no funcionales. Se consideró la transcendencia que implica en el desarrollo de software, así como un positivo criterio en torno a las técnicas investigativas utilizadas para ello.

"La investigación con título: "Estrategia para el desarrollo de requisitos no funcionales de software en proyectos productivos de la UCI", de los autores Lisett Pérez Quintero y Carlos Manuel Verdecia Guerra aborda uno de los ejes centrales del desarrollo de software, el desarrollo de requisitos. Temática en la que debemos fomentar la investigación, especialización para garantizar la salud de nuestra creciente industria de software.

El trabajo presenta un elevado nivel científico acorde con el estado del arte, sus creadores esbozan con claridad el problema como se plasma en el capítulo de "Diagnóstico del Estado Actual del Problema y Solución Propuesta" y nos brindan una solución de alto valor intelectual que puede ser puesta en práctica con pocos esfuerzos en nuestros proyectos productivos y en la docencia.

Este material revoluciona la ingeniería de requisitos en nuestro entorno y provee a un equipo de desarrollo de software de elementos básicos que tributan en disciplinas como la planificación, arquitectura y pruebas. En manos de un profesor de ingeniería de software fortalecería el estudio y comprensión de conceptos en los que debemos madurar.

Por lo antes expuesto considero que el trabajo científico cumple con los objetivos propuestos. Recomiendo su publicación en eventos investigativos y la aplicación práctica en nuestra universidad." (Ver Anexo 9).

2.7 Conclusiones del capítulo

Mediante el diagnóstico realizado se corroboró que en proyectos productivos de la universidad, no se contaba con una estrategia unificada para el desarrollo de los RNF de software, que estableciera artefactos estándares para etapas de este proceso, no solamente para la elicitación. También fue constatada la necesidad de lograr identificar un grupo de métodos dentro de los más aceptados, y la creación de plantillas que recogieran y agruparan un trabajo dirigido a los tipos de RNF mayormente utilizados. Teniendo esto en cuenta se evidenció la necesidad de una estrategia que contribuyera al desarrollo de los requisitos no funcionales en proyectos productivos de la UCI.

CONCLUSIONES

- Los elementos relacionados con la ingeniería de requisitos, permitieron una orientación teórica hacia la elaboración de la estrategia.
- La investigación tuvo en cuenta un grupo de métodos para el desarrollo de los RNF, e identificó las tendencias en torno a este proceso en la UCI.
- La estrategia propuesta contribuye:
    - Al diagnostico del estado del desarrollo de los requisitos no funcionales.
    - A la definición de un flujo de técnicas, lenguajes y herramientas.
    - A la unificación y potenciación de acciones dirigidas al desarrollo de los requisitos no funcionales.
    - A la provisión del equipo de desarrollo con un conjunto de artefactos para el desarrollo de requisitos no funcionales.

RECOMENDACIONES

- Que se aplique la estrategia propuesta en proyectos productivos de la Universidad de las Ciencias Informáticas.
- Que se efectúe el análisis de la posible vinculación con estrategias docentes e investigativas dentro del marco de la Ingeniería de Software.

BIBLIOGRAFÍA

1. **Alan M. Davis, Fellow, IEEE, Oscar Dieste, Member, IEEE, Ann M. Hickey, Member, IEEE, Natalia Juristo, Fellow, IEEE and Ana M. Moreno, Senior Member, IEEE.** Producción Científica en Ingeniería de Requisitos en España: Un Análisis en el Contexto Europeo. [En línea] http://ieeexplore.ieee.org/iel5/9907/34417/01642460.pdf.
2. **Arango, J. 2002.** *Tormenta de Ideas.* Colombia : Universidad EAFIT, 2002.
3. **Arias Chaves, Michael. 2007.** Revista de las Sedes Regionales de la Universidad de Costa Rica. *Revista de las Sedes Regionales de la Universidad de Costa Rica.* [En línea] 30 de 07 de 2007. [Citado el: 25 de 04 de 2008.] http://www.intersedes.ucr.ac.cr/pdfs_10/10-art_11.pdf. ISSN 1409-4746.
4. **Brito, I, Moreira, A and Araújo, J. 2002.** Universidad Twente. *A requeriments model for quality attributes.* [Online] 04 22, 2002. [Cited: 02 25, 2008.] http://www.utwente.nl/AOSD-EarlyAspectsWS/Papers/Brito.pdf.
5. **Chung, Lawrence, y otros. 1999.** *Non-Functional Requirements in Software Engineering.* London : Kluwer Academic Publishers, 1999. ISBN: 0792386663.
6. **CMMI Product Team. 2006.** *CMMI for Developments Version 1.2.* Pensilvania : Carnegie Mellon University, 2006. PA 15213-3890.
7. **Cysneiros, Luiz Marcio y Sampaio do Prado Leite, Julio César. 2001.** *Using UML to Reflect Non-Functional Requirements.* Brasil : s.n., 2001.
8. **Dávila, Nicolás Davyt. 2001.** Ingeniería de Requerimientos una guía para extraer, analizar, especificar y validar los requerimientos de un proyecto. *http://webs.montevideo.com.uy.* [En línea] 2001. [Citado el: 15 de Enero de 2008.] http://webs.montevideo.com.uy/nicolasd.
9. **Díez González, Oscar. 2006.** *Safety y Requisitos No Funcionales.* Madrid : Universidad politécnica de Madrid, 2006.
10. **DIGIÓN, Leda Beatriz. 2007.** UN ESTUDIO INICIAL DE REQUERIMIENTOS NO FUNCIONALES A TRAVES DEL CASO DE USO1. *Jornadas de Investigación y Desarrollo en Ingeniería de Software (JIDIS 2007).* [En línea] 2007. http://www.jidis.frc.utn.edu.ar/viewabstract.asp?id=29.

11. **Durán Toro, Amador y Bernárdez Jiménez, Beatriz. 2000.** *Metodología para la elicitación de requisitos de software. Versión 2.1.* Sevilla : Ministerio de Educación y Ciencia de España, 2000. LSI–2000–10.

12. **Durán, A. Bernardez, J. 2000.** *Metodología para la elicitación de requisitos de software. Versión 2.1.* Sevilla : s.n., 2000.

13. **Escalona, María José y Nora, Koch. 2002.** Ingeniería de Requisitos en Aplicaciones para la Web, un estudio comparativo. *Departamento de Lenguajes y Sistemas Informáticos. Universidad de Sevilla.* [En línea] Universidad de Sevilla, 2002. [Citado el: 10 de Diciembre de 2007.] http://lsiweb.lsi.us.es/docs/informes/LSI-2002-4.pdf.

14. **González-Baixauli, Bruno, Laguna, Miguel A y Sampaio do Prado Leite, Julio Cesar. 2005.** *Aplicación de un Enfoque Intencional al Análisis de Variabilidad.* Porto : Departamento de Informática. Universidad de Valladolid,Departamento de Informática. PUC do Rio de Janeiro, 2005.

15. **Grünbacher, Paul y Seyff, Norbert. 2005.** Requirements Negotiation. [aut. libro] Aybüke Aurum y Wohlin Claes. *Engineering and Managing Software Requirements.* Berlin : Springer, 2005, págs. 143-148.

16. **Hernández León, Rolando Alfredo y Coello González, Sayda. 2002.** *El paradigma cuantitativo de la investigación científica.* La Habana : Editorial Universitaria, 2002. ISBN: 959-16-0343-6.

17. **IEEE Standard. 1990.** *IEEE Standard Glossary of Software Engineering Terminology.* s.l. : Institute of Electrical and Electronics Engineers, 1990. 610.12–1990.

18. **Jackson, Michael. 2001.** *Software Requirements and Specifications.* s.l. : Addison-Wesley, 2001. ISBN: 0-201-87712-0.

19. **Jacobson, Ivan, Booch, Grady y Rumbaugh, James. 2000.** *El proceso unificado de desarrollo de software.* Madrid : Pearson Educación S.A., 2000. ISBN: 84-7829-036-2.

20. **Kaplan, Gladys N, y otros. 2001.** *Inspección del Léxico Extendido del Lenguaje.* Buenos Aires : s.n., 2001. págs. 70-91.

21. **Kybele, Grupo de investigación. 2007.** Universidad Rey Juan Carlos. *Universidad Rey Juan Carlos.* [En línea] 2007. [Citado el: 3 de abril de 2008.] http://kybele.escet.urjc.es/docencia/IS3/2007-2008/Material/%5BIS3-2007-2008%5DADE.Tema1.pdf.

22. **Malan, Ruth, Brendmeyer, Dana. 2001.** Resources for Software Architects. *Defining Non-Functional Requeriments.* [En línea] 2001. [Citado el: 10 de 12 de 2007.] http://www.bredemeyer.com/pdf_files/NonFunctReq.PDF.

23. **Marqués, José M. 2005.** Ingeniería del Software. *Universidad de Valladolid.* [En línea] 16 de 09 de 2005. http://www.infor.uva.es/~jmmc/ingsoft/isprincipal.html.

24. **Martín Cordero, Dayamí. 2007.** *Propuesta de estrategia para gestionar el conocimiento en la Dirección de Calidad de Software de la Universidad de las Ciencias Informáticas.* La Habana : Trabajo de Diploma. UCI, 2007.

25. **McGraw, Gary, Hope, Paco y Anton, Annie I. 2004.** *Misuse and abuse cases: getting past the positive.* North Carolina : Gary McGraw, 2004. págs. 90-92.

26. **Monferrer Agut, Raúl. 2001.** *Especificación de Requisitos Software según el estándar de IEEE 830.* España : s.n., 2001.

27. **Pressman, Roger. 2002.** *Ingeniería de software.Un enfoque práctico.* Madrid : McGraw-Hill, 2002.

28. **Robertson, Suzanne y Robertson, James. 1999.** *Mastering the Requirements Process.* Inglaterra : Pearson, 1999.

29. **Sabat Neto, Jaime de Melo, Cysneiros, Luiz Marcio y Sampaio do Prado Leite, Julio César. 2001.** A Framework for Integrating Non-Functional Requirements into Conceptual Models. *Requirements Engineering.* Rio de Janeiro : Springer, 2001.

30. **Sampaio do Prado Leite, Julio César y Cysneiros, Luiz Marcio. 2001.** *Using the Language Extended Lexicon to Support Non-Functional Requirements Elicitation.* Buenos Aires : s.n., 2001. págs. 139-151.

31. **Siglo21, Informática. 2006.** *Documento De Especificación Requerimientos No Funcionales del Proyecto Mejoramiento de procesos, Análisis y Diseño del Sistema de Información para la Vigilancia de eventos en salud pública en la Fase 1.* Colombia : s.n., 2006.

32. **Sommerville, Ian. 2004.** *Ingeniría de Software.* México : Addison-Wesley, 2004.

33. **Straud Barros, Pablo. 2005.** *Requisitos de Software.* Santiago de Chile : Universidad Diego Portales, 2005.

34. **Sumaro López, Maria de los Angeles. 2001.** *Áncora: Metodología para el Análisis de Requerimientos de Software conducente al Reuso.* Estado de Veracruz : s.n., 2001.

35. **Tabares, Marta S, Anaya, Raquel y Arango, Fernando. 2007.** *La ingeniería de requisitos orientada a aspectos: Una experiencia de aplicación en un sistema de línea.* Medellín : s.n., 2007. ISSN 0012-7353.

36. **Thayer, Richard H, y otros. 1997.** *Software Engineering.* s.l. : John Wiley & Sons, 1997. ISBN:076951555X.

37. **UCI. 2008.** Universidad de las Ciencias Informáticas. *Universidad de las Ciencias Informáticas.* [En línea] 2008. [Citado el: 19 de abril de 2008.] www.uci.cu.

38. **Wiegers, Karl Eugene. 2006.** *More About Software Requirements: Thorny Issues and Practical Advice.* s.l. : Microsoft Press, 2006.

39. **—. 2003.** *Software Requirements, Second Edition.* Washington : Microsoft Press. A Division of Microsoft Corporation One Microsoft Way., 2003. 98052-6399.

40. **Young, Ralph Rowland. 2004.** *The Requirements Engineering Handbook.* London : Artech House, 2004.

ANEXOS

Anexo 1: Matriz de cómo un requisito no funcional se equilibra con otros.

	Dis	Efic	Fle	Int	Inr	Man	Por	Con	Reu	Rob	Com	Usa
Disponibilidad								+		+		
Eficiencia			-		-	-	-	-		-	-	-
Flexibilidad		-		-		+	+	+			+	
Integridad		-			-				-		-	-
Interoperabilidad		-	+	-			+					
Mantenibilidad	+	-	+					+			+	
Portabilidad		-	+		+	-						
Confiabilidad	+	-	+			+						
Reusabilidad		-	+	-	+	+	+	-				
Robustez	+	-						+				
Comprobabilidad	+	-	+			+		+				
Usabilidad		-								+	-	

Tabla 2: Equilibrio entre requisitos no funcionales de acuerdo a los atributos de calidad.

- El signo de más (+) representa un aumento en la calidad del atributo que se encuentra en la fila correspondiente afectando positivamente el atributo que se encuentra en la columna,
- El signo de menos (-) representa que el aumento en el atributo correspondiente a la fila, afecta negativamente al atributo de la columna correspondiente, ejemplo, el aumento de la eficiencia tiene un impacto negativo en la mayoría de los atributos de calidad restantes, pues si se escribe el código de manera rápida, usando codificaciones personalizadas, confiando en la eficiencia de la ejecución, lo más probable sea que el código no sea reutilizable luego y a la hora de migrar a otra plataforma sea una situación complicada.
- Las celdas en blanco representan que la contraposición que existe entre ambos atributos es mínima, de muy poco impacto, casi nula.
- La matriz no es simétrica, pues si se tiene dos atributos, uno A y otro B, no hay necesidad de que el impacto que provoque el atributo A sobre el B sea el mismo que luego B sobre A.

Anexo 2: Plantilla para la elicitación de requisitos no funcionales propuesta por Bernandez y Duran.

RNF-<id>	<nombre descriptivo>
Versión	<n de la versión actual>(<fecha de la versión actual>)
Autores	<autor de la versión actual> (<organización del autor>) **……**
Fuentes	<fuente de la versión actual>(<organización de la fuente>) **……**
Objetivos asociados	OBJ-x <nombre del objetivo> **……**
Requisitos asociados	Rx-y <nombre del requisito> **……**
Descripción	El sistema deberá… <capacidad del sistema>
Importancia	<importancia del requisito>
Urgencia	<urgencia del requisito>
Estado	<estado del requisito>
Estabilidad	<estabilidad del requisito>
Comentarios.	<comentarios adicionales sobre el requisito>

Identificador y nombre descriptivo: cada requisito se debe identificar por un código único y un nombre descriptivo. Con objeto de conseguir una rápida identificación, los identificadores de los requisitos no funcionales comienzan con RNF.

Versión: para poder gestionar distintas versiones, este campo contiene el número y la fecha de la versión actual del requisito.

Autores, Fuentes: estos campos contienen el nombre y la organización de los autores (normalmente desarrolladores) y de las fuentes (clientes o usuarios), de la versión actual del requisito, de forma que la rastreabilidad pueda llegar hasta las personas que propusieron la necesidad del requisito.

Objetivos asociados: este campo debe contener una lista con los objetivos a los que está asociado el requisito. Esto permite conocer qué requisitos harán que el sistema a desarrollar alcance los objetivos propuestos y justifican de esta forma la existencia o propósito del requisito.

Requisitos asociados: en este campo se indican otros requisitos que estén asociados por algún motivo con el requisito que se está describiendo, permitiendo así tener una rastreabilidad horizontal.

Descripción: se usa un patrón–L que debe completarse con la capacidad que deberá presentar el sistema. Este seria el campo más importante de la tabla.

Importancia: este campo indica la importancia del cumplimiento del requisito para los clientes y usuarios. Se puede asignar un valor numérico o alguna expresión enumerada como vital, importante o quedaría bien. En el caso de que no se haya establecido aún la importancia, se puede indicar que está por determinar (PD), equivalente al TBD (To Be Determined) empleado en las especificaciones escritas en inglés.

Urgencia: este campo indica la urgencia del cumplimiento del requisito para los clientes y usuarios en el supuesto caso de un desarrollo incremental. Como en el caso anterior, se puede asignar un valor numérico o una expresión enumerada como inmediatamente, hay presión o puede esperar, o PD en el caso de que aún no se haya determinado.

Estado: este campo indica el estado del requisito desde el punto de vista de su desarrollo. El requisito puede estar en construcción si se está elaborando, pendiente de negociación si tiene algún conflicto asociado pendiente de solución, pendiente de validación si no tiene ningún conflicto pendiente y está a la espera de validación o, por último, puede estar validado si ha sido validado por clientes y usuarios.

Estabilidad: este campo indica la estabilidad del requisito, es decir una estimación de la probabilidad de que pueda sufrir cambios en el futuro. Esta estabilidad puede indicarse mediante un valor numérico o mediante una expresión enumerada como alta, media o baja o PD en el caso de que aún no se haya determinado.

Comentarios: cualquier otra información sobre el requisito que no encaje en los campos anteriores puede recogerse en este apartado. (Durán, 2000)

Anexo 3: Plantilla para el documento de especificación.

IDENTIFICADOR DE REQUERIMIENTO	**NOMBRE**
CLASIFICACION:	
DESCRIPCION:	
CONSIDERACIONES:	

Plantilla utilizada en un documento de especificación de RNF. (Siglo21, Informática, 2006)

Anexo 4: Estructura de una Especificación de Requerimientos de Software presentada por el estándar de IEEE 830

1 Introducción
- 1.1 Propósito
- 1.2 Ámbito del Sistema
- 1.3 Definiciones, Acrónimos y Abreviaturas
- 1.4 Referencias
- 1.5 Visión general del documento

2 Descripción General
- 2.1 Perspectiva del Producto
- 2.2 Funciones del Producto
- 2.3 Características de los usuarios
- 2.4 Restricciones
- 2.5 Suposiciones y Dependencias
- 2.6 Requisitos Futuros

3 Requisitos Específicos
- 3.1 Interfaces Externas
- 3.2 Funciones
- 3.3 Requisitos de Rendimiento
- 3.4 Restricciones de Diseño
- 3.5 Atributos del Sistema
- 3.6 Otros Requisitos

4 Apéndices

5 Índices.

Anexo 5: Proceso de gestión de requisitos.

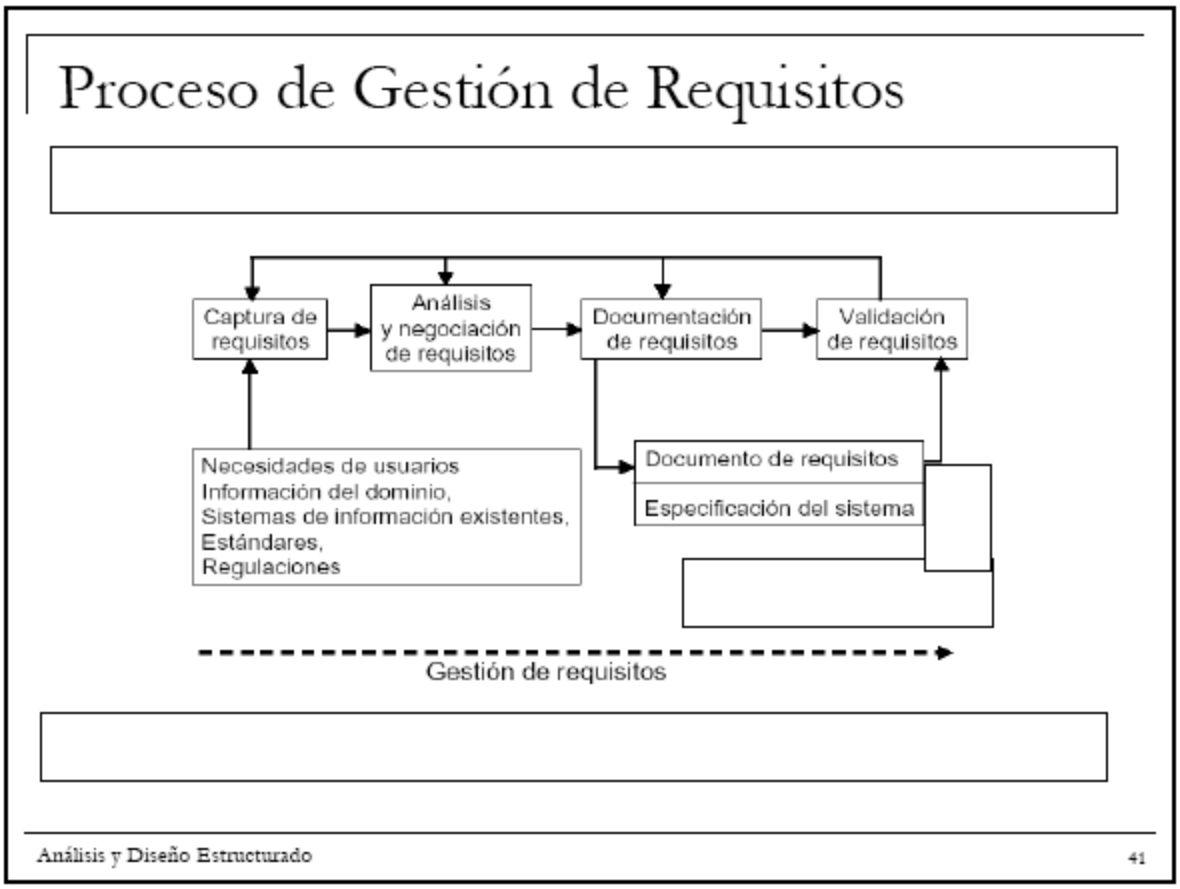

Figura 11: Proceso de Gestión de Requisitos.

Tomado de: http://kybele.escet.urjc.es/docencia/IS3/2007-2008/Material

Anexo 6: Descripción gráfica del framework de integración de requisitos no funcionales con modelos conceptuales.

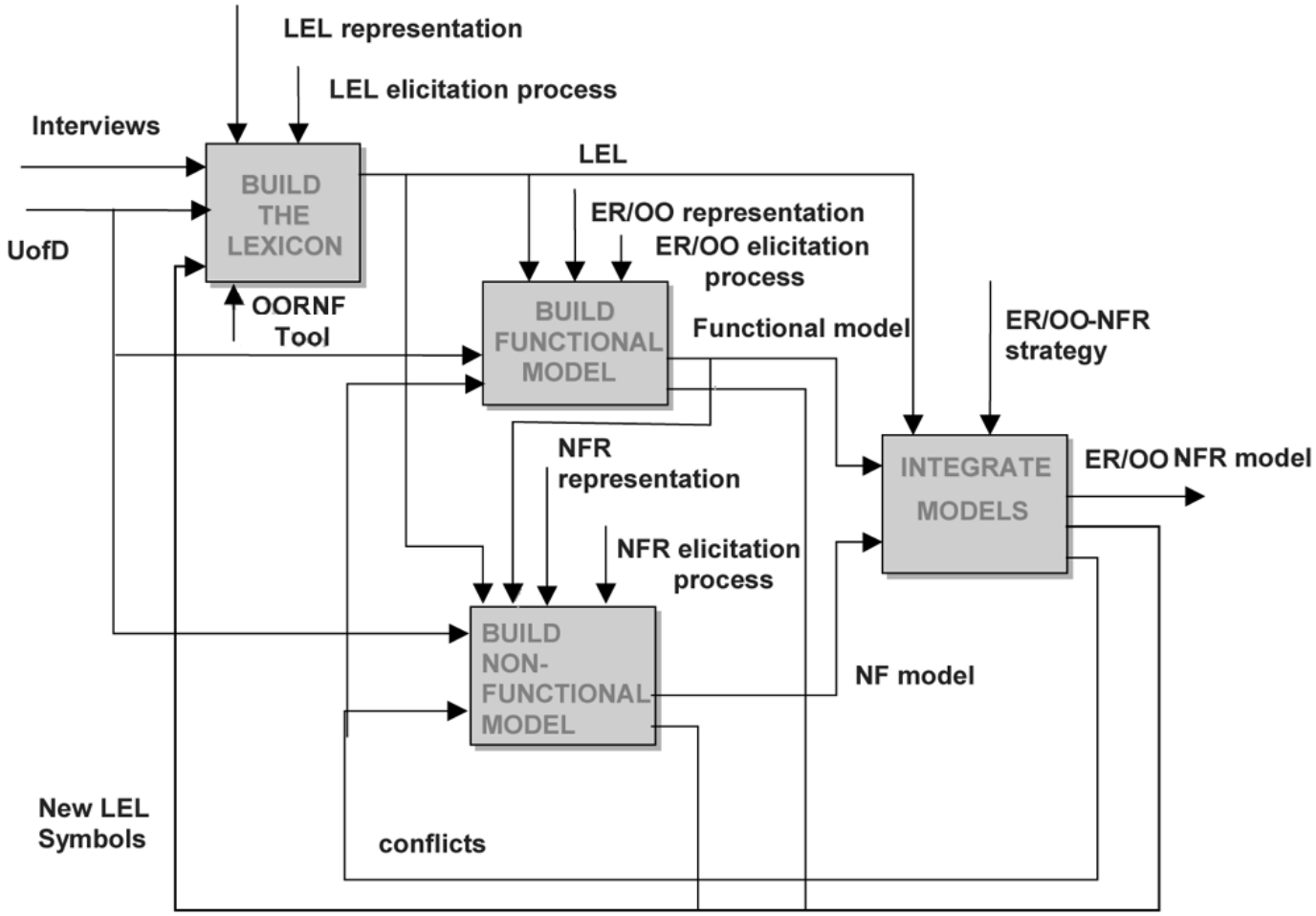

Figura 12: Descripción del framework de integración de RNF con modelos conceptuales. (Sabat Neto, y otros, 2001)

Anexo 7: Encuesta

Compañero(a):

Sus respuestas a las preguntas de la presente encuesta son de gran importancia para nuestra investigación referida al desarrollo (obtención, análisis, especificación y validación) de los requisitos no funcionales (RNF) de software en proyectos productivos de la Universidad de las Ciencias Informáticas.

Agradecemos su sinceridad absoluta en las respuestas a las preguntas que le presentamos a continuación y la ayuda que nos brinda.

1.1 ¿Qué actividades considera se debe realizar con alto grado de importancia durante la fase de Inicio del Proceso de desarrollo del Software, concretamente en el flujo de trabajo de Requisitos?
Seleccione cuales de estas y en caso de realizar otra(s) menciónelas a continuación.

___ Captura de un vocabulario común.
___ Encontrar actores y casos de uso.
___ Desarrollar el plan de gestión de los requisitos.
___ Desarrollar el documento visión.
___ Administrar las dependencias.
___ Estructurar el modelo de casos de uso.
___ Determinar las necesidades (requisitos) del cliente.

_____
_____
_____
_____

1.2 ¿Utiliza un modelo para capturar los requisitos del cliente?

___ Sí ___ No

¿Cuál?

_____

_____

1.3 ¿Tiene usted identificado los pasos a seguir para realizar la actividad mencionada en la pregunta anterior?

___ Sí ___ No ___ Depende del proyecto

1.4 De los siguientes aspectos seleccione cual piensa usted influyen en la calidad de la captura de los RNF de software.

__ El adecuado lenguaje de comunicación con el usuario.
__ El tiempo de duración la entrevista.
__ El entorno de desarrollo de la entrevista.
__ La firma con el cliente de los requisitos no funcionales que se identifican.
__ El grado de importancia con que se valore establecer previamente durante la entrevista con el cliente estos requisitos.
__ No clasificar estos requisitos o clasificarlos incorrectamente.
__ No adoptar un modelo para esta actividad.
__ Conocer buenas prácticas u experiencias vividas durante esta actividad en otros proyectos productivos de la UCI.

__ Otro(s). ¿Cuál (es)?
_____
___
_____
_____
_____

1.5 ¿Piensa que puede ser obviado el desarrollo de los RNF durante la fase de inicio en dependencia del tamaño o complejidad del proyecto? Explique brevemente por qué.

__ Sí. Porque_____
__ No. Porque_____

1.6 ¿Cómo valora la importancia de la actividad de especificar los requisitos de software?

___ Alta ___ Media ___ Baja ___ Nula

1.7 ¿Cómo valora la importancia de esta actividad para el caso de los RNF?

___ Alta ___ Media ___ Baja ___ Nula

1.8 ¿Qué tipos de RNF ha identificado en proyectos productivos?

___ Requisitos de Rendimiento
___ Requisitos de Fiabilidad
___ Requisitos de Disponibilidad
___ Requisitos de Seguridad
___ Requisitos de Portabilidad
___ Requisitos de Mantenibilidad
___ Requisitos de Escalabilidad
___ Requisitos de Reusabilidad
___ Requisitos de Interfaz Externa
___ Requisitos de Amigabilidad (Usabilidad)
___ Requisitos de Capacidad (Consumo de recursos)
___ Requisitos Culturales y Políticos (Robertson & Robertson)
___ Requisitos Legales (Robertson & Robertson)
___ Requisitos de Desempeño (Brito, Moreira y Araújo)
___ Requisitos Operacionales (Brito, Moreira y Araújo)
___ Requisitos Económicos (Brito, Moreira y Araújo)
___ Requisitos de Características en Tiempo de Ejecución (Malan y Bredemeyer)
___ Requisitos de Tiempo de Mercadeo (Viega y McGraw)
___ Requisitos de Simplicidad (Viega y McGraw)
___ Requisitos de Precisión (Consistencia Interna – Chung y Nixon)
___ Requisitos de Precisión (Consistencia Externa – Chung y Nixon)

___ Otro(s). ¿Cuál(es)? _____
_____

1.9 ¿Que métodos utiliza para identificar RNF?

___ Entrevistas y Cuestionarios
___ Comparación de terminología
___ Tormenta de Ideas
___ Sistemas existentes
___ Grabaciones de Audio y Video
___ Otro(s). ¿Cuál(es)? _____
_____

1.10 Que lenguaje considera más apropiado y eficiente para la especificación de estos RNF.

___ Lenguaje Natural
___ Lenguaje de Modelado ¿Cuál? _____
___ Lenguaje Formal ¿Cuál? _____
___ Otro. ¿Cuál? _____

1.11 ¿Qué le da la medida de la suficiencia alcanzada durante la especificación en el proceso de desarrollo? ¿De que vías de comprobación se apoya para esto?

___ Métricas orientadas al cliente
___ Métricas orientadas a la arquitectura
___ Otro(s). ¿Cuál(es)? _____
_____

1.12 ¿Qué herramientas se utilizan para el modelado de los RNF?

___ NFR Framework de Chung
___ NFR con UML
___ Otro(s). ¿Cuál(es)? _____
_____

2 ¿Intercambia experiencia del trabajo realizado relacionado con esta actividad con especialistas de otros proyectos que realicen funciones similares en el mismo o estén relacionados activamente con el tema?

___ Si ___ Con mucha frecuencia ___ Muy pocas veces ___ No

En caso de ser afirmativa la respuesta anterior explique brevemente cómo.
_____
_____
_____
_____

Anexo 8: Entrevista

1. ¿Qué importancia presentan los requisitos no funcionales en el proceso de desarrollo de software?
2. ¿Se lleva a cabo en la universidad alguna estrategia o modelo para trabajar con dichos requisitos?
3. ¿Quién o quiénes intervienen en el proceso de elicitar los requisitos no funcionales?
4. ¿Se le da algún seguimiento a los requisitos no funcionales una vez capturados?
5. ¿Cuál o cuáles son las técnicas de recopilación de información más utilizadas para elicitar los requisitos no funcionales?
6. Mencione otras técnicas, lenguajes o herramientas utilizadas para el desarrollo de requisitos no funcionales.
7. ¿Qué clasificaciones de requisitos no funcionales considera más utilizadas?
8. ¿Cree importante el intercambio de experiencias entre los recursos humanos de los diferentes proyectos sobre el tema de los requisitos no funcionales?
9. ¿Considera necesario la elaboración de una estrategia para el desarrollo de los requisitos no funcionales?

Anexo 9: Criterio de especialista

La Habana, 13 de mayo de 2008

Criterio de especialista

La investigación con título: "Estrategia para el desarrollo de requisitos no funcionales de software en proyectos productivos de la UCI", de los autores Lisett Pérez Quintero y Carlos Manuel Verdecia Guerra aborda uno de los ejes centrales del desarrollo de software, el desarrollo de requisitos. Temática en la que debemos fomentar la investigación, especialización para garantizar la salud de nuestra creciente industria de software.

El trabajo presenta un elevado nivel científico acorde con el estado del arte, sus creadores esbozan con claridad el problema como se plasma en el capítulo de "Diagnóstico del Estado Actual del Problema y Solución Propuesta" y nos brindan una solución de alto valor intelectual que puede ser puesta en práctica con pocos esfuerzos en nuestros proyectos productivos y en la docencia.

Este material revoluciona la ingeniería de requisitos en nuestro entorno y provee a un equipo de desarrollo de software de elementos básicos que tributan en disciplinas como la planificación, arquitectura y pruebas. En manos de un profesor de ingeniería de software fortalecería el estudio y comprensión de conceptos en los que debemos madurar.

Por lo antes expuesto considero que el trabajo científico cumple con los objetivos propuestos. Recomiendo su publicación en eventos investigativos y la aplicación práctica en nuestra universidad.

Ing. Amado Espinosa Hidalgo
Dpto. Ingeniería de Software, Facultad 5
Universidad de las Ciencias Informáticas.

GLOSARIO DE TÉRMINOS

1 **Stakeholders:** En la gestión de proyectos, los involucrados o interesados (stakeholders en inglés) son todas aquellas personas u organizaciones que afectan o son afectadas por el proyecto, ya sea de forma positiva o negativa. Una buena planificación de proyectos debe involucrar la identificación y clasificación de los interesados, así como el estudio y la determinación de sus necesidades y expectativas. Persona interesada o involucrada en el desarrollo de un sistema bajo una perspectiva. Esta puede ser económica o relacionada con otro beneficio por el desarrollo del sistema.

2 **Escenarios:** Son descripciones de situaciones particulares del macrosistema, escritos en lenguaje natural y presentando una simple estructura.

3 **Universo de discurso (UdeD):** es el contexto general donde el software se debe desarrollar, y debe funcionar. Este incluye a todas las fuentes de información así como a todos aquellos usuarios que están relacionados con el software. Estos usuarios también se les suele conocer como Actores en el universo de discurso.

4 **Softgoals:** es la representacion de un requisito no funcional de acuerdo al fremawork de Chung. Se pueden dividir en subsoftgoals y se representan en un gráfico de integracion con forma de nubes, existiendo interrelación entre ellos.

5 **Modelo Conceptual:** modelo que define vistas que representan la organización de los componentes, agentes o elementos de software que participan para lograr la funcionalidad requerida por el sistema.

6 **Técnica**: es un procedimiento o conjunto de procedimientos que tienen como objetivo obtener un resultado determinado, ya sea en el campo de la ciencia, de la tecnología, del arte o en cualquier otra actividad.

7 **Framework**: estructura de software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación.