

“UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS”

FACULTAD 1



**Propuesta de realización de la Ingeniería de Requisitos en
los proyectos de la Universidad de las Ciencias
Informáticas (UCI).**

TRABAJO PARA OPTAR POR EL TÍTULO DE INGENIERÍA EN INFORMÁTICA

AUTOR

Dailiany Soler Torriente

TUTOR

Ing. Yircy Diem Collazo Marín

Ciudad de la Habana, Junio 2007

“El trabajo del pensamiento se parece a la perforación de un pozo: el agua es turbia al principio, más luego se clarifica”

Chofi

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al Departamento de Calidad y Vicerrectoría de Formación de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Dailiany Soler Torriente

Yircy Diem Collazo Marín

OPINIÓN DEL USUARIO DEL TRABAJO DE DIPLOMA.

El Trabajo de Diploma titulado “Propuesta de realización de la Ingeniería de Requisitos en los proyectos de la Universidad de las Ciencias Informáticas (UCI)”, fue desarrollado en la Universidad de las Ciencias Informáticas. Esta entidad considera que, en correspondencia con los objetivos trazados, el trabajo realizado le satisface:

Totalmente _____

Parcialmente en un _____ %

Los resultados de este Trabajo de Diploma le reportan a esta entidad los beneficios siguientes:

Y para que así conste, se firma la presente a los ____ días del mes de _____ del año _____.

Representante de la entidad

Cargo

Firma

Cuño

AGRADECIMIENTOS

- A la Revolución Cubana, a nuestro Comandante en Jefe Fidel Castro Rúz y a la Universidad de las Ciencias Informáticas, por haberme dado la posibilidad de realizar nuestros estudios y obtener los conocimientos necesarios para realizar mi proyecto de tesis.
- A mis padres y hermano por haberme guiado siempre, gracias por su apoyo y dedicación. Por ustedes hoy soy lo que soy.
- A Noslen por comprenderme durante todo este tiempo.
- A mis amigos por estar junto a mí en estos años de estudios.
- A mi Vicedecana Matilde por haberme ayudado tanto durante estos 5 años.
- A mi tía Marina por comprenderme y quererme como una hija.
- A mis vecinos y familiares por creer en mí y apoyarme siempre.
- Y a todos aquellos que de una forma u otra han contribuido a mi formación personal y como futuro profesional, a ellos les digo que siempre estarán en mi corazón, gracias por ser como son.

DEDICATORIA

... a mis padres y hermano

RESUMEN

Cuando se desarrolla un sistema basado en ordenador la primera tarea, y quizás la más importante de todo el proceso, consiste en captar las necesidades de los clientes y usuarios. Este proceso se llama Ingeniería de Requisitos y ha sido reconocido como la etapa crucial de todo el proceso de desarrollo. Los escenarios son una herramienta valiosa para capturar y comprender requisitos y para analizar las interacciones entre personas y ordenadores. Un proceso estándar de ingeniería de requisitos basado en la utilización de escenarios tiene tres tareas principales. Entre la primera y segunda etapa se genera, a partir de los escenarios, especificaciones que describen el comportamiento del sistema. La tercera consiste en validar los escenarios con los usuarios mediante técnicas de simulación o prototipado. En este trabajo presentamos una propuesta de práctica asociada al campo de la ingeniería de requisitos. El método contiene tres etapas: captura, análisis y validación, las cuales cuentan, para su ejecución con analista del sistema, arquitecto, diseñador de interfaz, revisor y especificador de requisitos. Estas etapas incluyen la gestión de cambios de requisitos. Además se propone un curso de capacitación para estudiantes con el fin de lograr una mayor enseñanza sobre el tema de Ingeniería de requisitos. La validación de la propuesta dada es otra sección de este trabajo.

Palabras claves: Ingeniería de requisitos, gestión de requisitos, elicitación.

ÍNDICE GENERAL

INTRODUCCIÓN	1
Capítulo I: Estado del arte.	5
1.1 ¿Por qué surge la Ingeniería de software?	5
1.2 Ingeniería de requisitos	7
1.3 Importancia de la ingeniería de requisitos.....	11
1.3.1 Definición de ingeniería de requisitos.....	11
1.3.2 Dimensiones de la ingeniería de requisitos	13
1.4 Concepto de requisito. Características.....	14
1.4.1 Dimensiones de los requisitos.	14
1.4.2 Propiedades de los requisitos.....	15
1.4.3 Otras características de los requisitos	17
1.5 Herramientas para la Ingeniería de requisitos.....	19
1.6 Ingeniería de requisitos en el mundo	21
1.6.1 Otros modelos utilizados en la Ingeniería de requisitos.....	22
1.6.1.1 El modelo de Pohl	22
1.6.1.2 El modelo espiral.....	23
1.7 Conclusiones.....	24
Capítulo II: Solución propuesta	25
2.1 Estudio de la Ingeniería de Requisitos en los proyectos de la UCI	25
2.2 Propuesta para la realización de la Ingeniería de Requisitos en los proyectos productivos en la UCI	28
2.2.1 Captura de Requisitos	31
2.2.1.1 Obtener información sobre el dominio del problema y el sistema actual.	33
2.2.1.2 Preparar y realizar las reuniones de elicitación/negociación.....	34
Técnicas	35
Principales problemas para la captura de requisitos	38
2.2.1.3 Identificar los objetivos del sistema.....	39
2.2.1.4 Identificar los requisitos (de almacenamiento de información, funcional, no funcional).Crear catálogo de requisitos.....	40
2.2.1.5 Priorizar requisitos	42
2.2.1.6 Realizar matriz de trazabilidad	42
2.2.2 Análisis de requisitos	43
2.2.2.1 Analizar requisitos de almacenamiento información	45
2.2.2.2 Analizar requisitos funcionales.....	45
2.2.2.3 Analizar requisitos no funcionales.....	53
2.2.2.4 Desarrollar prototipos	54
2.2.2.5 Priorizar casos de uso.....	54
2.2.3 Validación de requisitos	55

2.2.3.1 Validar el catálogo de requisitos	55
Técnicas para la validación de requisitos	56
2.2.3.2 Cerrar la versión de requisitos	56
2.2.4 Gestión de cambios	57
2.2.4.1 Manejar los cambios realizados a los requisitos.	57
2.2.4.2 Creación de la línea base.....	58
2.3 Documento de Requisitos del Sistema (DRS)	59
2.3.1 Estructura del DRS	60
2.4 Curso para la capacitación de los estudiantes en la disciplina Ingeniería de Requisitos.	64
2.4.1 Objetivos Generales.....	64
2.4.2 Objetivos Educativos:.....	65
2.4.3 Metodología	65
2.4.4 Descripción de los temas	65
Bibliografía para el curso	67
Capítulo III: Evaluación de la propuesta.	70
3.1 Método Delphi. Características	70
3.1.1 Pasos para la realización del Método Delphi	71
3.2 Evaluación de la propuesta	73
3.2.1 Elaboración del cuestionario.....	73
3.2.2 Resultados de las encuestas	74
CONCLUSIONES.....	78
RECOMENDACIONES.....	79
REFERENCIAS BIBLIOGRÁFICAS	80
BIBLIOGRAFÍA.....	82
GLOSARIO DE TÉRMINOS.....	85
ANEXOS.....	88

ÍNDICE DE FIGURAS

Capítulo I

Figura 1. 1 Resultados del Informe del GAO.....	8
Figura 1. 2 Resultados del Informe CHAOS.....	9
Figura 1. 3: Modelo de actividades de Ingeniería de requisitos de Pohl.....	23
Figura 1. 4: Modelo espiral	24

Capítulo II

Figura 2. 1 Relación entre roles y actividades para la IR	29
Figura 2. 2: Diagrama de actividades para la Ingeniería de requisitos	30
Figura 2. 3: Relación de actividades para el rol de analista de sistema.	32
Figura 2. 4: Roles y actividades en el análisis de requisitos.....	44
Figura 2. 5: Diagramas de casos de uso.....	53
Figura 2. 6: Documento de Requisitos del Sistema	61

ÍNDICE DE TABLAS

Tabla 1: Plantilla para la descripción de los requisitos	41
Tabla 2: Matriz de trazabilidad	43
Tabla 3: Plantilla para actores de sistema.	47
Tabla 4: Plantilla para la descripción detallada de casos de uso.....	50
Tabla 5: Problemas encontrados por los expertos	75
Tabla 6: Técnicas evaluadas por expertos	76

INTRODUCCIÓN

Debido a la revolución educacional en la que estamos inmersos, las universidades cubanas se ven en la obligación de formar a grandes profesionales de una forma integral; la Universidad de las Ciencias Informáticas (UCI) no es la excepción de lo anteriormente planteado.

Muchos son los procesos productivos que se llevan a cabo en nuestra Universidad, teniendo en cuenta que la misma es un centro donde se vinculan directamente la docencia, la producción y la investigación. La producción de software en la UCI es de gran importancia, no solo para el prestigio de la universidad en el mundo de la computación, sino también para el reconocimiento de Cuba a nivel mundial en la rama de la informática, además de la necesidad de lograr grandes aportes económicos que ayuden a mejorar la situación económica de nuestro país con estos proyectos productivos.

Con el fin de lograr una organización en el proceso de desarrollo del software, se ha tomado como estándar la aplicación de técnicas para la Ingeniería de software, las cuales incluye diferentes flujos de trabajo.

Se hace necesario mencionar la utilización por los estudiantes de la UCI de metodologías de desarrollo de software que les permite realizar la disciplina Levantamiento de Requisitos, aunque hasta el momento, no se ha definido oficialmente un método para la Ingeniería de Requisitos. Esto ha traído como consecuencias, divergencias a la hora de realizar este flujo de trabajo en los proyectos productivos. Teniendo en cuenta que esta es una etapa de gran importancia para que el producto sea realizado con calidad donde se define la visión, los objetivos y el alcance del proyecto, tanto desde el punto de vista funcional como del técnico, se hace necesario establecer una correcta organización para este flujo de trabajo inicial, que de no

ser realizado correctamente, conlleva a que el software, una vez terminado, no tenga la calidad suficiente y que no cumpla con las condiciones que necesita el usuario.

Analizando el planteamiento anterior la situación problémica que vemos en la UCI es la siguiente:

Debido a la cantidad de proyectos productivos que hay en la UCI, muchos de los cuales con alto grado de complejidad y el gran número de estudiantes que están involucrados en cada uno de ellos, los especialistas se demoran mucho en el Flujo de Trabajo de Levantamiento de Requisitos, existiendo en muchas ocasiones discrepancias entre los entendidos que participan en esta disciplina. Además, los requisitos que se obtienen no reflejan las necesidades de los usuarios, no tienen la calidad deseada y no existe control sobre los cambios de los mismos cuando éstos han sido realizados en otra etapa de desarrollo del proyecto.

Nos encontramos entonces frente al problema científico de cómo mejorar la gestión de requisitos en los productos de software que se realizan en la Universidad de las Ciencias Informáticas para así asegurar la eficiencia y eficacia de los productos informáticos que se concluyan en este centro universitario.

Por tanto el objetivo de este trabajo es proponer un proceso para la aplicación de la Ingeniería de Requisitos en los proyectos productivos de la UCI.

Los Objetivos Específicos que se derivan de este trabajo son:

1. Investigar acerca de cómo se maneja la Ingeniería de Requisitos en el mundo.
2. Caracterizar la Ingeniería de Requisitos.
3. Investigar como se maneja la Ingeniería de Requisitos en los proyectos productivos de la UCI.

Propuesta de realización de la Ingeniería de requisitos en los proyectos de la Universidad de las Ciencias Informáticas (UCI). 2

Donde el campo de acción que abarca este trabajo son los proyectos productivos de la Universidad de las Ciencias Informáticas.

Como tareas de la investigación:

- Estudio de los principales términos y conceptos de la Ingeniería de software.
- Análisis del Proceso de desarrollo del Software, a partir de cada una de sus fases.
- Investigación sobre las metodologías existentes en un proyecto de desarrollo de software.
- Estudio profundo de las actividades que comprenden la disciplina Levantamiento de Requisitos.
- Indagación sobre alguna propuesta de procedimiento existente hasta el momento, para el control de la calidad en Cuba u otro país interesado en el tema
- Implantación de un procedimiento para la Ingeniería de Requisitos.

Posibles resultados: Obtener una propuesta de prácticas para mejorar la Ingeniería de Requisitos en los proyectos productivos de la UCI.

Este trabajo se compone por tres capítulos. El *Capítulo I* plasma el estado del arte del tema al cual se le realiza el estudio (Ingeniería de requisitos). En él se encuentra los conceptos fundamentales de esta disciplina, definición y características del requisito y se describen algunas herramientas utilizadas para la obtención de este flujo de trabajo.

En el *Capítulo II* se plantea una propuesta cuyo objetivo fundamental es mejorar el desarrollo de la disciplina de Ingeniería de requisitos. El mismo cuenta con subsecciones las cuales están dedicadas al estudio de la situación en que se encuentran los proyectos de producción de la UCI en esta esfera y se brinda un solución para cumplir con el objetivo general de este trabajo.

Propuesta de realización de la Ingeniería de requisitos en los proyectos de la Universidad de las Ciencias Informáticas (UCI). 3

El *capítulo III* plasma la evaluación de la propuesta que se brinda en el capítulo anterior, la cual es valorada por un panel de expertos seleccionados por sus conocimientos en el tema que se estudia.

Capítulo I: Estado del arte.

1.1 ¿Por qué surge la Ingeniería de software?

En los primeros días de la informática, los sistemas basados en computadora se desarrollaban usando técnicas de gestión orientadas al hardware. Los administradores de los proyectos se centraban en el hardware, debido a que era el factor principal del presupuesto en el desarrollo del sistema. Para controlar los costos del hardware, los administradores instituyeron controles formales y estándares técnicos, exigían un análisis y diseño completo antes de que algo se construyera y medían el proceso para determinar dónde podían hacerse mejoras. Dicho sencillamente, aplicaban los controles, métodos y herramientas que reconocemos como ingeniería del hardware. Desgraciadamente, el software normalmente no era más que un agregado. Estas particularidades hacían que el hardware fuera mucho más caro que el software, se ponía de manifiesto la superioridad en cuanto a complejidad del hardware sobre el software y el hardware que se construía en aquellos tiempos daba poca fiabilidad a los desarrolladores, debido a que la tecnología que se utilizaba para su implementación, podría sufrir alguna avería mientras era utilizada.

En la primera era, la programación se veía como un "arte": existían pocos métodos formales y pocas personas los usaban; el programador aprendía normalmente su "oficio" mediante ensayo y error. La jerga y los desafíos de la construcción del software crearon un "orden" en la que pocos ejecutivos se preocuparon por entrar. El mundo del software era virtualmente indisciplinado.

Los avances en microelectrónica dieron como resultado una mayor potencia de cálculo y aparecieron reducciones de los costos de desarrollo y mantenimiento del hardware, así como, aumenta la capacidad de memoria. Esto a su vez provoca, la demanda de software más

complejos, provocando una gran cantidad de problemas en el desarrollo de los mismos y comienza a verse en los proyectos de software un alto riesgo de fracaso.

El principal desafío sería mejorar la calidad y reducir el costo de las soluciones basadas en computadoras, soluciones que se implementan con el software, por lo que se hacía necesario cambiar la filosofía de construcción del mismo, teniendo en cuenta para ello la utilización de métodos que le dieran más calidad a el producto que se quería obtener. Es por esto que aparecieron los siguientes cuestionamientos:

- ¿Por qué lleva tanto tiempo terminar los programas?
- ¿Por qué es tan elevado el costo?
- ¿Por qué no se pueden encontrar todos los errores antes de entregar el software?
- ¿Por qué resulta difícil constatar el progreso conforme se desarrolla el software?

Estas y muchas otras preguntas son una manifestación del carácter del software y de la forma en que se desarrolla; un problema que ha llevado a la adopción de la *ingeniería de software* como práctica.

La Ingeniería del Software, término utilizado por primera vez por Fritz Bauer en la primera conferencia sobre desarrollo de software patrocinada por el Comité de Ciencia de la OTAN celebrada en Garmisch, Alemania, en octubre de 1968, puede definirse según Alan Davis como "*la aplicación inteligente de principios probados, técnicas, lenguajes y herramientas para la creación y mantenimiento, dentro de un coste razonable, de software que satisfaga las necesidades de los usuarios*".

Todas las definiciones de ingeniería de software coinciden en que ésta se basa en la construcción de sistemas de software mediante grupos o equipos de trabajo, utilizando principios de ingeniería en el desarrollo de estos sistemas, incluyendo tanto los aspectos técnicos como los no técnicos (el ingeniero de software, aparte del conocimiento sobre técnicas de computación, debe ser

capaz de comunicar en forma escrita y oral, conocer la importancia de la administración de proyectos y apreciar los problemas que los usuarios del sistema pueden tener al interactuar con el software cuyo funcionamiento pueden no entender).

Lo anteriormente mencionado justifica la necesidad de una ingeniería de software, la cual abarca un campo muy amplio. Por esta razón se hace necesario dividirla en otras disciplinas las cuales ayudan a la mejor realización de un producto con la calidad que requiere el usuario. Entre estas se encuentra la *ingeniería de requisitos* de la cual haremos referencia en el próximo apéndice.

1.2 Ingeniería de requisitos

La necesidad de aplicar los principios de otras ingenierías al desarrollo de software está plenamente justificada por el alto grado de fracasos en los procesos de desarrollo de software.

Desde 1968 se ha invertido un gran esfuerzo en determinar las causas y proponer soluciones para la crisis del software. En 1979, la oficina de Cuentas del gobierno norteamericano (GAO), realizó un estudio a 9 proyectos de desarrollo de software para el gobierno norteamericano cuyos contratos sumaban una cantidad total 6.800.000 dólares.

De esta cantidad, solo 119.000 dólares correspondían a un proyecto que se había utilizado tal y como se había integrado. Dicho proyecto se trataba sobre un procesador de COBOL, por lo que era un problema relativamente simple cuyos requisitos eran comprendidos por clientes y desarrolladores y que además no cambiaron durante el desarrollo.

La **figura1.1** refleja los resultados de dicha encuesta.

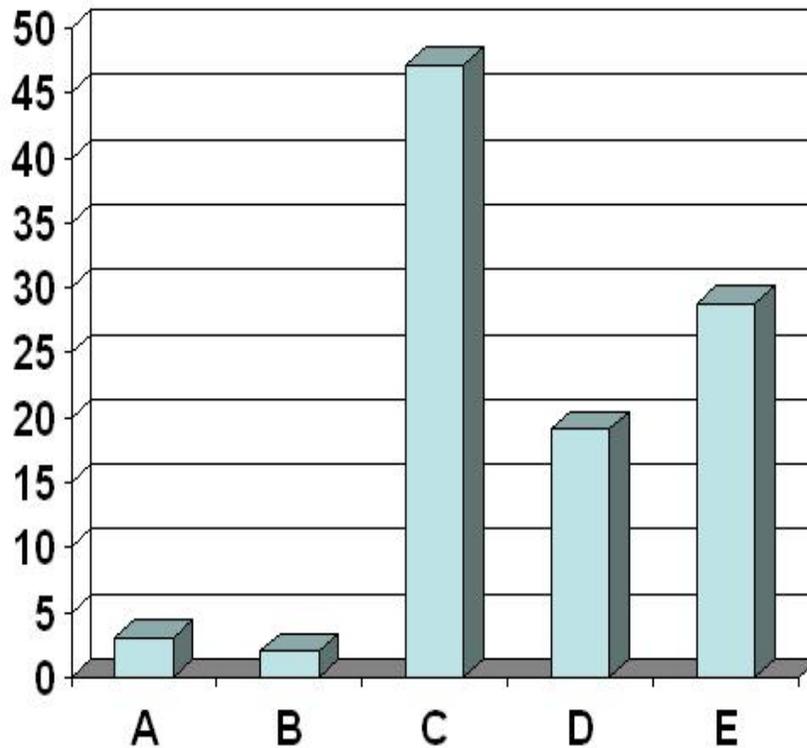


Figura 1. 1 Resultados del Informe del GAO

A: Usado después de cambios (3%)

B: Usado tal como se entregó (2%)

C: Entregado pero nunca usado (47.1%)

D: Usado pero ampliamente reelaborado o abandonado después. (19.1%)

E: Pagado pero no entregado. (28.7%)

En 1995, el Grupo Standish realiza un estudio (el informe CHAOS), mucho más amplio y significativo que el de GAO, cuyos resultados, a pesar de haber pasado más de 25 años, no reflejaban una mejoría sustancial. Ver figura 1.2.

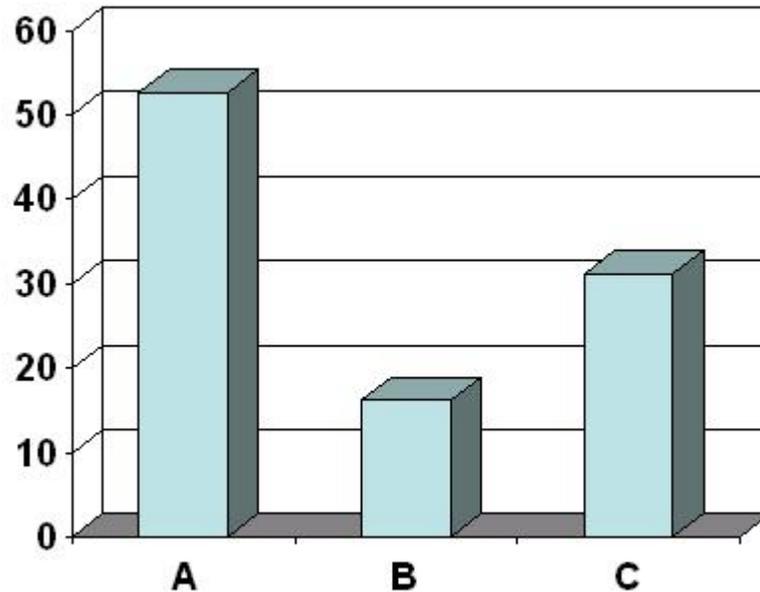


Figura 1. 2 Resultados del Informe CHAOS

A: Terminado y operativo pero fuera de plazo, presupuesto y sin satisfacer todos los requisitos. (52.7%)

B: Terminado dentro del plazo y presupuesto cumpliendo todos los requisitos. (16.2%)

C: Cancelado durante el desarrollo (31.1%)

Sin incluir el 16.2% de los proyectos terminados correctamente, la media del gasto final fue del 189% del presupuesto original, el tiempo necesario para su realización del 222% del plazo original y se cumplieron una media del 61% de los requisitos iniciales, cifras que también empeoraban en caso de grandes compañías.

En las encuestas realizadas a los directivos de los proyectos que participaron en el estudio, éstos indicaron que en su opinión los principales factores de éxito eran:

- Involucración de los usuarios
- Apoyo de los directivos
- Enunciado claro de los requisitos

Como factores de fracaso identificaron:

- Falta de información por parte de los usuarios.
- Especificaciones y requisitos incompletos
- Especificaciones y requisitos cambiantes.

Los resultados de estos informes muestran que casi un tercio de los proyectos de desarrollo de software se cancelan durante su desarrollo y que la gran mayoría presenta graves desviaciones respecto a plazos y presupuestos iniciales.

Además, las principales causas, tanto de éxito como de fracaso, indicadas por los directivos encuestados en los informes CHAOS, están relacionadas directamente con la correcta identificación y gestión de los requisitos que debían cumplir las aplicaciones desarrolladas para satisfacer las necesidades de los usuarios.

En 1996, el proyecto ESPITI (European Software Process Improvement training Initiative) realiza una investigación sobre los problemas en el desarrollo del software a nivel de Europa. Los resultados indicaron que los mayores problemas estaban también relacionados con las especificaciones y la ingeniería y gestión de los requisitos.

Estos informes ponen de manifiesto el hecho de que a pesar de que las herramientas de desarrollo de software han evolucionado a través del tiempo, se sigue produciendo software que no es satisfactorio para los clientes y usuarios, influyendo en estos resultados la primera etapa de desarrollo: cuando hay que definir las características del producto a desarrollar.

Otro hecho comprobado es que el coste de un cambio en los requisitos, una vez entregado el producto, es entre 60 y 100 veces superior al coste que hubiera representado el mismo cambio durante las fases iniciales de desarrollo, por lo que no es de extrañar que los proyectos en los

que no se determinan correctamente los requisitos y cambian frecuentemente durante el desarrollo, superan con creces su presupuesto iniciales.

Estos resultados dejan claro que para que la Ingeniería del Software alcance su madurez es necesario que aborde los aspectos sociales de la implantación del software en las organizaciones y, por extensión, en la sociedad del conocimiento.

1.3 Importancia de la ingeniería de requisitos

El éxito de un proyecto de desarrollo de una aplicación informática, depende en gran medida de la definición efectiva y del seguimiento, a lo largo de su ciclo de vida, de los requisitos que definen la funcionalidad, restricciones y características particulares del futuro sistema.

Una vez asumida la necesidad de una correcta ingeniería de requisitos, el reto al que se enfrenta la comunidad investigadora es identificar claramente los problemas a los que se enfrenta y dotar a dicha ingeniería de las herramientas teóricas y prácticas necesarias para solucionarlos, teniendo en cuenta la dificultad añadida y lo costoso de la obtención de datos empíricos tanto en ingeniería de software en general, como en ingeniería de requisitos en particular.

1.3.1 Definición de ingeniería de requisitos.

La ingeniería de requisitos, tradicionalmente entendida como una parte borrosa del ciclo de vida del software es una disciplina, que se encuentra dentro de la ingeniería de software. Su base se enmarca en la obtención de especificaciones formales de ideas informales. Entre las principales definiciones se encuentran:

Ingeniería de requisitos (1): Aplicación disciplinada de principios científicos y técnicas para desarrollar, comunicar y gestionar requisitos.

Ingeniería de requisitos (2): El proceso sistemáticos de desarrollar requisitos mediante un proceso iterativo y cooperativo de analizar el problema, documentar las observaciones resultantes en varios formatos de representación y comprobar la precisión del conocimiento obtenido. (Christel y Kang 1992).

Ingeniería de requisitos (3): Todas las actividades relacionadas con

(a) identificación y documentación de las necesidades del cliente y usuarios.

(b) creación de un documento que describe la conducta externa y las restricciones asociadas [de un sistema] que satisfará dichas necesidades.

(c) análisis y validación del documento de requisitos para asegurar consistencia, completión y viabilidad

(d) evolución de las necesidades.(Durán 2000)

Ingeniería de requisitos (4) (a) el proceso de estudiar las necesidades del usuario para llegar a una definición de requisitos de sistema, hardware o software.

(b) el proceso de estudiar y refinar los requisitos de sistema, hardware o software. [IEEE 1990].

Se hace evidente, analizando estos conceptos, que la base para la ingeniería de requisitos es el descubrimiento y comunicación de las necesidades del cliente y usuarios, así como la gestión de los cambios en dichas necesidades.

La Ingeniería de Requisitos, es el proceso encargado de la identificación, asignación y seguimiento de los requisitos, incluyendo el interfaz, verificación, modificación y control del estatus a lo largo del ciclo de vida. Es el conjunto de actividades que se concentra en el aseguramiento de las especificaciones, por ejemplo, los requisitos que son reunidos para la satisfacción del cliente. Es el proceso que inicia con la concepción de un proyecto y continúa hasta el resultado final del producto(s), después, ya no es necesario.

Debido a que un proyecto informático es susceptible de cambios, habría que proceder a su actualización o a la incorporación de nuevas funcionalidades o eliminar otras, esto obliga a mantener controlado y documentado el producto. Los cambios de requisitos deben ser gestionados para asegurar que la calidad de los mismos se mantenga, los problemas suscitados por los cambios de requisitos podrían incurrir en altos costos, siendo el requisito factor crítico de riesgo.

El proceso del ciclo de vida de la Ingeniería de Requisitos, debe ser flexible y adaptable para reunir las necesidades del proyecto. Las características del alcance e implementación del proceso del ciclo de vida de la Ingeniería de Requisitos en un proyecto, variará dependiendo de algunos factores claves:

- Tamaño y complejidad del proyecto
- Experiencia del personal del proyecto
- Experiencia de los clientes del proyecto
- Dominio de la aplicación
- El propósito y uso de esta aplicación.

Teniendo en cuenta la importancia de Ingeniería de requisitos, se considera que el aspecto más importante es la comunicación, la cual hace de la Ingeniería de requisitos una especialidad muy compleja, teniendo como base el factor humano.

Este factor es el responsable de que la ingeniería de requisitos tenga aspectos sociales y culturales y no solos técnicos .

1.3.2 Dimensiones de la ingeniería de requisitos

Una posible visión de la ingeniería de requisitos es considerada como un proceso de construcción de una especificación de requisitos en el que se avanza desde especificaciones iniciales, que no posee las propiedades oportunas, hasta especificaciones finales completas, formales y acordadas

entre todos los participantes. Por un lado están los factores psicológicos y cognitivos que afectan al grado de complejidad del conocimiento sobre el sistema que se desea desarrollar, es decir, el llegar a conocer la totalidad de los requisitos que debe satisfacer el sistema. (Pohl 1997).

Por otro lado está el grado de formalismo de la representación del conocimiento sobre dicho requisitos, teniendo en cuenta que un mayor grado de formalismo no implica necesariamente un mayor conocimiento.

Otro aspecto a tener en cuenta es el social, ya que al ser un proceso en el que participan personas con diferentes puntos de vista, es necesario llegar a un punto de acuerdo, normalmente mediante algún tipo de negociación.

1.4 Concepto de requisito. Características.

Una de las características de la Ingeniería de requisitos es la falta de uniformidad en la terminología empleada, tanto para los conceptos básicos como para los procesos y los productos.

A continuación se muestra un grupo de conceptos:

- requisito (1): Característica del sistema que es una condición para su aceptación.
- requisito (2): Propiedad que un sistema debe cumplir para tener éxito en el entorno que se usará.

A pesar de esta aparente simplicidad del concepto, muchas son los adjetivos que pueden acompañar a este término: de sistema, hardware, software, de usuario, de cliente, funcional, no funcional, etc.

1.4.1 Dimensiones de los requisitos.

Los requisitos pueden abarcar un amplio espacio. Es por eso que son divididos en tres grupos:

1. Característica que define: esta dimensión clasifica los requisitos en función de la naturaleza de la característica deseada que se especifica. Las clasificaciones más utilizadas

suelen ser la de requisitos funcionales (que funciones debe realizar el sistema) y no funcionales (otras características del sistema). En esta dimensión también aparece requisitos de información. A continuación se caracterizan los dos grupos más frecuentemente.

- **Requisitos funcionales:** especifican acciones que un sistema debe realizar, sin restricciones físicas de consideración. Los requisitos funcionales especifican la energía de entrada y el comportamiento de salida de un sistema.
- **Requisitos no funcionales:** describen sólo atributos del sistema o los atributos del ambiente del mismo.

2. **Audiencia:** esta dimensión indica la audiencia a la que esta dirigido el requisito (las personas que deben ser capaces de entenderlo). Podemos definir entre este grupo a dos tipos de audiencia: clientes y usuarios y también a los desarrolladores.

3. **Ámbito:** indica en que ámbito se debe entender el requisito. Se definen dentro de esta dimensión a los requisitos de software, hardware y sistema.

Otra de las principales características de los requisitos es que pueden interpretarse como restricciones que determinan el espacio de sistemas válidos.

1.4.2 Propiedades de los requisitos

- **Comprensible por clientes y usuarios:** la propiedad más fundamental de una especificación es que sirva como canal de comunicación entre los participantes en el proceso de ingeniería de requisitos. Se hace necesario entonces pensar en la audiencia a la que esta dirigidos los requisitos.
- **Correcta:** una especificación de requisitos es correcta si y solo si todo requisito contenido en ella representa alguna propiedad requerida por el sistema a desarrollar, es decir, ningún requisito debe ser utilizado innecesariamente ya que implicaría la reducción del espacio del sistema válido.

- No ambigua: una especificación de requisitos no es ambigua si y solo si todo requisito contenido en ella tiene una sola interpretación. Es importante señalar que para evitar interpretaciones erróneas, es necesario colocar en el glosario de términos, aquellos que puedan tener más de una interpretación y así quede aclarado su significado en este documento.
- Completa: Una especificación de requisitos es completa si cumple las siguientes propiedades:
 1. No faltan requisitos (en la especificación está todo lo que debe cumplir el sistema).
 2. Está especificada toda posible conducta del sistema desde un punto de vista externo o de caja negra. (Todas las respuestas del sistema a entradas tanto válidas como inválidas están especificadas).La especificación está organizada, los requisitos son fáciles de localizar y el documento es sintácticamente correcto. (IEEE 1993)
- Consistente: una especificación de requisitos es consistente si y solo si todo requisito contenido en ella no está en conflicto con otros documentos de nivel superior. Se definen como principales conflictos:
 1. Conflictos de conducta: dos o más requisitos especifican conductas distintas del sistema para las mismas condiciones y el mismo estímulo externo.
 2. Conflictos de términos: se utilizan términos distintos para referirse al mismo término.
 3. Conflictos de característica: dos o más requisitos especifican aspectos contradictorios para la misma característica del sistema.
 4. Conflictos temporales: dos o más requisitos exigen características temporales contradictorias al sistema.
- Verificable: una especificación de requisitos es verificable si y solo si todo el requisito contenido en ella es verificable (existe un proceso finito y de coste razonable). (IEEE 1993)

- Modificable: una especificación de requisito es modificable si y solo si su estructura y estilo de redacción permite que los cambios se puedan realizar fácilmente, completa y consistentemente.
- Rastreable: una especificación de requisito es rastreable si y solo si para cada requisito contenido en ella se conoce su origen y puede reverenciarse como origen en posteriores documentos durante el desarrollo, es decir, cada requisito puede rastrearse hacia atrás y hacia delante. (IEEE 1993)
- Anotada con importancia y estabilidad: una especificación de requisito está anotada con importancia y estabilidad si y solo si cada requisito contenido en ella está anotado con la importancia que tiene su cumplimiento para los clientes y usuarios y la estabilidad que se espera de los requisitos (probabilidad de que cambie en el desarrollo).
- Independiente del diseño y la implementación: una especificación de requisitos es independiente del diseño y de la implementación si y solo si no especifica una determinada descomposición del sistema (arquitectura) ningún aspecto de su posible implementación. (Duran 2002)

1.4.3 Otras características de los requisitos

Los requisitos, tienen otras características que los describen con mayor exactitud. Para ello se necesita comprender las condiciones que el sistema debe cumplir. Entre estas encontramos:

- La funcionabilidad: Los requisitos funcionales pueden incluir las capacidades del sistema así como la seguridad del mismo.
- La usabilidad: Los requisitos de usabilidad pueden incluir subcategorías como: los factores humanos, la estética, la consistencia en la interfaz de usuario, la documentación del usuario y los materiales de enseñanza.

- La fiabilidad: Los requisitos de fiabilidad a considerarse son: la frecuencia y la severidad de fracaso, la pronosticabilidad, la exactitud y el tiempo medio entre el fracaso (el tiempo medio entre fallos)
- La actuación: Un requisito de actuación impone condiciones en los requisitos funcionales. Por ejemplo, para una acción dada, puede especificar parámetros de actuación para la velocidad, eficiencia, disponibilidad, exactitud, rendimiento específico, tiempo de respuesta, tiempo de recuperación y uso del recurso.
- La capacidad para soportar: Los requisitos de capacidad para soportar pueden incluir: comprobabilidad, extensibilidad, adaptabilidad, compatibilidad, configurabilidad, funcionabilidad, inestabilidad y localizabilidad.
- El Requisito De Diseño: Un requisito de diseño, a menudo designado como una restricción del diseño, especifica el diseño de un sistema.
- El Requisito De Implementación: Un requisito de implementación especifica o constituye la codificación o construcción de un sistema. Los ejemplos son: normas requeridas, lenguajes de implementación, políticos para la integridad de la base de datos, límites del recurso y ambientes de operación
- El Requisito De La Interfaz: Un requisito de la interfaz específica: un artículo externo con el cual un sistema debe interactuar, las restricciones en formatos, las oportunidades del momento, u otros factores usaron por tal interacción
- El Requisito Físico: Un requisito físico especifica una característica física que un sistema debe poseer; Por ejemplo: *material, forma, tamaño y peso*. Este tipo de requisito puede usarse para representar requisitos del hardware, como las configuraciones de la red físicas requeridas.

- Requisitos de comunicaciones del sistema: Son requisitos de carácter técnico relativos a las comunicaciones que deberá soportar el sistema software a desarrollar. Por ejemplo: *el sistema deberá utilizar el protocolo TCP/IP para las comunicaciones con otros sistemas.*
- Requisitos de fiabilidad: Los requisitos de fiabilidad deben establecer los factores que se requieren para la fiabilidad del software en tiempo de explotación. La fiabilidad mide la probabilidad del sistema de producir una respuesta satisfactoria a las demandas del usuario. Por ejemplo: *la tasa de fallos del sistema no podrá ser superior a 2 fallos por semana.*
- Requisitos de entorno de desarrollo: Este tipo de requisitos especifican si el sistema debe desarrollarse con un producto específico. Por ejemplo: *el sistema deberá desarrollarse con SQL Server como servidor y clientes Visual Basic 4.*
- Requisitos de portabilidad: Los requisitos de portabilidad definen qué características deberá tener el software para que sea fácil utilizarlo en otra máquina o bajo otro sistema operativo. Por ejemplo: *el sistema deberá funcionar en los sistemas operativos Windows 95, Windows 98 y Windows NT 4.0, siendo además posible el acceso al sistema a través de Internet usando cualquier navegador compatible con HTML 3.0.*

1.5 Herramientas para la Ingeniería de requisitos

Las siguientes características básicas son necesarias para que una herramienta sea considerada como una herramienta de Gestión de Requisitos:

- identificación de requisitos "individuales"
- asignación a un destino y clasificación de requisitos
- grupo de requerimientos (recopilación), revisión, identificación / punto de arranque
- proveer un interfaz de datos básicos.

Una herramienta de Gestión de Requisitos puede soportar la disciplina de ingeniería y gestión de requisitos, así mismo, una herramienta debe poder coleccionar y gestionar los requisitos técnicos y programáticos. Funciones comunes realizadas por la herramienta, consisten en la identificación de requisitos, revisión y edición, rastreo de requisitos a su origen y generación de informes.(Landazuri 2004)

Las herramientas comerciales para la gestión de requisitos más conocidas son:

DOORS: herramienta de gestión de requisitos que se caracteriza por maximizar el valor de las iniciativas de optimización de procesos de negocio y aumentar la calidad de los proyectos de desarrollo de software e ingeniería de sistemas mejorando con esto la comunicación de los requisitos y la colaboración.

DOORS, creado principalmente para las organizaciones y proyectos que trabajan en el mismo sitio geográfico, mejora la calidad aumentando la visibilidad de los objetivos de negocio, necesidades del cliente, especificaciones técnicas y regulaciones.

Mediante sus potentes capacidades para capturar, vincular, analizar y gestionar los cambios en los requisitos y su trazabilidad, este sistema multi-plataforma garantiza la conformidad con los requisitos y con las regulaciones y estándares.

A diferencia del resto de las herramientas, considera los requisitos como objetos y los documentos como módulos. Tiene pues una orientación basada en objetos

Caliber-RM: herramienta con interfaces más prácticas y amigables y está enfocada sobre todo al trabajo en web. Caliber está basado en Internet y maneja referencia de documentos, responsabilidad de usuario, trazabilidad, prioridad y estado entre otras características. Esta herramienta también es utilizada para la gestión de requisitos en software libre.

En el caso de los proyectos productivos en la UCI, la herramienta utilizada para la gestión de requisitos es la herramienta CASE.

CASE: es una filosofía que se orienta a la mejor comprensión de los modelos de empresa, sus actividades y el desarrollo de los sistemas de información. Esta filosofía involucra además el uso de programas que permiten:

- Construir los modelos que describen la empresa.
- Describir el medio en el que se realizan las actividades.
- Llevar a cabo la planificación.

Dentro de esta filosofía se encuentra RequisitePro que es una herramienta centrada en documentos, que almacena los requisitos asociándolos a documentos (aunque también permite guardarlos directamente en la base de datos).

1.6 Ingeniería de requisitos en el mundo

Sin lugar dudas la Ingeniería de requisitos tiene gran importancia dentro de la gestión de software. Por esta razón muchas de las universidades hacen insistencia en la capacitación que sus educandos deben tener sobre este tema tan primordial para la obtención de una aplicación con calidad. Ejemplos hay muchos, entre estos se destacan la Universidad Politécnica de Sevilla y La Universidad de Munich. Estos centros universitarios consideran necesarios subdividir la Ingeniería de requisitos en tres fases fundamentales: elicitación, análisis y validación de requerimientos, con el fin de lograr una mejora en las funcionalidades del producto informático que se desea obtener.

Otro ejemplo es la Universidad de Murcia, donde utilizan el Método Siren para realizar el ciclo elicitación-validación de requisitos. Este procedimiento contempla las siguientes etapas:

obtención, selección, análisis y negociación de restricciones y por último validación de los mismos.

La Universidad EAFIT de Medellín, Colombia considera necesario dividir la Ingeniería de Requisitos en etapas: la primera para obtener qué es lo que quieren clientes y usuarios, la segunda se realiza para lograr comprender esas necesidades y llevarlas a un lenguaje que sirva en el diseño de la aplicación y la última dedicada a comprobar o verificar los requisitos obtenidos.

Si analizamos los ejemplos anteriores podemos concluir, como estas universidades ubican dentro de su plan de capacitación un amplio programa para esta disciplina. Muchas de ellas dedican en su estudio semestres completos, ya que comprenden la importancia de la misma dentro del proceso de desarrollo del software.

1.6.1 Otros modelos utilizados en la Ingeniería de requisitos.

En esta sección se comentan brevemente algunos de los modelos de procesos para Ingeniería de requisitos existentes y que han servido de base para describir la propuesta que se argumenta en el siguiente capítulo.

1.6.1.1 El modelo de Pohl

El modelo de Pohl (Pohl 1997) es un modelo iterativo en el que se describe las cuatro etapas que pueden verse en la **figura 1.3**, en el que aunque el orden de las actividades puede ser cualquiera, se asume una secuencia en la que los requisitos son elicitados, a continuación son negociados entre los participantes se integran con el resto de la documentación y posteriormente se validan.

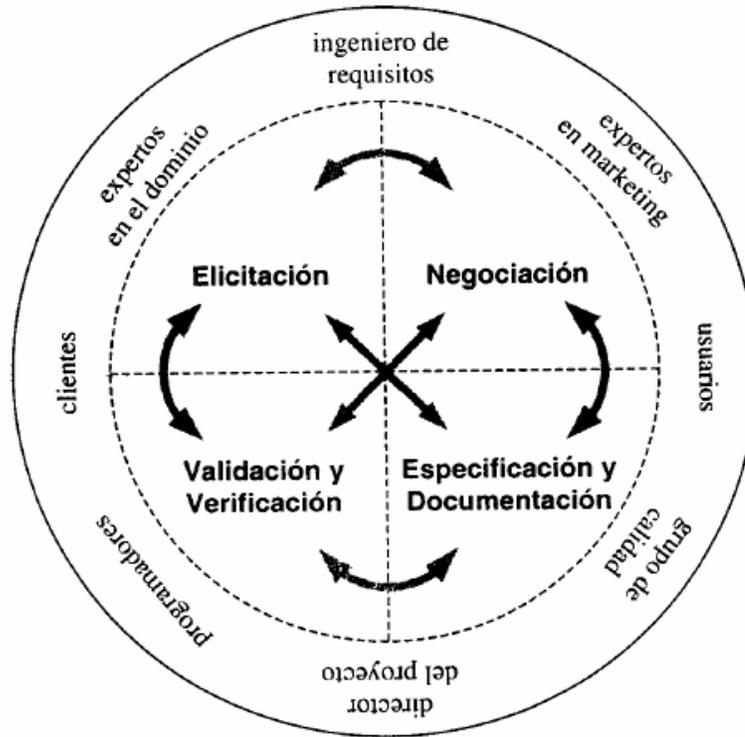


Figura 1. 3: Modelo de actividades de Ingeniería de requisitos de Pohl

1.6.1.2 El modelo espiral

El modelo espiral asume una naturaleza iterativa del proceso y la dificultad de establecer un punto de terminación del mismo dado que los requisitos nunca llegarán a ser perfectos.

Además de las actividades que se muestran en la **figura 1.4**, el proceso se adjudica una cuarta etapa dedicada a la gestión de requisitos, que se realiza durante todo el proceso y se encarga de gestionar la obtención incremental de los requisitos y los necesarios cambios a los que están sujetos.

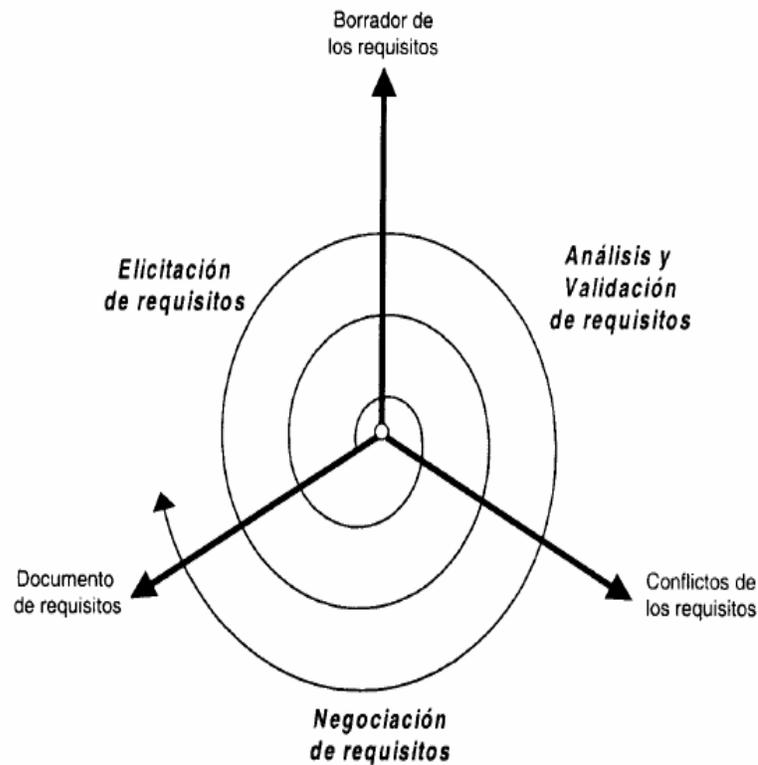


Figura 1. 4: Modelo espiral

1.7 Conclusiones

En este capítulo se le da respuesta a dos de los objetivos específicos que se propone en este trabajo: se realiza un estudio sobre los principales conceptos de la Ingeniería de requisitos y se realiza un análisis de cómo se maneja la Ingeniería de requisitos en el mundo.

Capítulo II: Solución propuesta

La estructura de este capítulo es la siguiente: Estudio de la Ingeniería de Requisitos en los proyectos de la UCI. En el epígrafe 2.2 estará conformado por la propuesta para realizar dicha ingeniería en la UCI. Dicha sección estará dividida en tres subsecciones: Captura de requisitos, Análisis de requisitos y Validación de los mismos, así como la gestión de cambios. El apartado 2.3 hace referencia a la estructura del Documento de Requisitos del sistema (DRS). En el 2.4 aparecerá una propuesta de curso para la capacitación de los estudiantes en la disciplina de Ing. de Requisitos.

2.1 Estudio de la Ingeniería de Requisitos en los proyectos de la UCI

Teniendo en cuenta que el campo de acción de este trabajo investigativo se enmarca en los proyectos productivos que se realizan en la Universidad de las Ciencias Informáticas, se hace obligatorio estudiar cómo se ha venido realizando la Ingeniería de requisitos en el centro educacional.

Se definen los siguientes objetivos:

- Identificar los principales problemas que presenta la disciplina Levantamiento de requisitos.
- Caracterizar el proceso actual de Ingeniería de requisitos en los proyectos productivos en el centro docente.
 - 🗺 Identificar técnicas para la captura de requisitos.
- Definir la importancia de la Ingeniería de requisitos para los proyectos de la UCI.

Entre los objetivos de esta encuesta se encuentra conocer los métodos o técnicas utilizadas por los analistas de los proyectos a la hora de realizar la captura de requisitos, ya que en esta

*Propuesta de realización de la Ingeniería de requisitos en los proyectos de la Universidad de las 25
Ciencias Informáticas (UCI).*

actividad se asegura la base para la obtención de requisitos con una calidad óptima. La diferencia en técnicas puede traer consigo mayor esfuerzo para los desarrolladores de la aplicación o para los clientes.

Se necesita además, identificar los principales problemas con los que se enfrentan nuestros analistas cuando realizan el levantamiento de requisitos y conocer que agentes reconocen como factores de éxitos y fracaso, con el fin de formular la propuesta teniendo en cuenta estas opiniones.

Como se explicará posteriormente, las tres etapas en las que se divide la Ingeniería de requisitos: captura, análisis y validación (incluida entre estas la gestión de requisitos), en muchas ocasiones no se realiza de la forma correcta y en ocasiones los proyectos omiten algunas de estas fases. Por esta razón se incluye en este cuestionario una pregunta para conocer si esta situación ocurre en los proyectos de la UCI. (Ver anexo 1)

Conociendo la estructura del programa de clases de la Ingeniería de software y el amplio contenido que abarca la Ingeniería de requisitos (IR), con su marcado carácter social, se considera importante conocer la opinión de los entrevistados sobre la capacitación sobre este tema y la importancia que le ven a la IR dentro de la etapa de desarrollo.

La muestra tomada para la encuesta fue de 150 personas, de ellos 25 líderes de proyectos, 50 analistas y 75 estudiantes que participan en los proyectos ejecutando diferentes roles.

Para procesar los resultados, teniendo en cuenta los conocimientos que tenían sobre el tema, se dividió la muestra en dos grupos: líder de proyecto-analista y estudiante.

Las encuestas reportan para el grupo de líder de proyecto-analista que la captura de requisitos se realizan mediante la técnica de entrevista en un 89% de los proyectos, un 3% utiliza los

cuestionarios y un 8% utiliza otras técnicas de elicitación. En el caso del segundo grupo los resultados fueron que un 57% realiza este flujo mediante la entrevista, un 4% se apoyan en los cuestionarios y un 3% utilizan otras técnicas. De los entrevistados un 36% no supo responder esta pregunta. (Ver anexo 2).

Los principales problemas que señalaron a la hora de ejecutar dichas técnicas en ambos grupos fueron la mala comunicación entre clientes, usuarios y desarrolladores (Ver anexo 3). Además reconocen como dificultad que una vez realizados los cambios en los requisitos, estos no quedan correctamente documentados y que en gran medida, los frecuentes cambios en los requisitos se deban a que los clientes reconocen sus necesidades y las posibilidades que ofrece la tecnología con el avance del tiempo y no a la hora que se realiza la entrevista. (Ver anexo 4)

Conocer si en los proyectos se desarrollan las tres etapas de la IR reportó para el primer grupo un 83% de que sí se realizan las tres etapas, un 12% de que no y un 5% no supo responder. Para el caso del segundo grupo un 28% respondió que si, un 26% que no y un 46% no supo responder. (Ver anexo 7)

Los factores de éxitos identificados para la obtención de una aplicación con calidad fueron la involucración de los usuarios, apoyo de los directivos y enunciado claro de los requisitos. Como factores de fracaso reconocieron falta de información por parte de los usuarios, especificaciones y requisitos incompletos y cambiantes.

Ambos grupos de entrevistados aseguraron que se necesita aumentar la capacitación en el tema de la Ingeniería de requisitos y le dan gran importancia a la misma. (Ver anexo 5 y 6)

2.2 Propuesta para la realización de la Ingeniería de Requisitos en los proyectos productivos en la UCI

En el proceso de creación de un sistema, el equipo de desarrollo se enfrenta al problema de la identificación de requisitos. La definición de las necesidades del sistema es un proceso complejo, pues en él hay que identificar los requisitos que el sistema debe cumplir para satisfacer las necesidades de los usuarios finales y de los clientes. Para realizar este proceso, no existe una única técnica estandarizada y estructurada que ofrezca un marco de desarrollo que garantice la calidad del resultado.

Se debe tener en cuenta que la selección de las técnicas y el éxito de los resultados que se obtengan dependen en gran medida tanto del equipo de análisis y desarrollo, como de los propios clientes o usuarios que en ella participen.

El propósito de esta disciplina es establecer y mantener un acuerdo con los clientes y usuarios que utilizarán el sistema. Además, provee a los desarrolladores de una mejor comprensión de los requisitos que se obtienen durante este flujo de trabajo. Otras de las funcionalidades de esta disciplina es la definición de los participantes en el proyecto, provee una base para planear los contenidos técnicos de las distintas iteraciones y se define una interfaz de usuario para el sistema, enfocando la atención en las necesidades y las metas de los usuarios.

A continuación se da una propuesta de cómo realizar en los proyectos productivos de la UCI la Ingeniería de requisitos, teniendo en cuenta para ello, los resultados de la encuesta que se realizó anteriormente.

Primeramente se muestran las diferentes actividades que se debe realizar para la correcta obtención de requisitos; cada una de ellas relacionadas con los diferentes roles que serán los encargados de realizar dichas actividades. (Ver **figura 2.1**)

Propuesta de realización de la Ingeniería de requisitos en los proyectos de la Universidad de las Ciencias Informáticas (UCI). 28



Figura 2. 1 Relación entre roles y actividades para la IR

Teniendo en cuenta la complejidad de la disciplina Ingeniería de requisitos; el proceso de especificación de requisitos se puede dividir en tres grandes actividades:

- 1- captura de requisitos
- 2- análisis de requisitos
- 3- validación de requisitos.

A estas actividades se le adiciona la gestión de cambios en los requisitos, la cual será realizada cada vez que exista mala calidad en los requerimientos que se vayan obteniendo durante todo el ciclo de desarrollo de la Ingeniería de requisitos.

En la **figura 2.2** se presenta el proceso de Ingeniería de requisitos que incluye estas actividades. Para la representación se ha usado la notación de diagrama de actividades.

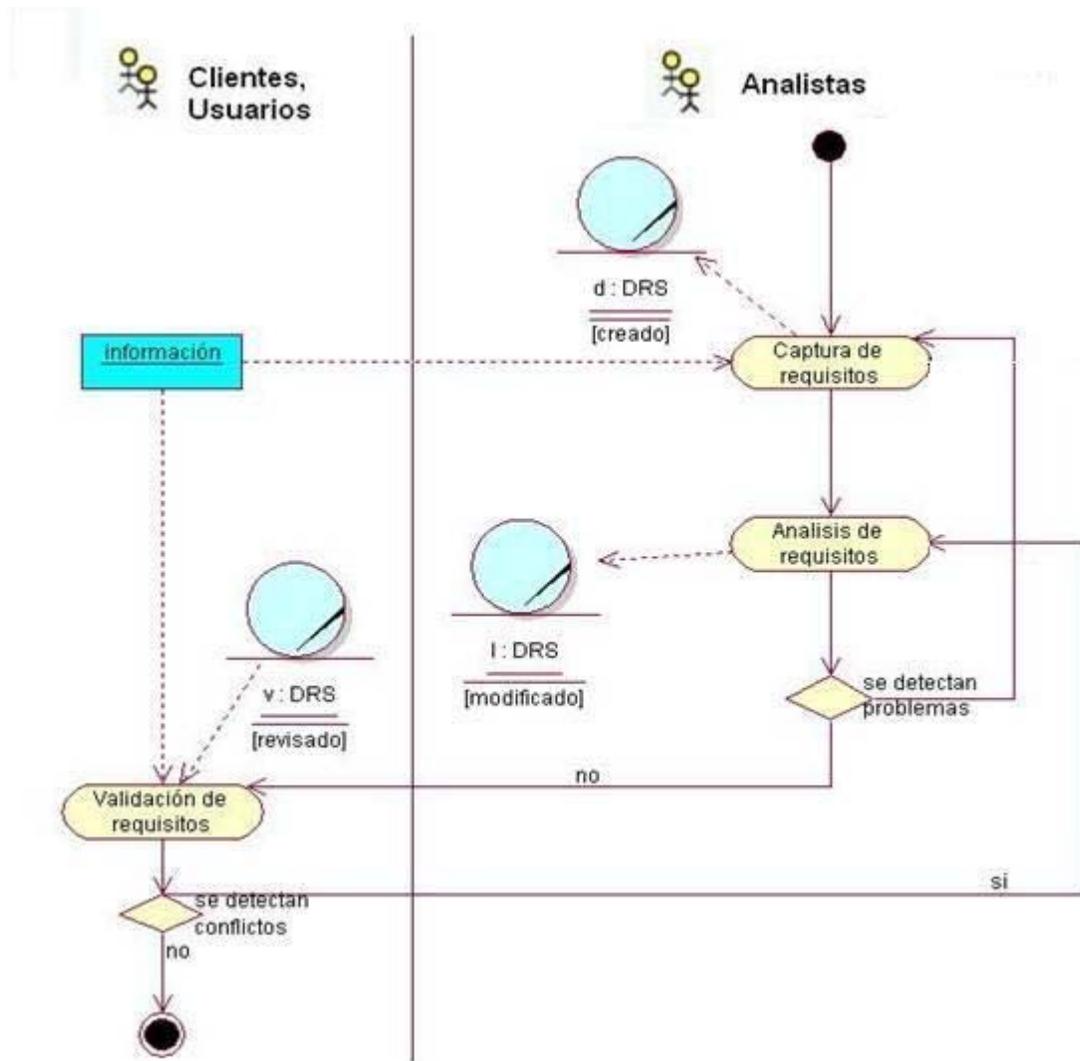


Figura 2. 2: Diagrama de actividades para la Ingeniería de requisitos

El proceso comienza con la realización de la captura de requisitos, el grupo de analistas toma la información suministrada por los usuarios y clientes. Esta información puede provenir de fuentes muy diversas: documentos, aplicaciones existentes, a través de entrevistas, etc. En base a ella, el equipo de desarrollo elabora el catálogo de requisitos, plasmándolos en el DRS. Ya en el análisis se procede al estudio intenso de los requisitos elicitados. En caso de encontrar algún

problema se analiza con el cliente para lograr eliminarlos. Finalmente, con la validación se procede a la valoración de los mismos, comprobando si existen inconsistencias, errores o si faltan requisitos por definir. El proceso de definición-validación es iterativo y en algunos proyectos complejos resulta necesario ejecutarlo varias veces. Como se ha descrito, todo este proceso está sujeto a sufrir alteraciones, las cuales podrán ser controladas mediante la gestión de cambio.

Con el fin de documentar correctamente este proceso se propone la creación del Documento de requisitos del sistema (DRS) (ver **epígrafe 2.3**).

En los siguientes apartados se presentan brevemente algunas técnicas clásicas para realizar las actividades de captura, definición, validación y gestión de cambios en los requisitos y se describe la estructura a seguir en la creación del DRS.

2.2.1 Captura de Requisitos

El objetivo principal de un sistema de información es automatizar tareas o actividades de un proceso de negocios, permitiendo a los actores organizacionales alcanzar sus metas particulares, así como las metas generales del negocio. Esta es la razón por la que el estudio del ambiente organizacional en el que se implantará el producto software ha sido reconocido como una parte fundamental de la ingeniería de requisitos.

La etapa de elicitación o captura de requisitos abarca la primera y quizás más importante fase dentro del desarrollo de un sistema informático. Uno de los retos más importantes de la elicitación de requisitos es garantizar que los requisitos del sistema sean consistentes con las necesidades de la organización donde se utilizará el mismo y con las futuras necesidades de los usuarios.

La **figura 2.3** muestra el rol involucrado en este subproceso relacionado con las actividades que realiza.



Figura 2. 3: Relación de actividades para el rol de analista de sistema.

El objetivo de este subproceso es la definición de las tareas a realizar, los productos a obtener y las técnicas a emplear durante la actividad de *captura de requisitos*.

En esta fase se comienza a elaborar el *Documento de Requisitos del Sistema*, el cual podrá ser modificado en las siguientes etapas de la ingeniería de requisitos.

Para realizar una elicitación de requisitos apropiadamente se recomienda seguir los siguientes pasos:

- Obtener información sobre el dominio del problema y el sistema actual.
- Preparar y realizar las reuniones de elicitación/negociación.
- Identificar los objetivos del sistema.
- Identificar los requisitos (de almacenamientos de información, funcionales, no funcionales).
- Priorizar requisitos.
- Realizar matriz de trazabilidad.

Se recomienda seguir el orden de los pasos según fueron escritos en este documento, teniendo en cuenta que la obtención de información sobre el dominio del problema y el sistema actual es un paso que puede o no realizarse.

Para esto se tendrá en cuenta el conocimiento previo que tenga el equipo de desarrollo sobre el dominio del problema y el sistema actual.

A continuación se describen cada uno de estos pasos.

2.2.1.1 Obtener información sobre el dominio del problema y el sistema actual.

Objetivos:

1. Conocer/estudiar el dominio del problema.
2. Conocer/estudiar la situación actual.
3. Definir un vocabulario común del dominio problemático.

Antes de mantener las reuniones con los clientes y usuarios e identificar los requisitos, es primordial conocer el dominio del problema y los contextos organizacional y operacional, es decir, la situación actual.

Enfrentarse a un desarrollo sin conocer las características principales ni el vocabulario propio de su dominio suele provocar que el producto final no sea el esperado por clientes y usuarios.

Por otro lado, mantener reuniones con clientes y usuarios sin conocer las características de su actividad hará que probablemente no se entiendan sus necesidades y que su confianza inicial hacia el desarrollo se vea deteriorada enormemente. Esta tarea es opcional, ya que puede que no sea necesario realizarla si el equipo de desarrollo tiene experiencia en el dominio del problema y el sistema actual es conocido.

El vocabulario común sobre el dominio del problema se utiliza consistentemente en todas las descripciones textuales del sistema, especialmente en descripciones de caso de uso. De este modo, se conserva las descripciones textuales coherentemente y se evita malentendidos entre miembros de proyecto acerca del uso y significado de condiciones propuestas por clientes y usuarios. Este vocabulario debe quedar documentado en el glosario de términos del DRS.

El resultado final de esta tarea es la entrega de una introducción, participantes en el proyecto, principalmente clientes y desarrolladores, y en el caso de cambios de funcionalidades de un sistema; se realizará la descripción del sistema actual. Todo esto formará parte del DRS. Para lograr este objetivo, el/los analista/s se pueden apoyar en la información recopilada: libros, artículos, folletos comerciales, desarrollos previos sobre el mismo dominio, etc. En el caso de que se trate de un dominio muy específico, puede ser necesario recurrir a fuentes internas al propio negocio del cliente, en cuyo caso se analizan las técnicas auxiliares de elicitación de requisitos como el estudio de documentación, cuestionarios, inmersión o aprendizaje, etc.

Si lo que se ejecuta es una modificación de un sistema entonces se procede al estudio de los modelos del sistema actual.

2.2.1.2 Preparar y realizar las reuniones de elicitación/negociación.

Objetivos:

- Identificar a los usuarios participantes.
- Conocer las necesidades de clientes y usuarios.
- Resolver posibles conflictos.

Teniendo en cuenta la información recopilada en el paso anterior, en esta tarea se deben preparar y realizar las reuniones con los clientes y usuarios participantes con el objetivo de obtener sus necesidades y resolver posibles conflictos que se hayan detectado en iteraciones previas del proceso.

Esta tarea es especialmente crítica y ha de realizarse con especial cuidado, ya que generalmente el equipo de desarrollo no conoce los detalles específicos de la organización para la que se va a desarrollar el sistema y, por otra parte, los clientes y posibles usuarios no saben qué necesita saber el equipo de desarrollo para llevar a cabo su labor.

Para esta iteración, ya se puede definir requisitos que quedaron claros durante la captura, mediante reuniones, entrevista o estudio el sistema actual; así como los participantes en el proyecto de forma precisa. Es muy importante para este paso las notas tomadas durante las reuniones, transcripciones o actas de reuniones, formularios, grabaciones en cinta o vídeo de las reuniones o cualquier otra documentación que se considere oportuna.

Las técnicas recomendadas (las más frecuentemente utilizadas) para conocer las necesidades de clientes y usuarios y así dejar definidos los objetivos y requisitos del sistema se describen en el siguiente apartado.

Técnicas

Entrevistas: Las entrevistas son la técnica de elicitación más utilizada, y de hecho son prácticamente inevitables en cualquier desarrollo ya que son una de las formas de comunicación más naturales entre personas. En el caso de la UCI, es la técnica más utilizada en los proyectos.

Las entrevistas no deben improvisarse. Primeramente se debe llevar a cabo un estudio del dominio del problema al cual se le quiere realizar el levantamiento de requisitos. Luego se procede a la selección de las personas que se van a entrevistar. Es importante señalar en este

punto, que las personas a entrevistar deben tener un basto conocimiento del dominio del problema, para evitar ambigüedades o la obtención de requisitos que no cumplan con las funcionalidades deseadas en el sistema. Además, si el personal está capacitado, la entrevista se hará con mayor rapidez y calidad.

Antes de fijar la fecha, hora y tiempo de duración de la entrevista, deben estar bien claros el objetivo y el contenido de la entrevista.

A la hora de realizar la entrevista, se deben evitar los monólogos y mantener el control por parte del entrevistador, contemplando la posibilidad de que una tercera persona tome notas durante la conversación. Durante esta etapa se pueden utilizar preguntas abiertas las cuales no pueden responderse con un sí o un no. Ej. "*¿Qué se hace para registrar un pedido?*", "*Dígame qué se debe hacer cuando un cliente pide una factura*" o "*¿Cómo se llena la plantilla de inscripción?*"; Estas preguntas deben estar conformadas por palabras apropiadas, fáciles de entender por el entrevistado. Es de gran importancia mantener siempre atención a cada respuesta que dé el entrevistado para evitar perder algún dato significativo.

Al terminar la reunión se debe recapitular para confirmar que no ha habido confusiones en la información recogida, agradecer al entrevistado su colaboración y citarles para un nuevo encuentro si fuera necesario, dejando siempre abierta la posibilidad de volver a contactar para aclarar dudas que surjan al estudiar la información o al contrastarla con otros entrevistados.

Una vez realizada la entrevista es necesario leer las notas tomadas, pasarlas a limpio, reorganizar la información, verificarla con otras entrevistas o fuentes de información, etc.

JAD (Joint Application Development/Desarrollo conjunto de aplicaciones): Esta técnica resulta una alternativa a las entrevistas. Es una práctica de grupo que se desarrolla durante varios días y en la que participan analistas, usuarios, administradores del sistema y clientes.

Está basada en cuatro principios fundamentales: dinámica de grupo, el uso de ayudas visuales para mejorar la comunicación, mantener un proceso organizado y racional y una filosofía de documentación WYSIWYG (What You See Is What You Get, lo que se ve es lo que se obtiene), es decir, durante la entrevista se trabajará sobre lo que se generará. Tras una fase de preparación del JAD al caso concreto, el equipo de trabajo se reúne en varias sesiones. En cada una de ellas se establecen los requisitos de alto nivel a trabajar, el ámbito del problema y la documentación. Durante la sesión se discute en grupo sobre estos temas llegando a una serie de conclusiones que se documentan. En cada sesión se van concretando más las necesidades del sistema. Esta técnica presenta una serie de ventajas frente a las entrevistas tradicionales ya que ahorra tiempo al evitar que las opiniones de los clientes se tengan que contrastar por separado. Pero requiere un grupo de participantes bien integrados y organizados.

Brainstorming (Tormenta de ideas): es también una técnica de reuniones en grupo cuyo objetivo es que los participantes muestren sus ideas de forma libre. Consiste en la mera acumulación de ideas y/o información sin evaluar las mismas. El grupo de personas que participa en estas reuniones no debe ser muy numeroso (máximo 10 personas), una de ellas debe asumir el rol de moderador de la sesión, pero sin carácter de controlador. Como técnica de captura de requisitos es sencilla de usar y de aplicar, contrariamente al JAD puesto que no requiere tanto trabajo en grupo como éste. Además suele ofrecer una visión general de las necesidades del sistema, pero normalmente no sirve para obtener detalles concretos del sistema, por lo que suele aplicarse en los primeros encuentros.

Cuestionarios y Checklists: Esta técnica requiere que el analista conozca el ámbito del problema en el que está trabajando. Consiste en redactar un documento con preguntas cuyas respuestas sean cortas y concretas, o incluso cerradas por unas cuantas opciones en el propio cuestionario (Checklist). Este cuestionario será cumplimentado por el grupo de personas entrevistadas o simplemente para recoger información en forma independiente de una entrevista.

De las técnicas mencionadas recomiendo utilizar la entrevista, ya que en ellas se contacta directamente con el cliente que desea realizar la aplicación. Siempre y cuando la entrevista esté bien elaborada, el entrevistador tenga conocimientos del dominio del problema y los entrevistados tengan noción de los privilegios que brindan las tecnologías de la información, los resultados de la entrevista serán de gran utilidad en la obtención de requisitos con calidad.

Principales problemas para la captura de requisitos

- *Expresión de sus necesidades por clientes y usuarios:* Si los usuarios y clientes analizan su situación actual de forma previa y cuales son sus necesidades se puede minimizar este problema. Es necesario precisarle cuales son las restricciones que tienen la aplicación en caso de que exista y las posibilidades que brinda un nuevo sistema. Se recomienda formalizar (en copia dura), cada una de los criterios dados.
- *Comunicación entre clientes, usuarios y desarrolladores:* es necesario lograr un entendimiento entre todos los involucrados en el proceso de captura de requisitos. Para lograrlo es necesario que los desarrolladores explique todas las ventajas que puede brindar la tecnología y realizar buenas técnicas de captura de requerimientos. Se debe definir además los diferentes roles que participaran en la captura, así como los entrevistados, que a la postren brindaran la mayor parte de las funcionalidades del sistema.
- *Los requisitos cambian con el tiempo ya que a medida que los usuarios y clientes van conociendo sus necesidades y las posibilidades que le brinda la tecnología, surge la necesidad de reconsiderar decisiones anteriores:* Al igual que el problema anterior, la base para el mejoramiento del mismo es una correcta elicitación, donde se logre comprender todas las necesidades de los usuarios y se logre transmitir el porqué de las funcionalidades de la aplicación que se desea realizar. Estar bien claros en lo que nos pide el cliente es un punto primordial para evacuar este problema ya que en la mayoría de los casos los requisitos cambian porque los clientes no reconocen (luego de la captura) lo que ellos han

señalado como restricciones. Las copias duras serian un buen avance para mejorar esta dificultad.

➤ *En problemas complejos, los requisitos tienden a ser cada vez más complejos:* la mejor forma de mejorar este conflicto es ampliando la capacitación del personal encargado de la captura y posterior análisis de los requisitos.

Es importante puntualizar las necesidades que han señalado los usuarios y clientes durante las reuniones de elicitación/negociación, ya que en ocasiones, cuando se llega a la etapa de validación de requisitos, los entrevistados no reconocen todas las necesidades que habían señalado anteriormente y en muchos casos adicionan otras nuevas, quedando en dudas la calidad del proceso elicitación-validación de los especialistas. Para ello se recomienda la creación de documentos, donde los usuarios y clientes confirmen, mediante firmas, la aceptación de los resultados que ha brindado la realización de esta actividad. Se propone el nombre de Documento de Confirmación de Clientes y Usuario (DCCU), el cual incrementará la calidad de este proceso.

2.2.1.3 Identificar los objetivos del sistema.

Objetivos:

- Identificar los objetivos que se esperan alcanzar mediante el sistema software a desarrollar.
- Revisar, en el caso de que haya conflictos, los objetivos previamente identificados.

En este paso se identifican los objetivos que se esperan alcanzar con la terminación del software, teniendo en cuenta para ello la información recopilada en el paso anterior y el estudio de la situación problemática. En caso de encontrar algún conflicto, entonces se lleva a cabo una revisión de los mismos con el fin de dejar claros los objetivos del sistema a realizar. Es

necesario analizar en este paso los factores críticos de éxito o alguna técnica similar de identificación de objetivos.

Al igual que en el paso anterior es necesario plasmar estos objetivos en el DCCU, el cual será leído y confirmado mediante firmas, por clientes y usuarios.

El DRS, en este paso aumenta ya que se le adiciona los objetivos del sistema.

2.2.1.4 Identificar los requisitos (de almacenamiento de información, funcional, no funcional). Crear catálogo de requisitos.

Objetivos:

- Identificar los actores del sistema del sistema software a desarrollar.
- Identificar los requisitos funcionales, de almacenamiento de información y no funcionales que deberá cumplir el sistema software a desarrollar.
- Revisar, en el caso de que haya conflictos, los requisitos funcionales, de almacenamiento de información y no funcionales previamente identificados.

A partir de la información obtenida en los dos primeros pasos y teniendo en cuenta los objetivos identificados en la paso 3, en esta tarea se identifica o revisa si existen conflictos. Para los requisitos de información (RI) se analiza qué información relevante para el cliente deberá gestionar y almacenar el sistema software a desarrollar. En el caso de los requisitos funcionales (RF) primero se identificarán los actores que interactúan con el sistema.

Se propone utilizar la plantilla siguiente para dejar especificados los requisitos obtenidos.

Identificador del requisito	<i><Nombre del requisito ></i>
Objetivos del sistema asociados al requisitos	<i><Identificador obj. , nombre del objetivo del sistema></i>
Descripción	<i><Según la dimensión del requisito definido: RF, RNF o RI ></i>

Importancia	<i><importancia del requisito></i>
Requisitos asociados	<i><Se indican otros requisitos que estén asociados por algún motivo con el requisito que se está describiendo.></i>
Comentario	<i><Comentarios adicionales sobre el requisito></i>

Tabla 1: Plantilla para la descripción de los requisitos

Descripción: para el caso de los RI se deberá completar con el concepto relevante sobre la información que se desea almacenar, para los RF este campo queda pendiente para la siguiente iteración ya que el llenado del mismo dependerá del caso de uso que se derive del requisito, es decir se tendrá en cuenta si el caso de uso sea abstracto o concreto; y para los RNF la descripción será la capacidad que gana el sistema con el requisito descrito.

Objetivos del sistema 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.

Importancia: este campo indica la importancia del objetivo, es decir, una estimación de la probabilidad de que pueda sufrir cambios en el futuro. Esta importancia puede indicarse mediante un valor numérico (de manera restringida) o mediante una expresión enumerada como *alta, media o baja*.

Para los requisitos no funcionales, es importante identificar los requisitos de comunicaciones del sistema, de interfaz de usuario, de fiabilidad, de entorno de desarrollo y de portabilidad, los cuales fueron explicados en el capítulo anterior.

En esta fase se proponen los actores del sistema, aunque su definición y descripción se realizará en el análisis.

Todos los artefactos que se generan durante esta actividad deben formar parte del DCCU.

2.2.1.5 Priorizar requisitos

Objetivos:

- Priorizar la realización de los requisitos funcionales.

En un proceso de desarrollo de software, es necesario crear un orden teniendo en cuenta las necesidades y urgencias funcionales que presentan usuarios y clientes sobre el sistema. Es por ello que una de las tareas que incluye la captura de requisitos, es llegar a un consenso entre analistas y las personas que necesitan el sistema en cuanto a la prioridad que tienen los requisitos dentro del período de creación o mejora de la aplicación.

2.2.1.6 Realizar matriz de trazabilidad

El principal objetivo de esta matriz es reflejar y conocer la relación que existe entre los objetivos que debe cumplir el sistema una vez esté en explotación y los requisitos que fueron hallados durante la elicitación. Para esta fase se hace un esbozo, ya que esta matriz quedará bien conformada en las fases posteriores. Se propone como plantilla la estructura de la **tabla 2**

	Obj. 1	Obj. 2	Obj. n
RI 1	•			
RI 2		•		
....				
RF 1		•		
....				
RNF 1	•			
....				

Tabla 2: Matriz de trazabilidad

Debemos apuntar que el DRS se comienza a elaborar en la captura de requisitos pero queda mejor definido en la siguiente etapa (análisis), es decir, se crea el documento teniendo en cuenta los resultados de las entrevistas o otras técnicas utilizadas durante la captura de los requisitos, dejando para la siguiente fase la definición y creación de escenarios y casos de uso.

2.2.2 Análisis de requisitos

El término análisis de requisitos se ha utilizado durante bastante tiempo para englobar todas las actividades relacionadas con los requisitos en el desarrollo del software. En este trabajo se considera el análisis de requisitos como una actividad dentro de la ingeniería de requisitos cuyo objetivo fundamental es la de descubrir los conflictos que pudieran tener los requerimientos seleccionados previamente, profundizando en el conocimiento del problema y obteniendo entonces requisitos que sean mas fáciles de entender por los desarrolladores y que puedan servir de base en la etapa del Diseño.(Johan 2006)

En esta fase se procede al análisis de los requisitos obtenidos teniendo en cuenta la información brindada por los usuarios con el fin de comprenderlos, y a partir de ellos, desarrollar una especificación de la aplicación, que deberá ser completa y consistente. En este proceso, encontraremos, habitualmente, gran cantidad de problemas en los requisitos: áreas no especificadas, requisitos contradictorios, y afirmaciones (aparentemente) vagas e irrelevantes. Eso nos llevará de vuelta a los usuarios con el fin de mejorar la calidad de los mismos: pero es preciso abordarles sabiendo lo que queremos conseguir, qué aspectos de los requisitos obtenidos inicialmente nos interesa aclarar, y el porqué.

En resumen, el objetivo de esta fase es realizar un reconocimiento de los diferentes requisitos que fueron seleccionados en la fase anterior, encontrar los conflictos que puedan existir y darle solución, de manera que éstos sean comprendidos por los desarrolladores que serán los encargados de realizar el diseño de la aplicación.

La **figura 2.4** muestra la relación entre los roles y actividades que se realizan en este subproceso.

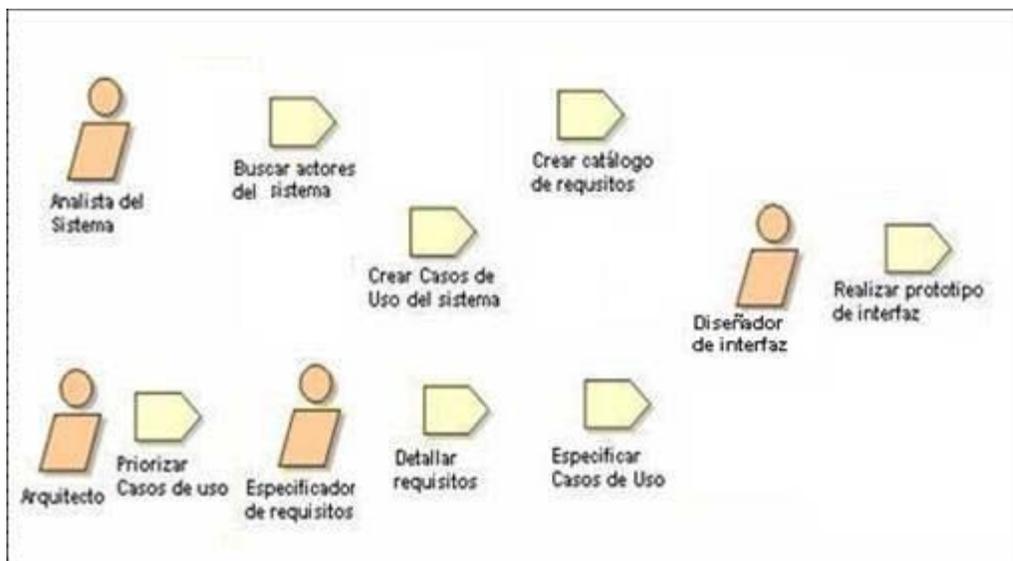


Figura 2. 4: Roles y actividades en el análisis de requisitos

Pasos recomendados

- Analizar requisitos de almacenamiento de información
- Analizar requisitos funcionales. Desarrollar el modelo de comportamiento del sistema
- Analizar requisitos no funcionales
- Desarrollar prototipos
- Priorizar casos de usos

2.2.2.1 Analizar requisitos de almacenamiento información

Objetivos:

- Descubrir posibles conflictos en los requisitos de almacenamiento de información
- Profundizar en el conocimiento de los requisitos.

En este paso se debe obtener el modelo estático del sistema a partir de los requisitos de información obtenidos en la actividad de elicitación. En caso de detectar algún conflicto en los requisitos anteriormente elicitados, se procede a intercambiar nuevamente con los clientes en busca de encontrar un acuerdo que permita utilizar de forma más eficiente este requisito por los desarrolladores.

Para este análisis se tomará como base la plantilla que fue llenada en la fase de elicitación de requisitos. Ver **tabla 1**

2.2.2.2 Analizar requisitos funcionales.

Objetivos:

- Detectar posibles conflictos en los requisitos funcionales
- Definir los actores del sistema
- Definir los casos de uso del sistema
- Realizar diagramas de casos de uso
- Profundizar en el conocimiento de los requisitos

En esta tarea se debe obtener el modelo de comportamiento del sistema a partir de los requisitos funcionales, tanto los expresados de forma tradicional como los expresados mediante casos de uso, obtenidos en la actividad de elicitación. Al igual que en el paso anterior de detectarse algún conflicto en los requisitos elicitados, se debe regresar junto al cliente, para volver a negociar el requisito y que éste quede apto para ser utilizado por el equipo de desarrollo. Otras de las acciones que se realizan son la definición de los casos de uso del sistema, confección de los diagramas de casos de uso y la definición de actores del sistema; todo esto teniendo en cuenta los datos obtenidos en la captura. (Amador 2000)

Los siguientes apartados describen como realizar estas actividades y se proponen las plantillas para estandarizar dichas acciones.

➤ Detectar posibles conflictos en los requisitos funcionales:

Para este paso, se necesita analizar la estructura y gramática de los requisitos funcionales.

➤ Definir actores del sistema:

Encontrar a los actores tiene gran importancia en el desarrollo de la Ingeniería de requisitos en un sistema. Cada uno de los fenómenos externos con el cual el sistema debe interactuar es representado por un actor. Para encontrar a los actores, es preciso realizar las siguientes preguntas:

¿Cuáles grupos de usuario requieren ayuda del sistema para realizar sus tareas?

¿Cuáles grupos de usuario son necesarios para que funcione nuestro sistema?

¿Cuáles grupos de usuario están obligados a realizar funciones secundarias, como el mantenimiento de sistema?

¿Interactuará el sistema con cualquier hardware externo o sistema del software?

Para determinar si se han seleccionado los actores correctos (en el caso de ser hombres), se puede tratar de bautizar a dos o tres personas que pueden funcionar como actores, y luego ver

si el set de actores que se ha creado es suficiente para las necesidades de los las personas que se han bautizado.

En un principio puede ser difícil encontrar los actores adecuados. Esto se basa generalmente a que en esta etapa aun no se han encontrado todos los casos de uso. Trabajar con los casos de uso es lo único que le da una comprensión más profunda del ambiente del sistema. Cuando se logra definir todos los casos de uso, se regresa a los actores definidos anteriormente y por lo general se crean nuevos actores. Es importante tener mucho cuidado a la hora de cambiar actores ya que esto puede afectar a los casos de uso. Cualquier modificación para los actores constituye una alteración principal en las interfaces del sistema y el comportamiento del mismo.

Con el fin de lograr estandarizar el proceso de definición de actores, se recomienda el uso de la plantilla que se observa en la **tabla 3**. Con el apoyo de ésta, se pueden definir y describir un listado de actores del sistema.

Id <autor>	Nombre <i>descriptivo</i> Ej.: Estudiante, Secretaria.
Descripción	Este actor representa a <rol que representa el actor>
Comentario	<comentarios adicionales sobre el actor>

Tabla 3: Plantilla para actores de sistema.

Nombre descriptivo: El nombre debe denotar claramente el papel del actor. Es necesario asegurarse de que en una etapa futura no exista alguna confusión con el nombre del actor.

Descripción: Breve descripción que incluye el área de responsabilidad del actor, y lo que el mismo necesita del sistema. Como los actores representan una parte externa del sistema, usted no necesita describirlos detalladamente.

➤ Descripción de Casos de uso del sistema:

Cada caso de uso describe detalladamente, la aparición gradual de cómo interactúa el sistema con los actores y lo que el sistema hace en cada uno de estas interacciones. Los casos de uso funcionan como un hilo unificador a todo lo largo del proceso de desarrollo del software.

Cuando en un primer contorno los actores son seleccionados completamente, el siguiente paso es buscar los casos de uso del sistema. Los primeros casos de uso son muy preliminares, y solo serán cambiados en caso que sea necesario con el fin de lograr su estabilidad. Si la vista del sistema o los requisitos son deficientes, entonces el análisis de sistema es ambiguo y por consiguiente la funcionabilidad del sistema será poco clara.

Cada paso en la búsqueda de los casos de uso es importante y conviene prepararse para agregar, remover, combinar, y dividir los casos de uso antes de llegar a una versión final. La mejor comprensión de los casos de uso se realiza una vez se concluya su descripción detalladamente.

La mejor forma para encontrar casos de uso es considerar lo que cada actor requiere del sistema. Se recuerda que el sistema existe sólo para sus usuarios, y por consiguiente debería basarse en las necesidades de los usuarios. Muchas de las necesidades de los actores se reconocen a través de los requisitos funcionales.

Uno de los métodos más utilizados para identificar los casos de uso es mediante la elaboración de interrogaciones. Para cada actor, sea humano o no, se debe realizar las siguientes preguntas:

- ¿Qué tareas tiene que realizar el actor para que el sistema funcione?
- ¿El actor creará, almacenará, cambiará, quitará, o leerá los datos en el sistema?
- ¿Necesitará el actor informar el sistema acerca de los cambios externos?

- ¿Necesita el actor estar al corriente de ciertas ocurrencias en el sistema?
- ¿Realizará el actor una puesta en marcha o un cierre de sistema?

Las respuestas para estas preguntas representan los flujos de acontecimientos que identifican los casos de uso candidatos. Una cierta cantidad puede ser modelada como las variantes del mismo caso de uso, aunque siempre no es fácil decidir lo que es una variante y cuál es un caso separado. Sin embargo, se volverá más claro cuando se hace una descripción detallada de los flujos de eventos. (RATIONAL 2006)

Generalmente, para la creación de los casos de uso, se toman como base a los requisitos funcionales. En busca de lograr un patrón se propone utilizar la plantilla ubicada en la **tabla 4**.

Identificador del caso de uso	<i><Nombre del caso de uso ></i>	
Objetivos del sistema asociados al requisitos	<i><Identificador obj. , nombre del objetivo del sistema></i>	
Requisitos asociados	<i><Se indican otros requisitos que estén asociados por algún motivo con el requisito que se está describiendo.></i>	
Actor	<i><Se relaciona el actor con el caso de uso, qué acciones realiza el mismo.></i>	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso { durante la realización de los casos de uso <i><lista de casos de uso></i> , cuando <i><evento de activación></i> }	
Precondición	<i><Condiciones necesarias para que se pueda realizar el caso de uso></i>	
Flujo normal de eventos	Paso	Acción
		<i><Sucesión normal de interacciones del caso de uso.></i>
Flujo alterno		
Poscondición	<i><Condiciones que se deben cumplir después de la terminación normal del caso de uso></i>	
Importancia	<i><importancia del caso de uso></i>	

Comentario	<Comentarios adicionales sobre el caso de uso >
------------	---

Tabla 4: Plantilla para la descripción detallada de casos de uso

Identificador del caso de uso: Cada caso de uso debe tener un nombre que indica lo logrado por sus interacciones con el actor. El nombre puede tener varias palabras para ser entendido. No pueden existir dos casos de uso con el mismo nombre.

Descripción: Debe completarse de forma distinta en función de que el caso de uso sea abstracto o concreto. Si el caso de uso es abstracto, deben indicarse los casos de uso en los que se debe realizar, es decir, aquellos desde los que es *incluido* o a los que *extiende*. Si, por el contrario, se trata de un caso de uso concreto, se debe indicar el *evento de activación* que provoca su realización. Apoyarse para esta descripción en el vocabulario común que se definió en el glosario de términos.

Flujo normal de eventos: En este punto, se necesita crear un borrador del flujo de acontecimientos del caso de uso. Se describe el flujo de eventos de cada caso de uso como los instantes breves de actuación, pero no se entra en detalles. Ej.:

El cliente presiona el botón "Comenzar".

El cliente inserta los artículos.

El sistema comprueba el tipo de los artículos insertos por el cliente.

El cliente presiona el botón "Imprimir".

El sistema imprime el recibo.

Para lograr elaborar un flujo de eventos con consistencia se recomienda utilizar las siguientes preguntas:

- ¿Cómo arranca el caso de uso? El principio del caso de uso debe describir la señal que activa el caso de uso de forma clara. Ej.: "El caso de uso comienza cuando ocurre...".
- ¿Cómo termina el caso de uso? Se declara claramente pase lo que pase en el transcurso del flujo de eventos cuando termina el caso de uso. Ej.: "Cuando... ocurre, el caso de uso termina".

- ¿Cómo interactúa el caso de uso con los actores? Para minimizar cualquier riesgo de malentendido estructure la descripción como una serie de párrafos, en el cual cada párrafo expresa una acción en el formato: "Cuando el actor hace..., el sistema realiza...". Se puede enfatizar la interacción escribiendo que el caso de uso envía y recibe señales de actores, por ejemplo: "El caso de uso arranca cuando recibe la señal ' inicio ' del Operador".
- ¿Cómo intercambia el caso de uso datos con un actor? Esta pregunta se refiere a la discusión de señales. Ej.: "el caso de uso echa a andar cuándo el Usuario pone en el sistema su Usuario y contraseña".
- ¿Cómo repite el caso de uso algún comportamiento? Este cuestionamiento debe expresarse en el lenguaje natural. Sin embargo, existen casos excepcionales, en el cual se recomienda el uso de algunas sentencias de lenguajes de programación tales como "WHILE-END, IF-THEN-THEN ELSE," y " el ciclo FOR," si las condiciones naturales y correspondientes de lenguaje son difícil de expresar. En general, se debe evitar el uso de este lenguaje porque son difíciles de leer y comprender para las personas que no tienen conocimientos sobre el tema.
- ¿Hay cualquier situación optativa en el flujo de eventos del caso de uso? Algunas veces un actor puede seleccionar varias actividades, partiendo desde un mismo punto. Esto se escribe conforme ocurre. Por ejemplo:
 - “El actor escoge la opción...
 - A). . .en la que realiza.....
 - B). . .en el cual introduce...
- ¿Cómo el caso de uso debería estar descrito a fin de que el cliente y los usuarios lo pueden entender? El uso de terminología especificada en la metodología, como el caso de uso, el actor, y la señal, podrían hacer el texto difícil de entender. Para simplificar el texto a leer, se puede enumerar las acciones o adoptar algunas otra estrategia en el que se describa toda la actividad que define el caso de uso.

Si el número de actores o casos de uso se vuelve demasiado grande, se divide en paquetes de caso de uso para simplificar el mantenimiento del modelo de caso de uso. Esto hace que los casos de uso sean más fáciles de entender y simplifica la asignación de responsabilidades en cada uno de ellos. Uno de los argumentos más utilizados en la confección de paquetes es si los casos de uso interactúan con el mismo actor. (RATIONAL 2006)

➤ Diagramas de casos de uso:

Contiene los diagramas de casos de uso del sistema que se hayan realizado. En estos diagramas, los actores se representan en forma de pequeños monigotes y los casos de uso se representan por elipses contenidas dentro de un rectángulo que representa al sistema. La participación de los actores en los casos de uso se indica por una flecha entre el actor y el caso de uso que apunta en la dirección en la que fluye la información. Ver **figura 2.5**

Las relaciones entre casos de uso son:

- Includidos: (el caso de uso B es incluido del caso de uso A cuando el caso de uso B es una parte del caso de uso A).
- Extendidos (el caso A es extendido del caso B cuando A es una subsecuencia de interacciones de B que ocurre en una determinada circunstancia).
- Generalización/Especialización: Si dos o más casos de uso tienen similitudes en la estructura y el comportamiento, entonces se puede factorizar el comportamiento común para crear un *caso de uso padre*.

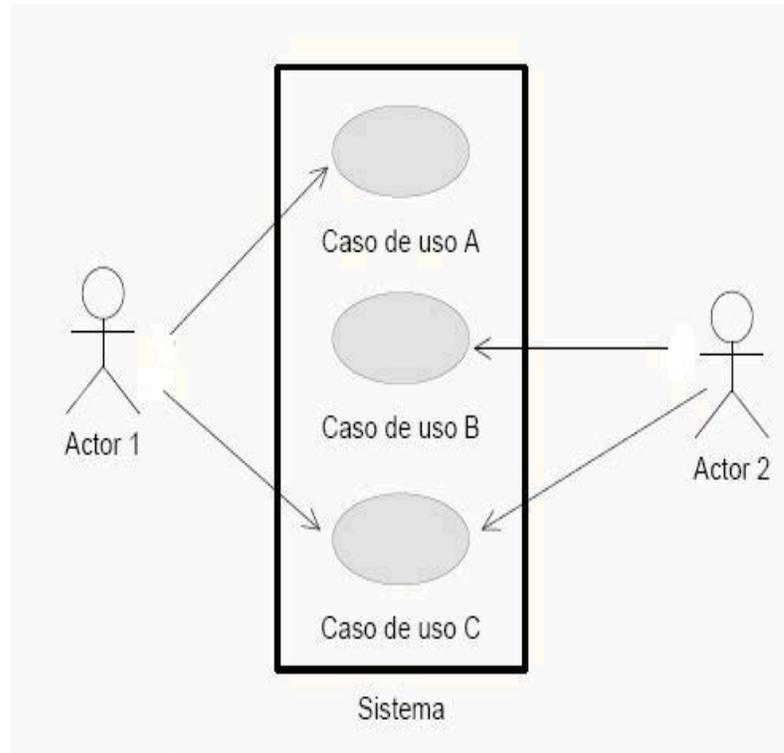


Figura 2. 5: Diagramas de casos de uso

2.2.2.3 Analizar requisitos no funcionales

Objetivos:

- Detectar posibles conflictos en los requisitos funcionales.
- Profundizar en el conocimiento de los requisitos.

Se procede al análisis de los requisitos no funcionales que han sido negociados con el cliente para ver si éstos tienen la calidad necesaria y pueden ser perfectamente entendidos por el

equipo de desarrollo. En caso de encontrar algún conflicto en los mismos, se procede a cambiarlos en colaboración con el cliente.

Para este análisis se tomará como base la plantilla que se generó en la captura de requisitos. Ver **tabla 1**

2.2.2.4 Desarrollar prototipos

Objetivos:

- Obtener requisitos relativos a la interfaz de usuario.
- Detectar posibles conflictos en los requisitos.
- Profundizar en el conocimiento de los requisitos.

En esta tarea se deben desarrollar prototipos que permitan tanto a los clientes y usuarios como a los desarrolladores tener una idea más clara del sistema a desarrollar e identificar nuevos requisitos o conflictos que hayan permanecidos ocultos hasta el momento. Lo más habitual es que el prototipo sea *desechable*, es decir, que una vez que se haya utilizado no se desarrolle tomando su código como base.

2.2.2.5 Priorizar casos de uso

El arquitecto del software propone los contenidos técnicos y el orden de iteraciones sucesivas, seleccionando un cierto número casos de uso que serán analizados y diseñados con mayor importancia. Esta propuesta especializada se debe completar y refinar teniendo en cuenta, la disponibilidad de empleados, los requisitos que fueron propuestos por el cliente, disponibilidad de herramientas y las necesidades de otros proyectos. Como referencia se utilizan los siguientes criterios: importante, crítico, útil.

Para cumplir con esta tarea (priorizar los casos de uso) hay que analizar los siguientes aspectos:

- El impacto arquitectónico del caso de uso (importancia que presenta el caso de uso dentro del sistema).
- Los riesgos a ser mitigados (actuación, disponibilidad de un producto, y acomodamiento de un componente).
- La terminación de la cobertura de la arquitectura (asegurarse de que al final de la fase de Elaboración, cada caso de uso a desarrollar haya sido implementado).
- Otras restricciones u objetivos tácticos: El demo para el usuario final, etcétera.

2.2.3 Validación de requisitos

Los requisitos una vez determinados necesitan ser validados. La validación de requisitos tiene como misión demostrar que la definición de los requisitos puntualiza realmente el sistema que el usuario necesita o el cliente desea. Es obligatorio asegurar que el análisis realizado y los resultados obtenidos de la etapa de definición de requisitos sean correctos. Pocas son las propuestas existentes que ofrecen técnicas para la realización de la validación y muchas de ellas consisten en revisar los modelos obtenidos en la definición de requisitos con el usuario para detectar errores o inconsistencias. Para esta etapa el rol de revisor procede a la revisión del DRS.

Pasos recomendados

- Validar el catálogo de requisitos
- Cerrar la versión de los requisitos

2.2.3.1 Validar el catálogo de requisitos

Objetivo:

- Validar los requisitos de almacenamiento, funcionales y no funcionales.

Par el cumplimiento del objetivo se debe asegurar que los requisitos obtenidos representen realmente las necesidades de clientes y usuarios.

Se propone para ello utilizar las siguientes técnicas:

Técnicas para la validación de requisitos

- **Reviews o Walk-throughs:** Está técnica consiste en la lectura y corrección completa de la documentación o modelado de la definición de requisitos. Con ello solamente se puede validar la correcta interpretación de la información transmitida. Lo más difícil es verificar la consistencia de la documentación o información faltante.
- **Auditorias:** La revisión de la documentación con esta técnica consiste en un chequeo de los resultados contra una checklist predefinida o definida a comienzos del proceso, es decir que sólo una muestra es revisada.
- **Matrices de trazabilidad:** Esta técnica consiste en marcar los objetivos del sistema y chequearlos contra los requisitos del mismo. Es necesario ir viendo qué objetivos cubre cada requisito, de esta forma se podrán detectar inconsistencias u objetivos no cubiertos. Luego debemos comparar esta matriz con la que se realizó en la fase de elicitación.

Es recomendable además de la técnica de Reviews o Walk-throughs utilizar la matriz de trazabilidad, ya que esta permite relacionar los objetivos del sistema con los requisitos obtenidos en el proceso de ingeniería.

2.2.3.2 Cerrar la versión de requisitos

Objetivo:

- Dar una versión completa del DRS.

Si no han aparecido nuevos conflictos durante el proceso de validación, se debe llegar a un acuerdo entre clientes y desarrolladores, cerrando la versión actual de los requisitos, siempre teniendo en cuenta, que representa el conocimiento actual de los requerimientos y que podrán sufrir cambios en un futuro.

2.2.4 Gestión de cambios

Objetivo:

- Manejar los cambios realizados a los requisitos.
- Creación de la línea base

2.2.4.1 Manejar los cambios realizados a los requisitos.

Los cambios en los requisitos son manejados por el Plan para la Administración de Requisitos el cual tiene como objetivo documentar requisitos, sus atributos y sus líneas directivas para dar posibilidad de rastreo y contribuir con la gestión de cambios en los mismos. Este plan se realiza antes de comenzar con la descripción de los requisitos del sistema, puesto que en él se decide como documentar, organizar y utilizar los requisitos durante todo el proceso de desarrollo del software. Alguno de los puntos a tener en cuenta para la creación de este plan son:

- Evaluar el impacto que tiene un cambio de requisito sobre el proyecto.
- Evaluar el impacto que provoca un fracaso en la prueba de los requisitos (o sea si la prueba falla, entonces el requisito no puede ser correcto).
- Manejar el alcance del proyecto.
- Comprobar que todos los requisitos del sistema se cumplen en la implementación.
- Comprobar que la aplicación hace sólo lo que fue pretendida para hacer.
- Manejar cambios en requisitos.

Algunas de las principales tareas a seguir para la gestión de requisitos son:

- Reconsiderar los requisitos obtenidos y su respectiva trazabilidad: Aun si un requisito no se ha alterado durante todo el proceso, los atributos y la posibilidad de rastreo asociada con el mismo pueden alterarse. El analista de sistema es responsable de mantener esta información totalmente actualizada en el Plan para la Administración de Requerimientos y en el DRS.
- Realizar cambios en los requisitos: Un cambio para un requisito puede tener una "onda" de efectos sobre otros requisitos relacionados y con esto, afectaciones sobre otros artefactos que se originan durante la realización de esta disciplina. Para manejar este efecto, se aconseja cambiar los requisitos de arriba hacia abajo. Para manejar el impacto de cambio de requisitos, se incluye la revisión de la información relacionada a la actividad que referencia el requerimiento. Las informaciones de posibilidad de rastreo son útiles para la determinación de los elementos potencialmente afectados.

2.2.4.2 Creación de la línea base

El IEEE define una línea base como una especificación o producto que se ha revisado formalmente y sobre los que se ha llegado a un acuerdo, y que de ahí en adelante sirve como base para un desarrollo posterior y que puede cambiarse solamente a través de procedimientos formales de control de cambios. (IEEE 1990)

Para el caso de la Ingeniería de requisitos la creación de una línea base sería obtener un punto de partida dentro del proceso de captura-validación definido anteriormente, con el fin llevar una revisión continua de los cambios existentes dentro de este ciclo de desarrollo, con eficiencia de tiempo, ya que con la creación de esta línea base no es necesario realizar la revisión del proceso desde el principio sino desde el punto que ya ha sido revisado por el equipo de desarrollo. Esta actividad incluye la creación de casos de pruebas para los requisitos

funcionales, a partir de los casos de uso. Con esto se genera automáticamente un borrador de pruebas de aceptación del sistema.

2.3 Documento de Requisitos del Sistema (DRS)

Muchos programadores están convencidos de que escribir un documento de requisitos es una pérdida de tiempo y que por muchos defectos que pueda tener ir directamente al diseño después de algunas charlas informales con el cliente, por lo menos esta estrategia evita el tiempo y el coste de escribir un enorme documento que no sirve para nada.

La comunidad científica ha destacado la importancia de contabilizar todos los pasos que se realizan durante el proceso de Ingeniería de requisitos, pasos que deben ser guardados en documentos donde queden reflejados todos los artefactos que genera este flujo de trabajo y las conclusiones que se llegue con los usuarios y clientes. Para ello se propone la creación el DRS.

El DRS establece lo que espera el sistema, incluyendo tanto la definición como especificación de los requisitos. Es importante mencionar que este no es un documento de diseño; es decir, debe establecer QUÉ hace el sistema, pero no el CÓMO. Además los requisitos deben establecerse de forma que sea posible su seguimiento durante todo el proceso de desarrollo.

Según los aspectos que fueron señalados en las secciones anteriores que debían formar parte del DRS, se propone la siguiente estructura para dicho documento con el fin de lograr un estándar y ganar en organización. Mencionar antes de mostrar la estructura del DRS, que muchos de los puntos que se proponen a continuación se realizan en la documentación de los productos de software que se realizan actualmente; pero no con la estandarización necesaria, lo que impide en ocasiones la correcta corrección de errores o estudio del sistema en caso de que sea necesario. El DRS tiene que estar basado en el Plan para la Administración de Requisitos (ver **sección 2.2.4.1**).

Los principales aspectos que debe tener este documento se muestran en la **figura 2.6**

2.3.1 Estructura del DRS

Presentación

Como toda presentación, para el caso del DRS, este contará con el nombre del proyecto que esta realizando el levantamiento de requisitos y fecha de publicación del Documento. Además contendrá el equipo de desarrolladores que esta realizando esta tarea y el nombre del cliente, que generalmente es una empresa. Además contendrá la versión del documento.

Índice

El índice del DRS se utilizará como guía para indicar la página en la que comienza cada sección, subsección o apartado del documento.

Introducción

Esta sección debe contener una descripción breve de las principales características del sistema software que se va a desarrollar, se debe especificar la situación actual que genera la necesidad del nuevo desarrollo, la problemática que se acomete, y cualquier otra consideración que sirva de apoyo al lector a la hora de comprender el Documento.

Participantes en el proyecto

Aquí debe quedar plasmada una lista con todos los participantes en el proyecto, tanto desarrolladores como clientes y usuarios. Para cada participante se deberá indicar su nombre y el papel que desempeña en el proyecto.

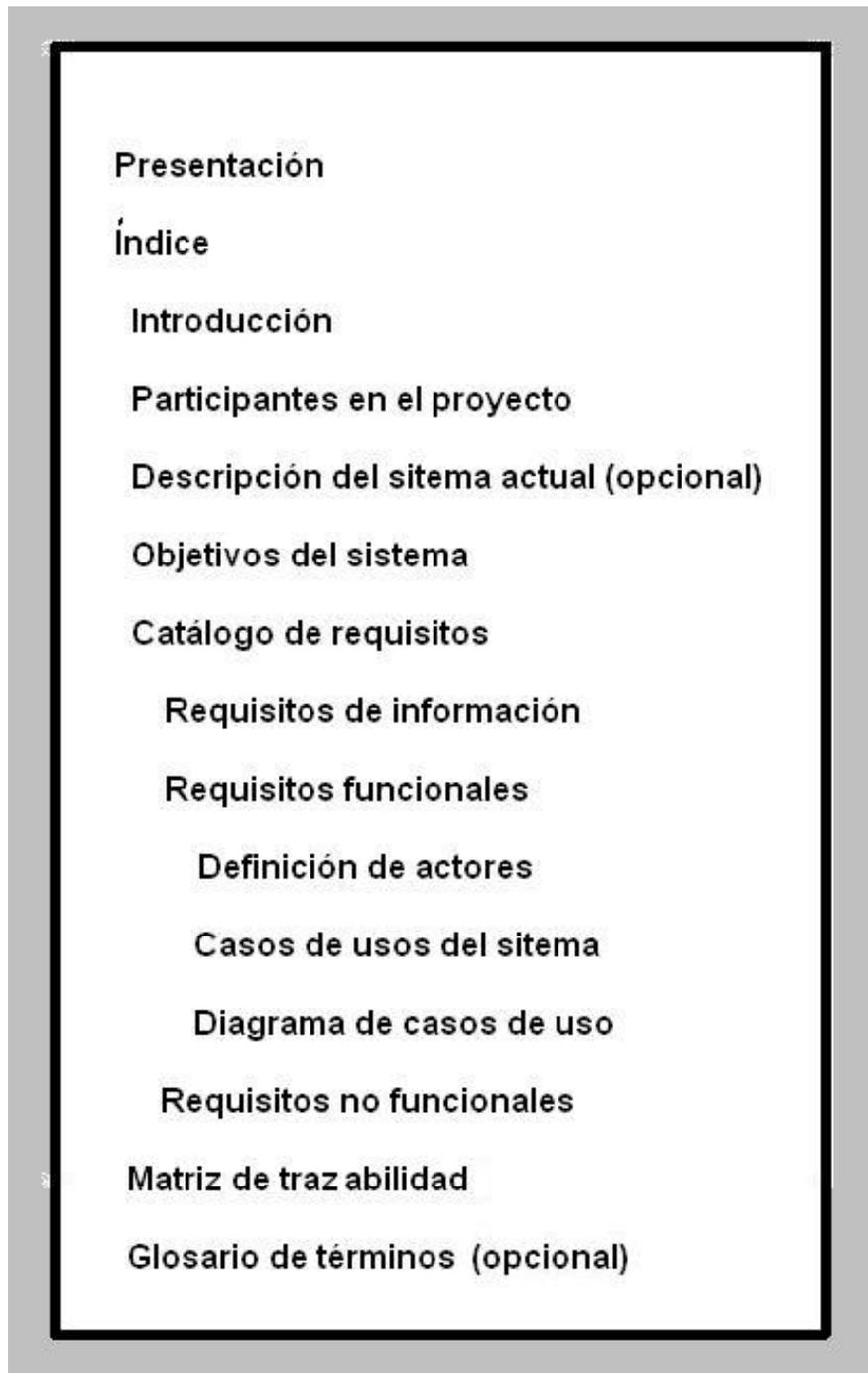


Figura 2. 6: Documento de Requisitos del Sistema

Descripción del sistema actual [opcional].

Esta sección es opcional teniendo en cuenta que puede o no existir sistema actual, es decir, que se quiera mejorar un sistema que ha sido creado anteriormente y que necesita ser mejorado o cambiarle sus funcionalidades.

Objetivos del sistema

En esta sección se plasmará los objetivos que deberá cumplir el sistema una vez el software esté en explotación.

Se recomienda enumerar los objetivos para facilitar su utilización en tareas posteriores, dar una pequeña descripción e importancia que tiene dentro del proceso de desarrollo. En caso de existir algún tipo de aclaración también se le puede adicionar.

Catálogo de requisitos del sistema

En esta sección se especificarán cada uno de los requisitos que fueron capturados teniendo en cuenta su naturaleza. Como fue anteriormente analizado los requisitos se dividen en tres grandes grupos: requisitos de almacenamiento de información, requisitos funcionales y los no funcionales. Por esta razón esta sección contará con subsecciones que darán mayor organización a este documento.

Requisitos de almacenamiento de información.

En esta subsección se debe listar aquellos requisitos correspondientes a esta característica que fueron encontrados durante el levantamiento de requisitos. Utilizar tabla 1

Requisitos funcionales

Esta subsección debe contener la lista de requisitos funcionales que se hayan identificado. Para ello se recomienda la utilización de la tabla 1, la cual sólo se diferencia de la que se utiliza para el grupo anterior, en el campo de descripción del requisito. Además cuenta con la definición de actores, casos de usos del sistema y diagramas de casos de uso.

Requisitos no funcionales

En esta subsección se debe listar aquellos requisitos no funcionales que fueron encontrados durante el levantamiento de requisitos. Para ello se recomienda la utilización de la tabla 1.

Matriz de trazabilidad objetivos/requisitos

El principal objetivo de esta matriz es reflejar y conocer la relación que existe entre los objetivos que deben cumplir el sistema y los requisitos.

Glosario de términos [opcional]

Esta sección, será incluida de forma opcional. Esto dependerá si existen o no términos específicos del dominio del problema que puedan provocar algún tipo de dificultad al lector a la hora de captar el mensaje del documento. Conviene realizar glosario de términos en el caso de haber utilizado abreviaturas. Para cada uno de los términos se deberá plasmar su significado.

Este apartado es importante porque define una terminología común que está usada consistentemente a través del proyecto u organización.

2.4 Curso para la capacitación de los estudiantes en la disciplina Ingeniería de Requisitos.

Con fin de mejorar el conocimiento de los estudiantes de la Universidad de las Ciencias informáticas se propone, para dichos educandos impartir el curso que se expone seguidamente.

Datos Generales			
Disciplina:	Ingeniería y gestión de software		
Curso Optativo:	Introducción a la Ingeniería de Requisitos		
Perfil:	Calidad		
Año:	3er Año	Semestre:	2do

2.4.1 Objetivos Generales

- Transmitir los conceptos básicos, estado actual y el futuro de la Ingeniería de Requisitos y del prototipado.
- Mostrar ejemplos de casos reales que ilustren la importancia de aplicar adecuadamente Ingeniería de requisitos en el proceso de desarrollo de software.
- Comparar ventajas e inconvenientes de utilizar técnicas de especificación formal frente a técnicas clásicas en la especificación y análisis de requisitos.
- Se enseñará cómo producir una documentación correcta y completa de especificación de requisitos de software de acuerdo con los estándares internacionales, guías nacionales y prácticas más comunes.
- Se enseñarán las principales técnicas de la notación estándar UML, que pueden utilizarse en el modelado de requisitos.
- Se mostrarán los principales enfoques, estrategias y modelos de proceso en la aplicación de la Ingeniería de requisitos.

2.4.2 Objetivos Educativos:

- Consolidar un estilo de trabajo independiente colectivo y creativo en la solución de los problemas, con un nivel de profesionalidad que garantice la sencillez y elegancia, así como la eficiencia y el ahorro de recursos en las soluciones adoptadas.
- Que los estudiantes utilicen y desarrollen una forma dialéctica de pensamiento y que lo apliquen consecuentemente en su enfoque sistémico de análisis.
- Que los estudiantes desarrollen la previsión, que sean capaces de anticiparse a problemas futuros y asumir posiciones responsables una vez que los conocen.
- La realización de trabajos prácticos ayudará a los alumnos a alcanzar una cierta capacitación para valorar, planificar y controlar las primeras etapas en el proceso de producción de software, como miembros de un grupo de desarrollo.

2.4.3 Metodología

Teórico-práctico, con lectura previa de parte de los participantes sobre documentos sugeridos, exposiciones magistrales por parte del profesor. Trabajo práctico en forma de taller. Aplicación de seminarios y preguntas escritas.

2.4.4 Descripción de los temas

Presentación de la asignatura – Se dedicará el primer día a presentar la asignatura, organización de prácticas, métodos de evaluación, etc.

Tema 1. Introducción a la Ingeniería de Requisitos

- 1.1 Ingeniería del software: una revisión
- 1.2 Ingeniería de Requisitos e Ingeniería del Software
- 1.3 Importancia de la Ingeniería de Requisitos en la práctica
- 1.4 Factores de calidad del software

Propuesta de realización de la Ingeniería de requisitos en los proyectos de la Universidad de las Ciencias Informáticas (UCI).

- 1.5 Estrategias más importantes de desarrollo de software
- 1.6 Resumen de fases genéricas en Ingeniería de Requisitos

Tema 2. Actividades y conceptos básicos en Ingeniería de Requisitos

- 2.1 Definiciones, objetivos y actores relacionados con los requisitos
- 2.2 Tipos de requisitos que debe incluir la especificación
- 2.3 El documento de especificación de requisitos
- 2.4 Algunas realidades sobre I.R.

Tema 3: Análisis del problema

- 3.1 Entendimiento del problema
- 3.2 Definición de términos del problema
- 3.3 Definir la visión del proyecto
- 3.4 Derechos y responsabilidades de clientes y usuarios

Tema 4: Elicitación de Requisitos

- 4.1 Introducción
- 4.2 La elicitación de requisitos
- 4.3 Problemas de la elicitación de requisitos (articulación, comunicación, limitaciones cognitivas, conducta humana, técnicos)
- 4.4 Técnicas de elicitación de requisitos
 - 4.4.1 Entrevistas
 - 4.4.2 Joint Application Development
 - 4.4.3 Brainstorming

Tema 5: Análisis de requisitos

5.1 El propósito del análisis de requisitos

5.2 Plantillas y patrones lingüísticos para análisis de requisitos

5.3 Definición de casos de uso. Descripción

5.4 Diagramas de casos de uso.

5.5 Documento de requisitos del sistema. Beneficios.

5.6 ¿Qué se incluye en el documento de especificación de requisitos del software (DRS)?

Tema 6. Validación de Requisitos

6.1 El proceso de validación y verificación de requisitos

6.2 Criterios de calidad de los requisitos según su tipo

6.3 Técnicas de validación de requisitos

6.4 Definición de prototipo orientado a clientes / usuario

6.5 Clases de prototipos. Ventajas e inconvenientes.

Tema 7: Gestión de cambios

7.1 Plan de administración de cambios. Características. Ventajas y desventajas

7.2 Línea base para la Ingeniería de requisitos

Bibliografía para el curso

- Se utilizará el Texto - Guía de la asignatura, preparado para este fin: “Ingeniería del Software. Gestión de Requisitos”, A. Toval; J. Nicolás, ed. DM, ICE-Universidad de Murcia, 1999.
- Ingeniería del Software: Un enfoque práctico, Roger S. Pressman, McGraw Hill, quinta edición
- Requirements Engineering. Processes and Techniques, Kotonya, G and Sommerville, I, John Wiley & Sons. 1998

- Mastering the requirement process. Robertson, S and Robertson, J, Addison-Wesley. 1999
- Software Requirements, Karl E. Wieggers, Microsoft Press, 1999
- Software Engineering (6th edition). Sommerville, I, Pearson Education Limited. 2001
- “Requirements Engineering: A Framework for understanding”, R.J. Wieringa, John Wiley&Sons LTD., 1996
- Software Requirements, A. Davis, Prentice Hall, 1.993
- REQUIREMENTS ENGINEERING: a good practice guide. Sommerville, I.Sawyer, Wiley, 30 APR 1997
- The Unified Modeling Language, G. Booch, J. Rumbaugh, I. Jacobson, Addison-Wesley 1999
- Formal Methods can deliver Provable Software, Nina Hall. Scientific Computing World. March 1995.

2.5 Conclusiones.

En este capítulo se ha dado respuesta al objetivo general de este trabajo ya que en el se define la propuesta para mejorar la Ingeniería de requisitos en los proyectos productivos de la UCI. Además se realiza un estudio de estos proyectos con el fin de basar dicha propuesta sobre el resultado obtenido en el mismo.

Es importante señalar las ventajas que brinda el proceso propuesto, ya que éste se basa en la interpretación de los principales problemas que se encuentran en la UCI en la aplicación de la Ingeniería de requisitos y les brinda una solución a los mismos.

Capítulo III: Evaluación de la propuesta.

Este capítulo cuenta en el epígrafe 3.1 con la definición y características del método Delphi. La sección siguiente corresponde a descripción y análisis de resultados que produjo el panel de expertos.

3.1 Método Delphi. Características

Teniendo en cuenta que el objetivo de este trabajo, es dar una propuesta para la aplicación de la Ingeniería de requisitos en la UCI, la misma debe ser evaluada por un personal con alta capacitación sobre el tema.

Con este fin se utiliza el Método Delphi, el cual consiste en la selección de un grupo de expertos a los que se les pregunta su opinión sobre cuestiones referidas a acontecimientos del futuro. Este procedimiento extrae y maximiza las ventajas que presentan los métodos basados en grupos de expertos y minimiza sus inconvenientes. Para ello se aprovecha la sinergia del debate en el grupo y se eliminan las interacciones sociales indeseadas que existen dentro de todo grupo. De esta forma se espera obtener un consenso lo más fiable posible del grupo de expertos.

Este método presenta tres características fundamentales:

- Anonimato: Durante un Delphi, ningún experto conoce la identidad de los otros que componen el grupo de debate. Esto tiene una serie de aspectos positivos, como son:
 - Impide la posibilidad de que un miembro del grupo sea influenciado por la reputación de otro de los miembros o por el peso que supone oponerse a la mayoría. La única influencia posible es la de la congruencia de los argumentos.
 - Permite que un miembro pueda cambiar sus opiniones sin que eso suponga una pérdida de imagen.

- El experto puede defender sus argumentos con la tranquilidad que da saber que en caso de que sean erróneos, su equivocación no va a ser conocida por los otros expertos.
- Iteración y realimentación controlada: La iteración se consigue al presentar varias veces el mismo cuestionario. Como, además, se van presentando los resultados obtenidos con los cuestionarios anteriores, se consigue que los expertos vayan conociendo los distintos puntos de vista y puedan ir modificando su opinión si los argumentos presentados les parecen más apropiados que los suyos.
- Respuesta del grupo en forma estadística: La información que se presenta a los expertos no es sólo el punto de vista de la mayoría, sino que se presentan todas las opiniones indicando el grado de acuerdo que se ha obtenido. (Caseres 2006)

Es decir, el método Delphi, clasificado como un método cualitativo o subjetivo procede por medio de la interrogación a expertos con la ayuda de cuestionarios sucesivos a la evaluación de un proyecto que se pondrá en práctica o alguna propuesta de desarrollo. La encuesta se lleva a cabo de una manera anónima (actualmente es habitual realizarla haciendo uso del correo electrónico o mediante cuestionarios web establecidos al efecto) para evitar los efectos de "líderes". Las preguntas se refieren, por ejemplo, a las probabilidades de realización de hipótesis o de acontecimientos con relación al tema de estudio.

Aunque, la formulación teórica del método Delphi propiamente dicho comprende varias etapas sucesivas de envíos de cuestionarios, de vaciado y de explotación, en buena parte de los casos puede limitarse a dos etapas, lo que sin embargo no afecta a la calidad de los resultados.

3.1.1 Pasos para la realización del Método Delphi

De manera resumida los pasos que se llevan a cabo para garantizar la calidad de los resultados son los siguientes:

Capítulo III: Evaluación de la propuesta

➤ Paso 1: Formulación del problema

Se trata de una etapa fundamental en la realización de un Delphi. En un método de expertos, la importancia de definir con precisión el campo de investigación es muy grande por cuanto que es preciso estar muy seguros de que los expertos reclutados y consultados poseen todas las mismas nociones de este campo. La elaboración del cuestionario debe ser llevada a cabo según ciertas reglas: las preguntas deben ser precisas, cuantificables (versan por ejemplo sobre probabilidades de realización de hipótesis y/o acontecimientos, la mayoría de las veces sobre datos de realización de acontecimientos) e independientes (la supuesta realización de una de las cuestiones en una fecha determinada no influye sobre la realización de alguna otra cuestión).

➤ Fase 2: Elección de expertos

La etapa es importante ya que el término de "experto" es ambiguo. Con independencia de sus títulos, su función o su nivel jerárquico, el experto será elegido por su capacidad de encarar el futuro y los conocimientos que posea sobre el tema consultado.

La falta de independencia de los expertos puede constituir un inconveniente; por esta razón los expertos son aislados y sus opiniones son recogidas por vía postal o electrónica y de forma anónima; así pues se obtiene la opinión real de cada experto y no la opinión más o menos falseada por un proceso de grupo (se trata de eliminar el efecto de los líderes).

➤ Fase 3: Elaboración y lanzamiento de los cuestionarios (en paralelo con la fase 2)

Los cuestionarios se elaboran de manera que faciliten, en la medida en que una investigación de estas características lo permite, la respuesta por parte de los consultados.

➤ Fase 4: Desarrollo práctico y explotación de resultados

Propuesta de realización de la Ingeniería de requisitos en los proyectos de la Universidad de las 72 Ciencias Informáticas (UCI).

El cuestionario es enviado a cierto número de expertos (hay que tener en cuenta las no-respuestas y abandonos. Naturalmente el cuestionario va acompañado por una nota de presentación que precisa las finalidades, el espíritu del Delphi, así como las condiciones prácticas del desarrollo de la encuesta (plazo de respuesta, garantía de anonimato).

Luego se procede al estudio de resultados para llegar a conclusiones.

3.2 Evaluación de la propuesta

Para la evaluación de la propuesta brindada en el capítulo anterior, se han escogido 7 expertos integrantes del claustro de profesores de la UCI o personal vinculados a algún proyecto que radica en este centro universitario. (Ver anexo 9)

El objetivo de este proceso evaluativo es darle validez a la solución propuesta, mediante la aplicación de cuestionarios a la muestra seleccionada. Esta selección se lleva a cabo, teniendo en cuenta la experiencia y capacitación que presenta el profesional sobre el tema. Las respuestas de este grupo de expertos pueden contribuir a perfeccionar el proceso propuesto.

3.2.1 Elaboración del cuestionario

Para la elaboración del cuestionario se tuvieron en cuenta los principios básicos que debería cumplir la propuesta presentada para su implementación en el proceso productivo de la UCI; además de los elementos que existen dentro de la organización en la actualidad y que podrían imposibilitar su ejecución. La encuesta fue conformada con preguntas abiertas de enfoque investigativo, sobre la validez de la solución propuesta al problema planteado y la evaluación del proceso, cerrando el entorno de las respuestas a los puntos básicos del problema tratado durante el desarrollo del trabajo. Se prefirieron estas preguntas, porque tienen la ventaja de

proporcionar una mayor riqueza en las respuestas que son brindadas por los especialistas. También se les dio la posibilidad de presentar su opinión general del objeto de evolución, para que tuvieran la libertad de expresar todo lo que se pudo obviar en la encuesta. (Ver anexo 8)

En todos los casos los expertos recibieron la documentación del proceso propuesto y se les requirió cumplir con un lapso de tiempo determinado para dar las respuestas o hacer las preguntas pertinentes que les hubiesen surgido al estudiar el documento presentado.

3.2.2 Resultados de las encuestas

Después de procesada la encuesta se pudo determinar que los expertos estuvieron de acuerdo en que era importante asegurar una correcta aplicación de la Ingeniería de requisitos en los proyectos productivos de la UCI.

Las referencias del por qué de esa respuesta, influyeron en que de esa manera se podía obtener en la UCI un software con mayor calidad, basada su elaboración en las funcionalidades descritas por requisitos con superior grado de eficacia. Además, alegaron la importancia de presentar una documentación completa de este proceso, ya que con ella se evita mal entendidos entre las personas que participan en esta disciplina.

Los criterios brindados por los expertos sobre la necesidad de un mejoramiento en esta disciplina en la universidad se basan en que ésta es la única manera de tener claridad en lo que se quiere con la aplicación. Además, con ella se define el qué desde un inicio y es la forma de prepararse en temas de organización y procesos que a veces no se imparten en la carrera. Otros criterios se refieren a la repercusión que tiene una mala gestión de requisitos dentro del proceso de desarrollo de software, lo cual puede provocar incumplimientos en el plan organizacional del proyecto, teniendo en cuenta que esto podría provocar demoras e insuficiencias en la calidad del producto.

Propuesta de realización de la Ingeniería de requisitos en los proyectos de la Universidad de las 74 Ciencias Informáticas (UCI).

Capítulo III: Evaluación de la propuesta

Lo anterior puede lograr un impulso en la obtención de una calidad sostenida en los requisitos y mejora el prestigio de la universidad, ya que esta etapa tiene como principal objetivo encontrar las necesidades de clientes y usuarios. Logrando este objetivo correctamente, se avanza muchísimo en el campo de obtención de software de excelencia.

En el cuestionario se indagó sobre cuáles eran los principales problemas que se presentaba en la disciplina Ingeniería de requisitos. Los expertos reconocieron muchos de los aspectos señalados en la encuesta e identificaron otros que consideraron de importancia. La tabla 5 muestra las coincidencias que hubo en sus respuestas y se adicionan aquellos problemas que ellos identificaron. Para el caso de los problemas adicionados, éstos parecerán en la tabla con letra cursiva.

Problemas en la captura de requisitos	Exp1	Exp2	Exp3	Exp4	Exp5	Exp6	Exp7	Total
Expresión de sus necesidades por clientes y usuarios.	1	1	0	0	0	1	1	4
Comunicación entre clientes, usuarios y desarrolladores.	1	1	0	1	1	1	0	5
Los requisitos cambian con el tiempo ya que a medida que los usuarios y clientes van conociendo sus necesidades y las posibilidades que le brinda la tecnología, surge la necesidad de reconsiderar decisiones anteriores.	1	1	1	1	1	1	1	7
En problemas complejos, los requisitos tienden a ser cada vez más complejos.	1	1	1	1	1	0	1	6
<i>Poca preparación del personal que realiza la captura de requisitos</i>	1	0	0	0	0	0	1	2
<i>Falta de seguimiento y validación de requisitos.</i>	1	0	0	0	0	1	0	2

Tabla 5: Problemas encontrados por los expertos

Capítulo III: Evaluación de la propuesta

Como se muestra en la figura los expertos reconocen que los problemas descritos en la propuesta, son los más encontrados durante la realización de la disciplina estudiada.

Otro de los puntos analizados en el cuestionario fue la selección de técnicas para lograr una mejor captura de restricciones. La propuesta reconoce como la más óptima la entrevista, teniendo en cuenta las condiciones que presentan los proyectos productivos en la UCI. El criterio valorativo de los expertos reportó los resultados que se muestran en la tabla 6

Técnicas para la captura	Exp1	Exp2	Exp3	Exp4	Exp5	Exp6	Exp7	Total
Entrevistas	1	1	1	1	1	1	1	7
Tormenta de Ideas(Brainstorming)	1	1	1	1	0	1	0	5
Cuestionarios o Checklists	1	1	0	0	0	0	0	2

Tabla 6: Técnicas evaluadas por expertos

Además de estas técnicas los expertos sugirieron investigaciones previas a documentos de la organización con que se trabaja, en caso de existir; y observar directamente la forma de trabajo y el flujo de información del objeto a informatizar.

Si analizamos los resultados comprobamos la aceptación de la entrevista como método eficiente en la captura de requisitos. La justificación dada por los expertos así lo demuestra donde definen a esta técnica como un método eficaz. Recargan además, que la calidad de esta actividad estará determinada por la preparación previa del encuestador que poseerá sobre el negocio en cuestión. La idea no es improvisar en la encuesta y preguntar lo que se nos ocurra en el momento. La preparación de esta encuesta es fundamental para ir a la misma con las ideas que realmente se quieren conocer, pues muchas veces el propio cliente, no conoce lo que quiere y es nuestro deber obtener esa información.

Todos los entrevistados estuvieron de acuerdo con la realización completa de las cuatro etapas que se definen en el proceso propuesto, enfatizando su evaluación, en la necesidad de mantener la trazabilidad desde los requisitos iniciales, conociendo sus variaciones durante todo el proceso y logrando su identificación a través de los casos de uso que fueron implementados

Capítulo III: Evaluación de la propuesta

en el caso de los requisitos funcionales. El porqué de estas respuestas se basa en que estas fases suponen un mejor refinamiento en el flujo de trabajo y que éstas no pueden realizarse de forma aislada, sino que debe existir una interrelación que posibilita la reutilización de los artefactos que se obtienen en cada una de ellas, reconociendo la necesidad de que este proceso se realice de forma iterativa.

A favor del proceso, el panel determinó que la UCI necesita mejorar la realización de la Ingeniería de requisitos, con la ejecución correcta dentro de su proceso productivo, de las cuatro etapas que se definen en la propuesta, ya que aumentan la calidad de los productos que se obtendrán. Otra de las ventajas señaladas es iteratividad que presenta el proceso ya que reconocen que los requisitos están sujetos a cambios los cuales tienen que ser analizados teniendo en cuenta en la etapa en que se origina el canje.

La creación de un DRS, fue otro de los aspectos señalados como positivos dentro de la propuesta, ya que en él se recoge todos los artefactos que se producen en el ciclo de desarrollo de esta disciplina. Además, consideran como ventaja, el aumento de la capacitación de los educandos del centro universitario sobre este tema.

En resumen, la propuesta fue evaluada como positiva ya que asegura un mejoramiento en la disciplina de Ingeniería de requisitos.

CONCLUSIONES

El estudio de la Ingeniería de requisitos es una fuente fundamental para lograr software con calidad. Es por eso que en este trabajo se realiza un estudio de las principales características de esta disciplina. Se ha logrado además, analizar la situación actual en que se encuentra la Universidad de Ciencias Informáticas en cuanto a la gestión de requisitos. Basado en este análisis se ha propuesto un grupo de actividades para lograr la realización del flujo de trabajo en un entorno iterativo.

La correcta realización de captura, análisis y validación de requisitos, conllevará a la mejora sustancial de la calidad de los mismos. Igualmente se le da gran importancia a la gestión de requisitos, encargada de darle seguimiento y controlar los cambios de éstos, en cualquier etapa de desarrollo del proyecto.

Gracias al estudio realizado se reafirma la necesidad de aumentar la capacitación de los estudiantes del centro universitario, siendo preciso, en la propuesta, especificar un curso para afianzar los conocimientos de los educandos en el tema de la Ingeniería de requisitos.

Con todo lo anterior, se puede concluir que este trabajo logra cumplir con los objetivos propuestos de forma satisfactoria.

RECOMENDACIONES

Para el buen desempeño y obtención de los resultados esperados se recomienda:

- Aplicar el proceso propuesto para la realización de la Ingeniería de requisitos en los proyectos productivos de la UCI.
- No aplicar este proceso si no se encuentra definido un Plan de Administración de Requisitos previamente.
- Integrar en los proyectos de la UCI los diferentes roles que se definen en el proceso propuesto.
- Estudiar las herramientas que existen en el mundo utilizadas para la gestión de requisitos.

REFERENCIAS BIBLIOGRÁFICAS

Amador Durán Toro, B. B. J. (2000). *Metodología para el Análisis de Requisitos de Sistemas Software*. Departamento de Lenguajes y Sistemas Informáticos .Escuela Técnica Superior de Ingeniería Informática. Sevilla, Universidad de Sevilla. Versión 2.2.

Amador Durán Toro, B. B. J. (2000). *Metodología para la Elicitación de Requisitos de Sistemas Software*. Sevilla, Departamento de Lenguajes y Sistemas Informáticos .Facultad de Informática y Estadística de Sevilla.

Caseres, E. (2006). *El método Delphi.Características*. [Disponible en: <http://www.codesyntax.com>.]

IEEE/EIA (1998). *Software Life Cycle Processes*.

Landazuri, B. A. M. (2004). *Definición de perfiles en herramientas de Gestión de Requisitos*. Departamento de Lenguaje y Sistemas Informáticos. Madrid, . Facultad Politécnica de Madrid.

M. G.Christe, K. C. K. (1993). *Issues in Requirements Elicitation*.

Pohl (1997). *Requirements Engineering*.

RATIONAL(2006) RUP Help.

Pressman (1997). *Ingeniería de software un enfoque practico*.

Referencias Bibliográficas

Johan (2006). *Análisis de requisitos*. Consultado 05/04/2007.[Disponible en: <http://ceds.nauta.es/index.htm>]

BIBLIOGRAFÍA

A.Duran, b. B., A.Ruiz. M.Toro (1999). *A requirements Elicitation Approach in templates and Paterns*.

Alcázar, E. G. (1996). *Análisis de Requerimientos y Trazabilidad en el Modelo de Objetos*. I Jornadas de OO de Madrid .

Autores, G. d. (2005). *Método Delphi*. Consultado 10/5/2007, [Disponible en: <http://www.gtlic.ssr.upm.es/encuestas/delphi.htm>]

B.W.Boehm, T. E. G. (1999). *Prototyping versus Specifying*.

Barra, C. (2003). *Software e Ingeniería de Software*, UNIVERSIDADES DE AMERICA LATINA.

Caseres, E. (2006). *El método Delphi.Características*. [Disponible en <http://www.codesyntax.com>]

Autores, (2006). *Clases de Ingeniería del Software I, curso 2006-2007*, UCI.

Duran, A. (2002). *Metodología para la elicitación de requisitos de software*. Sevilla, Universidad de Sevilla.

Hugo Estrada, A. M., Oscar Pastor, Juan Sánchez (2001). *Generación de Especificaciones de Requisitos de Software a partir de Modelos de Negocios: un enfoque basado en metas*. Valencia, Universidad politecnica de Valencia.

Isabel Villanueva, J. S., Óscar Pastor. (2005). *Elicitación de requisitos en sistemas de gestión orientados a procesos*. [Disponible en: // www.BDBComp.com]

Joaquín Lasheras, A. T., Joaquín Nicolás, Begoña Moros(2001) *Análisis de requisitos*, Universidad de Murcia.

Joaquín Lasheras, A. T., Joaquín Nicolás, Begoña Moros (2007). *Método SirenTool*. Departamento de Informática y Sistemas. Universidad de Murcia.

Juan Sánchez, J. B., Pablo Belenguer, David Pascual (2002). *VRU: Un método para validar requisitos y generar interfaces de usuarios multiplataforma*. Departamento Sistemas Informáticos y Computación . Valencia, Universidad Politécnica de Valencia.

LANDETA, J. (1999). *El método Delphi*. Sevilla, Universidad de Sevilla.

León, A. G. d. (2005). *Universidades latinoamericanas*. Consultado 10/05/2007 [Disponible en: <http://www.rau.edu.uy/universidad/univ.htm>]

M. G.Christe, K. C. K. (1995). *Issues in Requirements Elicitation*.

Manuel Resinas Arias de Reyna , P. T. M.-A. (2006). *Ingeniería del Software de Gestión II*. Consultado 05/04, 2007, [Disponible en: <http://www.lsi.us.es/docencia/>].

Peralta, Mario(2001) *Estimación del esfuerzo basada en casos de uso*. Centro de Ingeniería del Software e Ingeniería del Conocimiento, Buenos Aires, Argentina.

P.Sawyer , I. S. (1997). *Requirements Process Improvement throug The Phased Introductio ok Good Practice*.

Pohl (2000). *The three Dimensions or Requirements Engineerin*.

profesores, C. d. (2006). *Ingeniería de requisitos*. [Disponible en: http://fondosdigitales.us.es/public_thesis].

RATIONAL (2006). *RUP Help*.

Victor Anaya , P. I. (2004)*Traza de requisitos adaptadas a las necesidades del proyecto*. Departamento de sistemas informáticos y computación, Universidad Politécnica de Valencia.

Víctor Anaya, P. L. (2003). *SmarTTrace: Una Herramienta para Trazabilidad de Requisitos en Proyectos basados en UML*. Departamento de Sistemas Informáticos y Computación. Valencia, .Universidad Politécnica de Valencia.

Wikipedia. (2006). *Viabilidad técnica*. Consultado 05/06, 2007, [Disponible en: <http://es.wikipedia.org/wiki/>].

GLOSARIO DE TÉRMINOS

Actores del sistema: personas u otros sistemas que serán los orígenes o destinos de la información que consumirá o producirá el sistema a desarrollar y que forman su entorno.

Administración de proyectos: Proceso en el cual se lleva a cabo el análisis del proyecto a realizar. Se verifica tiempo de duración, personal involucrado y coste del proyecto.

Agentes: factores que pueden incidir en el fracaso o éxito de una aplicación.

CASE: Computer Aided Software Engineering

Checklist: cuestionarios. Técnica para la captura de requisitos.

DCCU: Documento de Confirmación de Clientes y Usuarios. Documento utilizado para confirmar las declaraciones de las necesidades de clientes y usuarios durante el proceso de captura de requisitos.

Demo: versión de un producto de software

DRS: Documento de Requisitos del Sistema. Documento utilizado para plasmar los artefactos que se obtienen durante la Ingeniería de requisitos.

Elicitación: término utilizado para referirse a la captura de requisitos

ESPITI: European Software Process Improvement Training Initiative

Etapas del Diseño: Fase de la Ingeniería de software donde se realiza el diseño del sistema

GAO: Oficina de Cuentas del gobierno norteamericano

Hardware: Parte física de una PC

Herramientas CASE: Herramientas utilizadas para el desarrollo de proyectos de Ingeniería de Software.

Implementación: Relación de códigos que dan cumplimiento a un diseño de software determinado

Interfaz: Frontera convencional entre dos sistemas o dos unidades, que permite intercambio de informaciones.

IR: Ingeniería de requisitos. Disciplina que se enmarca dentro de la Ingeniería de software.

Iteraciones: Secuencia de acciones.

JAD: Joint Application Development/Desarrollo conjunto de aplicaciones. Técnica para la captura de requisitos.

Microelectrónica: es la aplicación de la ciencia electrónica a componentes y circuitos de dimensiones muy pequeñas, microscópicas y hasta de nivel molecular para producir dispositivos y equipos electrónicos de dimensiones reducidas pero altamente funcionales.

Ordenador: Sinónimo de PC. Computadora Personal.

Plataforma: En informática, una plataforma es precisamente el principio, ya sea de hardware o software, sobre el cual un programa puede ejecutarse.

Proceso: Secuencia de actividades invocadas para producir un producto de software.

Prototipo: es una representación limitada del diseño de un producto que permite a las partes responsables de su creación experimentar su uso, probarlo en situaciones reales y explorar su uso.

Restricciones: Término utilizado en este trabajo como sinónimo de requisitos.

Requerimiento: sinónimo de requisito. Condiciones que debe cumplir un sistema.

Rol: Papel, cometido o función que tiene o desempeña que interpreta un actor.

RUP: El Proceso Unificado Rational (RUP) es una metodología de desarrollo para la programación orientada a objetos. Según Rational (diseñadores de Rose Rational y el Idioma Modelado Unificado), RUP está como un mentor en línea que mantiene pautas, plantillas, y ejemplos de todos los aspectos y fases de desarrollo del programa.

Simulación: Descubrir el comportamiento de un sistema.

Sinergia: sinónimo de reunión.

Sistema: Conjunto de funcionalidades que se obtienen durante un ciclo de desarrollo de software.

Sitio Web: Es un conjunto de páginas Web, típicamente comunes a un dominio de Internet o subdominio en la World Wide Web en Internet.

Software: Palabra en inglés utilizada para identificar a los programas de computadoras, a las aplicaciones.

UCI: Universidad de las Ciencias Informáticas

UML: “Unified Modeling Language” Lenguaje gráfico que brinda un vocabulario y reglas para especificar, construir, visualizar y documentar los artefactos de un sistema utilizando el enfoque orientado a objetos.

Validación: verificación de requisitos. Comprobar si los requisitos obtenidos son los deseados por los clientes.

Viabilidad: Condición que hace posible el funcionamiento del sistema, proyecto o idea al que califica, atendiendo a sus características tecnológicas y a las leyes de la naturaleza involucradas.

WYSIWYG: What You See Is What You Get, lo que se ve es lo que se obtiene. Técnica para la captura de requisitos.

ANEXOS

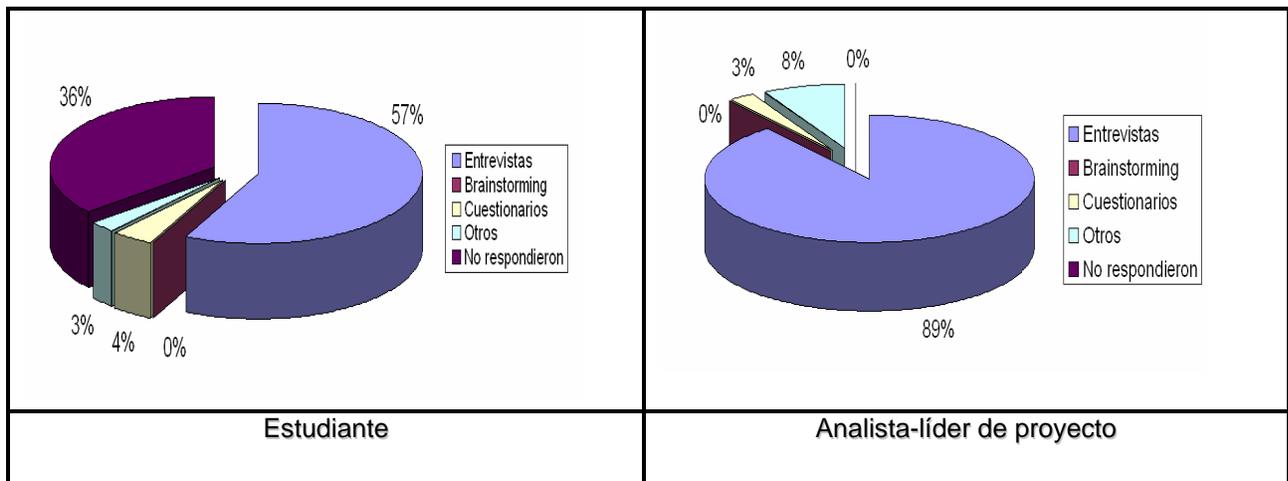
Anexo 1

1. ¿Existe en este proyecto un ingeniero de requisitos? En caso de respuesta negativa diga que rol se dedica ala realización del flujo de trabajo Levantamiento de requisitos.
2. ¿En este proyecto se realizan las tres etapas que forman parte de la Ingeniería de requisitos?
3. ¿Cuál técnica se utiliza en la captura de requisitos en este proyecto? --- Panel de experto. --- Entrevistas. --- Casos de Uso --- Otras.
4. Diga cual de lo siguientes aspectos reconoces como factor de éxitos y cuales como fracaso. ___ Especificaciones y requisitos cambiantes. ___ Involucración de los usuarios. ___ Apoyo de los directivos. ___ Falta de información por parte de los usuarios. ___ Especificaciones y requisitos incompletos. ___ Enunciado claro de los requisitos.
5. Listado de problemas que afectan la calidad del software, haciendo hincapié en los problemas que afectan la captura de requisitos a la hora de desarrollar un proyecto. Diga que importancia le das a estos problemas. (o a 5).En caso del 0, es para los problemas que usted considera que no son aplicables en este proyecto. ➤ Expresión de sus necesidades por parte de clientes y usuarios y comprensión de dichas necesidades por partes de los desarrolladores. ➤ Comunicación entre clientes, usuarios y desarrolladores. ➤ Existencia de limitaciones cognitivas (el ingeniero de requisitos debe tener conocimiento adecuado del dominio del problema y no hacer suposiciones sobre ello, al igual que clientes y usuarios no deben hacer suposiciones sobre aspectos tecnológicos). ➤ En problemas grandes y complejos, a la hora de realizar las entrevistas, algunas personas tienden a hacer simplificaciones no válidas, a ignorar partes más complejas o a centrarse únicamente en los aspectos que más conoce o que más les afectan. ➤ El software debe resolver problemas muy complejos por lo que los requisitos son cada vez más complejos. ➤ Los requisitos cambian con el tiempo ya que a medida que los usuarios y clientes van conociendo sus necesidades y las posibilidades que le ofrece la tecnología,

<p>surge la necesidad de reconsiderar decisiones anteriores o descubrir nuevas necesidades.</p> <p>➤ Una vez realizado los cambios en los requisitos, estos no quedan correctamente documentados.</p>
<p>6. Considera necesario aumentar la capacitación para realizar correctamente el flujo de trabajo de levantamiento de requisitos.</p> <p>___ Si ___ No</p>
<p>7. ¿Qué importancia le ves a la ingeniería de requisitos dentro de la ingeniería de software para lograr un producto con calidad? (Del 1 al 5)</p>

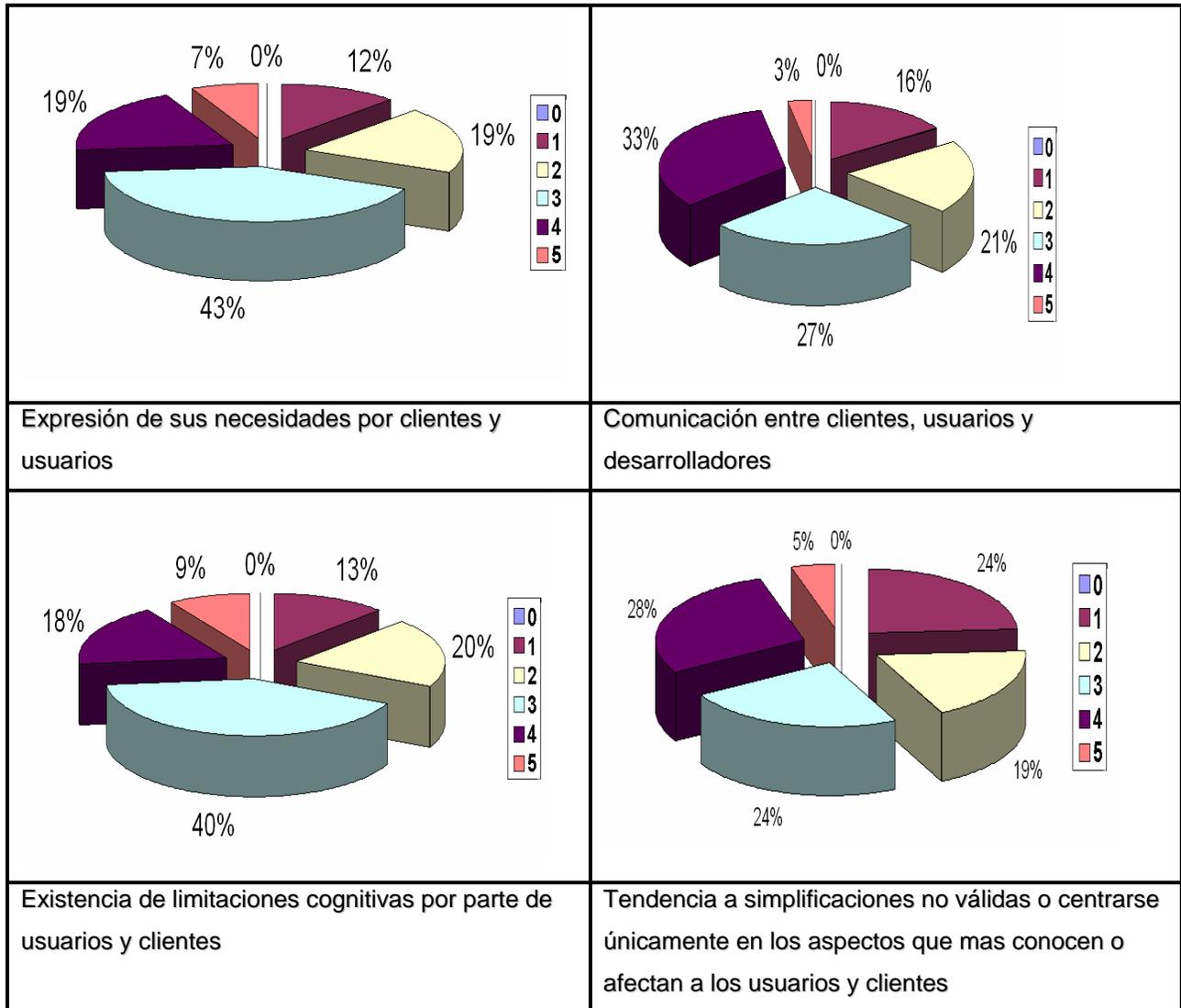
Encuesta realizada a analistas, líderes de proyecto y estudiantes

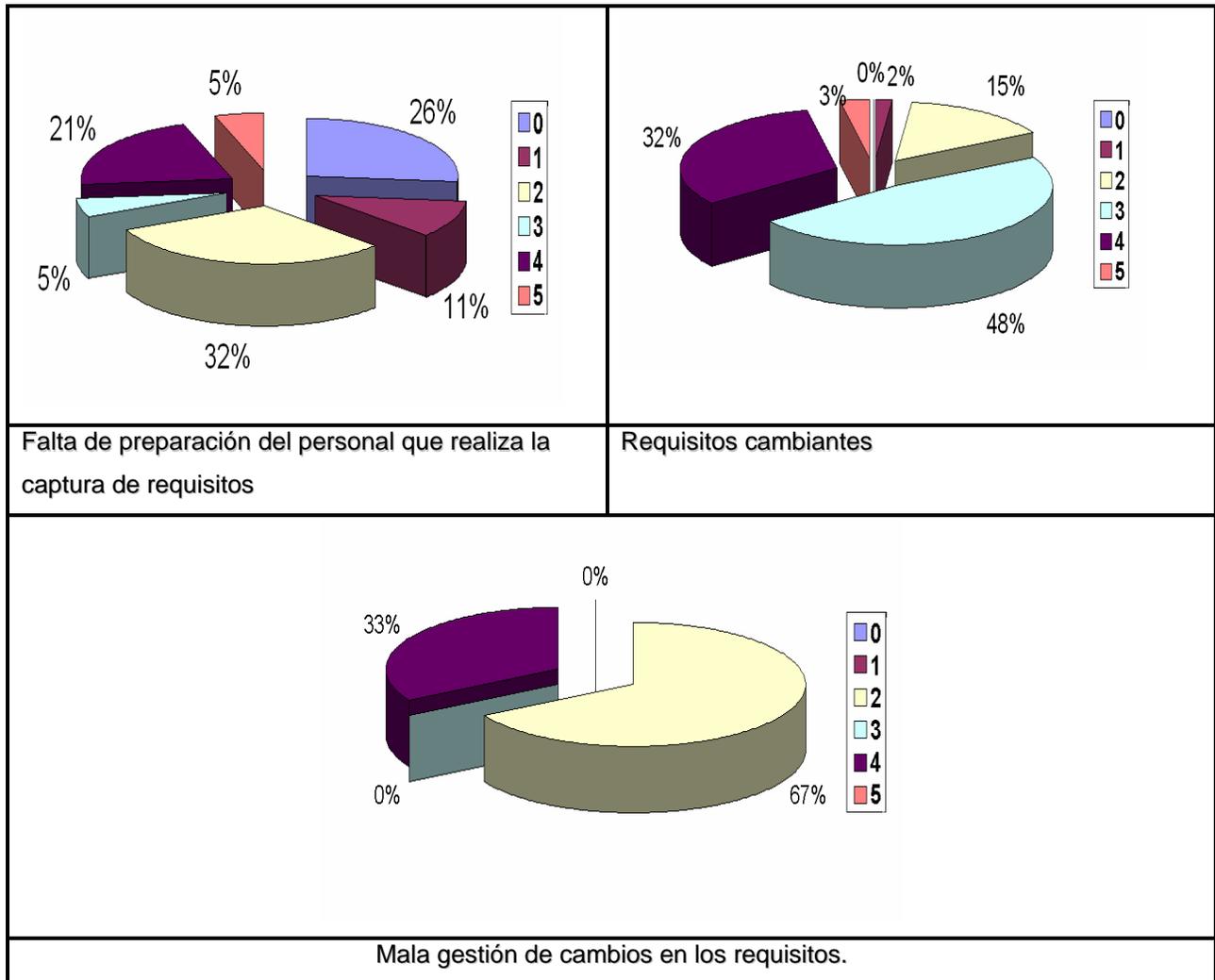
Anexo 2



Técnicas para la captura de requisitos.

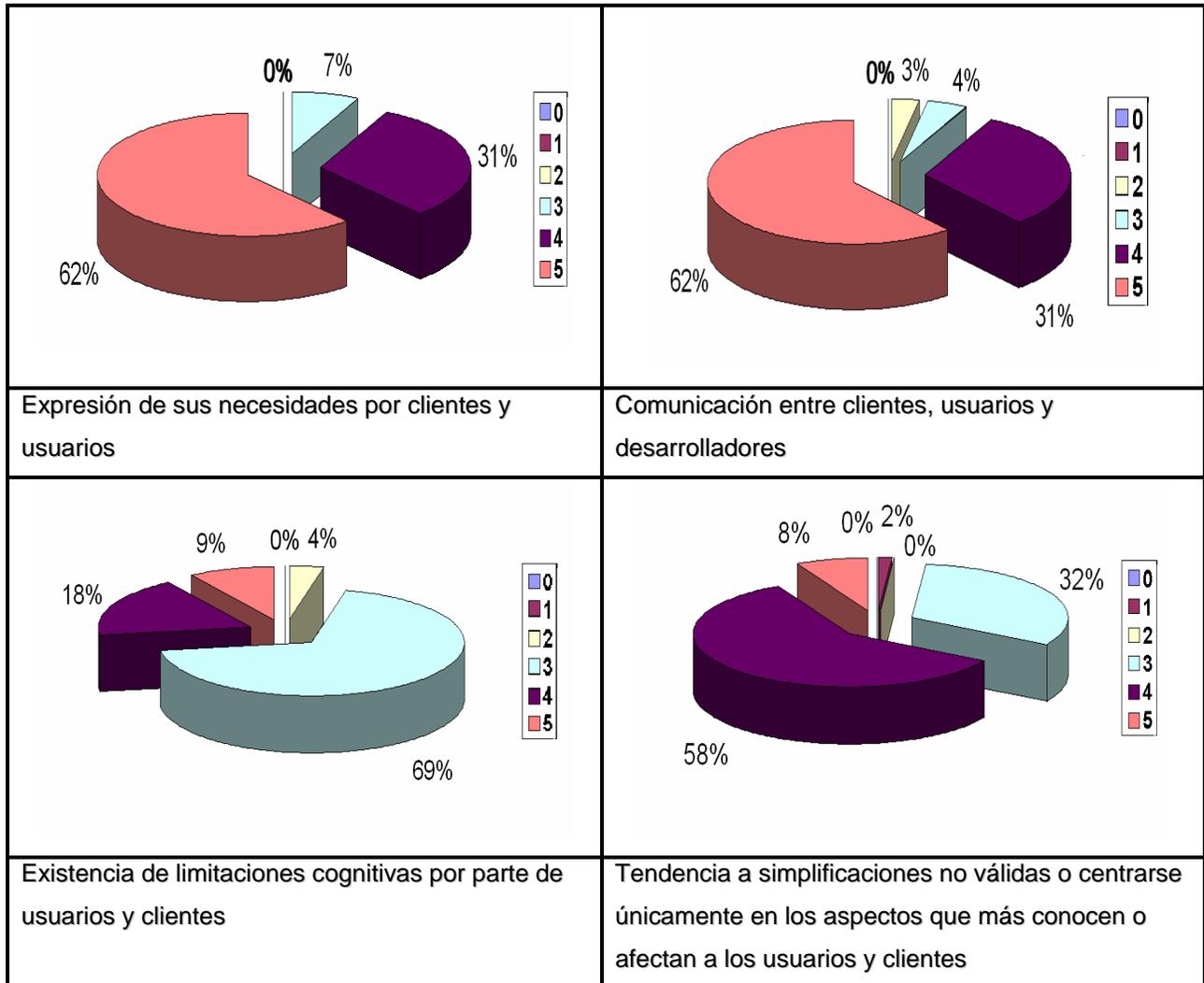
Anexo 3

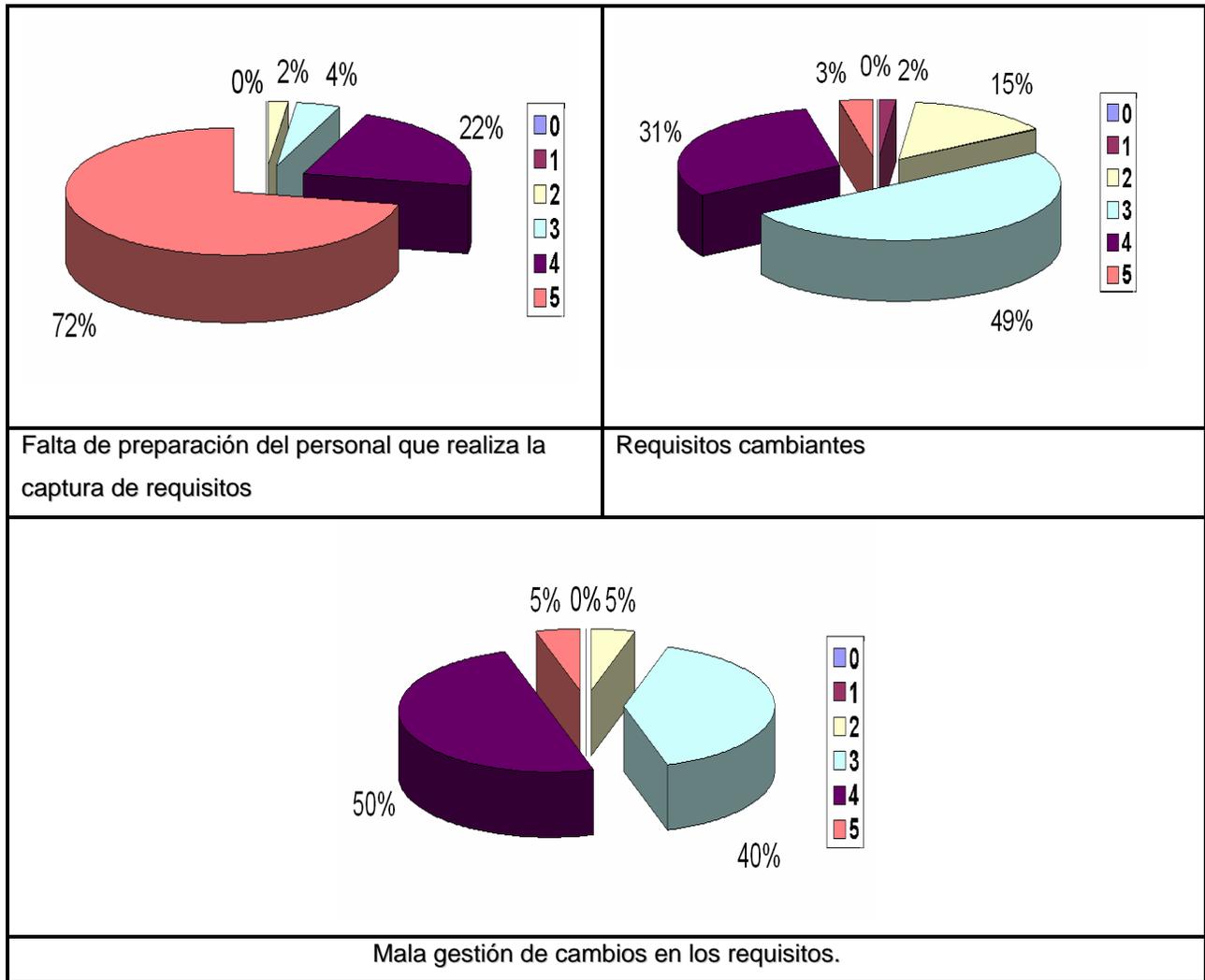




Grado de importancia de los problemas más frecuentes en la captura de requisitos. Resultados del grupo estudiantes.

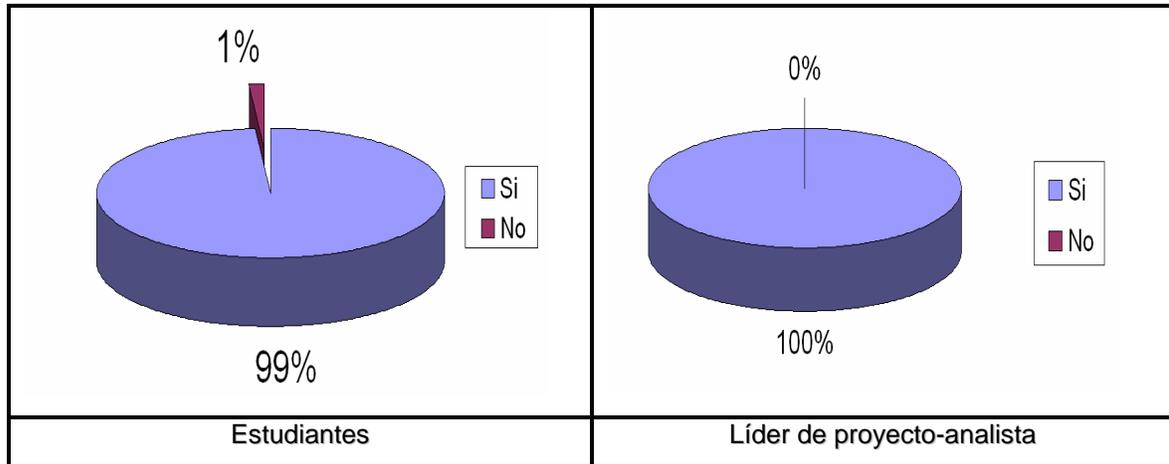
Anexo 4





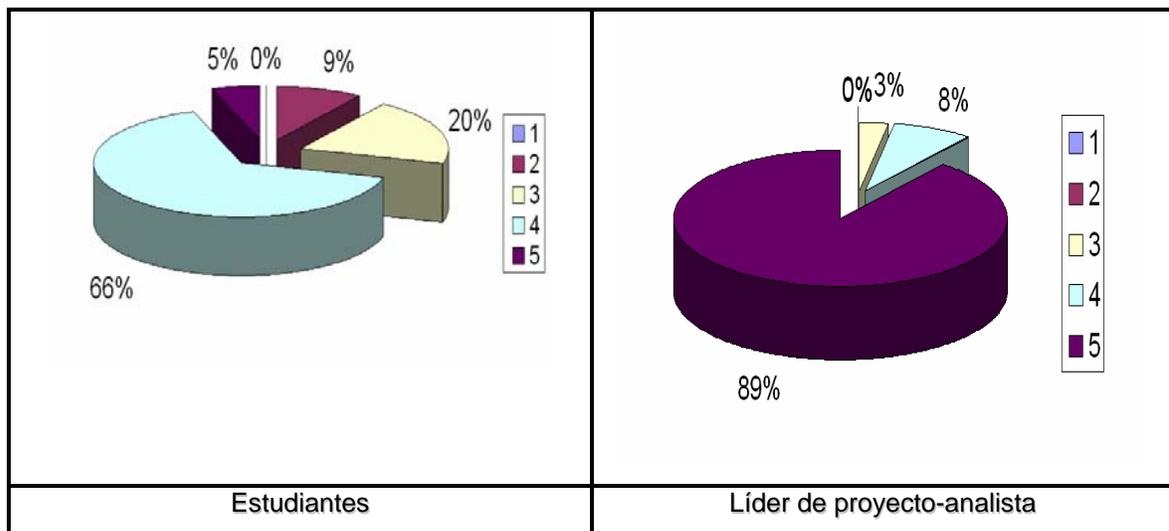
Grado de importancia de los problemas más frecuentes en la captura de requisitos. Resultados del grupo líder de proyecto-analista.

Anexo 5



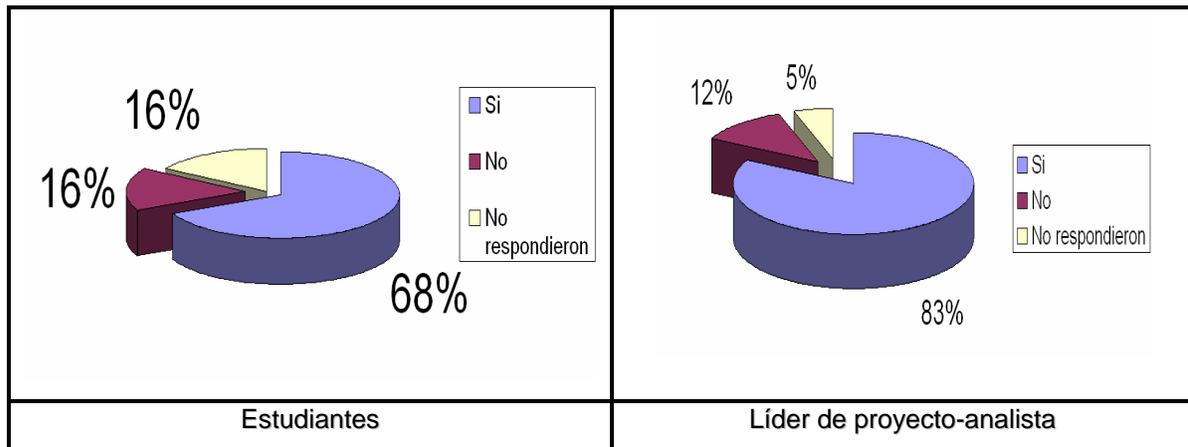
Necesidad de aumentar la capacitación sobre temas de la Ingeniería de requisitos.

Anexo 6



Grado de importancia que le asignan los entrevistados a la Ingeniería de requisitos.

Anexo 7



Realización de las tres etapas que conforman la Ingeniería de requisitos.

Anexo 8

<p>1. ¿En su opinión, la Ingeniería de Requisitos se debe aplicar a los proyectos de la UCI?</p> <p>___ Si ___ No ¿Por qué?:</p>
<p>2. ¿Cuáles de los problemas descritos reconoces a la hora de obtener los requisitos con calidad?</p> <p>___ Expresión de sus necesidades por clientes y usuarios</p> <p>___ Comunicación entre clientes, usuarios y desarrolladores</p> <p>___ Los requisitos cambian con el tiempo ya que a medida que los usuarios y clientes van conociendo sus necesidades y las posibilidades que le brinda la tecnología, surge la necesidad de reconsiderar decisiones anteriores.</p> <p>___ En problemas complejos, los requisitos tienden a ser cada vez más complejos.</p> <p>___ Otros</p>
<p>3. ¿Qué técnicas recomiendas para la captura de requisitos? Justifique.</p> <p>___ Entrevistas</p> <p>___ Tormenta de ideas (Brainstorming)</p>

<p>___ Cuestionarios y Checklists ___ Otros</p> <p>4. ¿Considera usted que es necesario cumplir con las tres etapas en la que se divide la Ingeniería de Requisitos? ___Si ___No ¿Por qué?</p> <p>a) En caso de que su respuesta sea negativa marque cuales de estas etapas son imprescindible para lograr la Ingeniería de Requisitos</p> <p>___ Captura de requisitos ___ Análisis de requisitos ___ Validación de requisitos</p>
<p>4. ¿Qué importancia le asigna a la Ingeniería de Requisitos dentro de la Ingeniería de software?</p> <p>___ Alta ___ media ___ ninguna</p>
<p>5. Haga una breve valoración de los beneficios o dificultades que pudiera presentar la aplicación del proceso propuesto en los proyectos productivos de la UCI.</p>

Encuesta realizada a expertos

Anexo 9

Experto	Graduado de	Años vinculados a la UCI	Eventos científicos	Experiencia profesional
Experto 1	Ingeniero- Máster en Ciencias Técnicas	3	Seminario sobre Inteligencia Empresarial y Dirección de Empresas, impartido en la Escuela de Administración de la Industria Básica en el año 1999. Aplicación de una Metodología de Dirección Integrada de Proyectos incluyendo el Sistema de Aseguramiento de la Calidad de los proyectos informáticos del Sistema Integrado de Gestión	Especialista del Sistema Automatizado de Control de Carga del Despacho de Carga de la Unión Eléctrica de Cuba, durante 4 años. Director de Informática de la Unión Eléctrica de Cuba durante 6 años. Responsable de una Fuerza de Tarea de Calidad de Software durante un año dentro del

			<p>Empresarial de la Unión Eléctrica.</p> <p>Aplicación del SIGEMETODO V1.0, Metodología cubana de Gestión de Proyectos Informáticos en Empresas Informáticas. Definición del Modelo de Procesos, Mapa de Procesos.</p> <p>Creación de TOT Model / Una herramienta ideal para la mejora de la Calidad del proceso de desarrollo de software en el área clave de Gestión de Requerimientos.</p>	<p>Ministerio de Informática y las Comunicaciones.</p> <p>Director de Informatización de la Universidad de las Ciencias Informáticas durante 1 año</p> <p>Especialista Principal en Calidad y Desarrollo de la Empresa de Desarrollo de Software para el sector de la Salud, denominada SOFTEL durante 1 año.</p> <p>Director de Calidad de la Empresa Nacional de Software, DESOFT, durante 8 meses.</p>
Experto 2	Ingeniería Informática	4	<p>Uciencia, SISOFT 2005 (Rep. Dominicana), SEPGLA 2005(México): Rup vs pmbok: practicas compatibles?</p> <p>El cierre de proyecto, la fase olvidada y La gestión de errores en el proceso de desarrollo de software en Informática 2007.</p>	Líder de proyecto Labora durante 4 años
Experto 3	Ingeniera informática	3	<p>En Uciencia: Estrategia organizativa del proyecto APS, Resultados de la aplicación de la Técnica por Encuentro de la Facultad asignatura ISW. Propuesta de sistema de gestión para los recursos humanos en la Facultad</p>	<p>Jefa de Dpto. de la especialidad y asesora de calidad de la Facultad 7. Profesora de IS y Metodología de la Investigación. 3 años de experiencia laboral</p>
Experto 4	Lic. Matemática	4	Desarrolló software base	Máster en Sistemas

			<p>para las minicomputadoras cubanas, incluyendo compiladores. Amplia experiencia en el desarrollo de aplicaciones sobre MS-DOS, UNIX y Windows. Actualmente participa en el desarrollo de equipos médicos de alta tecnología. Trabaja sobre Linux</p> <p>Premio de plata en concurso Centro Regional Enseñanza Informática, España. Condición de Mención y de Destacado en Fóruns Nacionales de Ciencia y Técnica. Premio Orlando Ramos por ponencia más técnica en evento TECBIOMED del ICID.</p>	<p>Digitales 1979. Investigador Auxiliar desde Mayo 1982. Profesor Titular Adjunto desde 1996. Trabaja desde 1973 en el Instituto Central de Investigación Digital.</p>
Experto 5	Ingeniero Informático	3	Uciencia: Técnica para la mejora de calidad en la gestión empresarial.	J' Polo Productivo Gestión Empresarial Ciencias Básicas Fac 2
Experto 6	Ingeniero informático	2	Convenciones de informática	Especialista de la calidad del software de la UCI
Experto 7	Ingeniero informático	3	Convenciones de informática	Asesor de gestión de software