UCI

UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 9

PRINCIPIOS PARA EL LEVANTAMIENTO DE REQUISITOS EN EL POLO SISTEMAS GEOLÓGICOS DE LA UCI

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

AUTORA: Alaraisy Pérez Fer.

TUTOR: Ing. Ramsés Ibarrola Suárez.

ASESORA: Lic. Yanisley Álvarez Arencibia

Ciudad de la Habana, Julio 2008

"Año 50 de la Revolución"

A Papi y a mi Cosita Linda...Enriquito.

RESUMEN

Hoy en día el proceso de desarrollo de software se ha convertido en un eslabón fundamental para el desarrollo de nuevas tecnologías; uno de los principales problemas que se detectan en este proceso está inmerso en al etapa de Ingeniería de Requerimientos (IR.), etapa de vital importancia para que un software sea desarrollado con la calidad debida. La Universidad de las Ciencias Informáticas (UCI), desde sus inicios ha estado vinculada al proceso de desarrollo de software, pero la falta de experiencia, la falta de personal calificado y la utilización de pocas técnicas para realizar la captura de requerimientos/requisitos, principalmente en la etapa de Obtención de Requerimientos, ha traído consigo retraso en la entrega de los productos. El proceso de levantamiento de requisitos en el Polo Sisternas Geológicos, el cual constituye objeto de estudio de esta investigación, no ha estado ajeno a esta situación; es por ello que en este trabajo de tesis se hace mención de un grupo de técnicas y metodologías, además de las ya utilizadas en el Polo, que formarán parte de la guía a proponer.

Este trabajo de tesis está estructurado en dos capítulos. El capítulo 1, se hace alusión a los principales conceptos y argumentos que esclarecen el objeto de estudio y en el capítulo 2 se describen las nuevas técnicas que formarán parte de la guía a utilizar por el Polo así como las ya existentes, además de una breve clasificación de dichas técnicas y la propuesta de una herramienta de código abierto para la administración de requerimientos.

PALABRAS CLAVES

Proceso, Software, Tecnologías, Ingeniería de Software, Ingeniería de Requerimientos, Guía, Técnicas.

ÍNDICE

ÍNDICE DE TABLAS Y FIGURAS

TABLAS:

FIGURAS:

INTRODUCCIÓN

Actualmente los sistemas computacionales están presentes en todas las áreas del conocimiento, por lo que los sistemas deben garantizar la confiabilidad de los datos y de la información. De la información producida por los sistemas, los usuarios toman decisiones y acciones que afectan directamente los beneficios de su negocio. Es preciso señalar que el desarrollo de software debe ser un proceso controlado y planeado, en donde se toman en cuenta las necesidades y sugerencias de los usuarios y clientes que solicitan lo que el sistema debe y no debe hacer.

Los errores en el software pueden causar daños o pérdidas en la información generada, y esto trasciende en las decisiones que tomen los usuarios del sistema. Las decisiones que se tomen con información errónea pueden producir bajas en el negocio, daños en el medio ambiente o hasta provocar la muerte de los usuarios en sistemas de alta criticidad.

La construcción de software requiere de la aplicación de un proceso de desarrollo que permita garantizar la calidad de los productos generados y que satisfagan las necesidades de las personas que lo usarán. La utilización de modelos de representación en el proceso de desarrollo de software permite reducir la probabilidad de riesgos y fallas, además garantizan la entrega en tiempo y forma del sistema al cliente. Los productos son entregados con calidad y dentro de los costos estimados.

La Ingeniería de Software es una disciplina que proporciona a los desarrolladores y creadores de software, un conjunto de procedimientos y técnicas para llevar a cabo la especificación, análisis, diseño, implementación, validación e incluso el mantenimiento de software. Con la aplicación de la Ingeniería de Software se pueden reducir riesgos de fallos en un sistema en desarrollo e incrementar la posibilidad de la entrega del producto en el tiempo estimado, con calidad y dentro de los costos presupuestados. Generalmente las etapas utilizadas en el desarrollo de software son: Ingeniería de Requerimientos, Diseño del Sistema, Implementación, Validación y Mantenimiento.

Una etapa inicial y muy importante dentro del proceso de la Ingeniería de Software, es la Ingeniería de Requerimientos, donde se lleva a cabo el proceso de descubrir, analizar, escribir y verificar los servicios y restricciones, del sistema de software que se desea producir; este proceso se realiza mediante la obtención, el análisis, la especificación, la validación y la administración de los requerimientos del software. Los requerimientos de software son las necesidades de los clientes, los servicios que los usuarios desean que proporcione el sistema de desarrollo y las restricciones en las que debe operar (Medina 2004). La importancia de esta etapa consiste en que, de la definición de los

requerimientos dependerá la definición de las etapas subsecuentes del desarrollo del software, es decir, que si no se descubren los requerimientos que se encuentran en el ambiente del sistema o son encontrados en una etapa avanzada del desarrollo del sistema, se tendrá que retroceder nuevamente a la etapa de requerimientos y esto provocaría cambios en el sistema y consecuentemente retraso en la entrega del mismo, además de alteración en los costos presupuestados o la cancelación del proyecto. Un caso peor, es que no se encontraran y especificaran todos los requerimientos del sistema en el proceso de desarrollo de software, lo cual produciría la entrega de un producto de software defectuoso o poco funcional. Por eso el resultado del proceso de la Ingeniería de Requerimientos constituye la base para el diseño, la implementación y la evaluación del software.

A través de los años se ha podido constatar que los requerimientos o requisitos son la pieza fundamental en un proyecto de desarrollo de software, ya que marcan el punto de partida para actividades como la planeación, básicamente en lo que se refiere a las estimaciones de tiempos y costos, así como la definición de recursos necesarios y la elaboración de cronogramas que será uno de los principales mecanismos de control con los que se contará durante la etapa de desarrollo. Además la especificación de requerimientos es la base que permite verificar si se alcanzaron o no los objetivos establecidos en el proyecto ya que estos son un reflejo detallado de las necesidades de los clientes o usuarios del sistema y es contra lo que se va a estar verificando si se están cumpliendo las metas trazadas. Es muy frecuente escuchar entre los conocedores del desarrollo de software (programas de computadoras), que un gran número de los proyectos de software fracasan por no realizar una adecuada definición, especificación, y administración de los requerimientos. Dentro de esa mala administración se pueden encontrar factores como la falta de participación del usuario, requerimientos incompletos y el mal manejo del cambio de los requerimientos. La Ingeniería de Requerimientos (IR) cumple un papel primordial en el proceso de producción de software, ya que se enfoca un área fundamental: la definición de lo que se desea producir. Su principal tarea consiste en la generación de especificaciones correctas que describan con claridad, sin ambigüedades, en forma consistente y compacta, las necesidades de los usuarios o clientes; de esta manera, se pretende minimizar los problemas relacionados por la mala gestión de los requerimientos en el desarrollo de sistemas (Arias 2005).

A pesar de los avances de la Ingeniería de Software, existe una gran cantidad de desarrolladores que no hacen uso de procedimientos y actividades que proporciona la Ingeniería de Software en el desarrollo de sistemas; esto provoca que un gran número de proyectos de software sean abandonados en pleno proceso de desarrollo o que sean entregados con retrasos y a costos mayores a los

estimados. A esta problemática se le une el hecho de que existen una gran cantidad de métodos y técnicas, que lejos de ser estándares, confunden más a los desarrolladores y dificultan la definición de los requerimientos.

La Universidad de las Ciencias Informáticas, es un programa de la revolución nacido bajo el calor de la batallas de ideas, que abrió sus puertas en el año 2002 organizando la distribución de sus estudiantes por facultades, las cuales fueron estructuradas y diseñadas para llevar paralelamente a la docencia un perfil que le permitiera formar Ingenieros Informáticos mucho más capacitados y capaces de desarrollar habilidades en otros campos relacionados con la especialidad, en aras del desarrollo económico del país y en beneficios de la sociedad, es por ello que desde los primeros años de estudio se vincula la docencia con la producción. Debido a la falta de experiencia y de personal capacitado para desarrollar dentro del proceso de la Ingeniería de Software, la Ingeniería de Requerimientos etapa de vital importancia para el desarrollo de software, ha traído como consecuencia, en muchos casos, productos de mala calidad o retraso en al entrega de los mismos. Uno de los principales factores que afecta el proceso de desarrollo de estos software, es el mal o poco uso de este tipo de ingeniería, debido a que no se le da la importancia requerida o se le asigna muy poco tiempo para su desarrollo, en muchos casos no se hace un análisis detallado de este proceso tan complejo trayendo consigo que a lo largo del desarrollo del software aparezcan nuevos requisitos y halla que empezar una y otra vez desde el levantamiento de requisitos; otro de los problemas que afecta este proceso es que no se aplican un gran número de técnicas y estándares establecidos para llevar a cabo la captura de todos los requisitos necesarios para el desarrollo de un producto con la calidad requerida (Ver Anexo 1).

El Polo de Sistemas Geológicos de nuestra universidad, como parte del grupo de desarrollo de software de la misma, no ha estado expenso a lo anteriormente planteado, la falta de personal capacitado y de experiencia (Ver Anexo 2), ha traído como consecuencia retraso en la entrega del producto, debido al mal o poco uso del proceso de captura de requerimientos.

Por lo expuesto anteriormente surge la necesidad de **perfeccionar el Proceso de Ingeniería de Requerimientos aplicado en el Polo Sistemas Geológicos de la Universidad de las Ciencias Informáticas (UCI).**

Para resolver el problema referenciado anteriormente se hizo necesario que el **objetivo general** de esta investigación esté dirigido hacia la elaboración de una guía para el desarrollo del proceso de levantamiento de requerimientos en el Polo Sistemas Geológicos de la Universidad de las Ciencias Informáticas (UCI).

Para darle cumplimiento a dicho objetivo fue necesario centrar la investigación en el proceso de levantamiento de requerimientos del Polo Sistemas Geológicos de la UCI, lo cual constituye el **objeto de estudio**.

La guía que será propuesta como solución de esta investigación tendrá su aplicación en el proceso de desarrollo de software del Polo Sistemas Geológicos de la Facultad # 9 de la UCI, el cual determina el campo de acción de la misma.

Para darle cumplimiento al objetivo trazado fue necesario llevar a cabo una serie de tareas, ellas son:

1- Hacer un estudio de las técnicas y actividades dentro del proceso de la Ingeniería de Requerimientos.

2- Realizar entrevistas a la dirección de Calidad UCI, y al líder del Polo Sistemas Geológicos.

3- Investigar cómo se desarrolla este proceso en los diferentes módulos del Polo Sistemas Geológico.

4- Resumir la información recuperada.

5- Elaborar una guía para el proceso de la Ingeniería de Requerimientos del Polo Sistemas Geológicos de la UCI.

El desarrollo exitoso de las tareas expuestas anteriormente dará cumplimiento a la hipótesis de trabajo de esta investigación: Si se aplica una guía en el proceso de la Ingeniería de Requerimientos de los software (módulos) del Polo Sistemas Geológicos de la Facultad #9 de la UCI, se logrará una mayor calidad en el proceso de desarrollo del software.

Para el desarrollo de la investigación se utilizaron métodos teóricos y empíricos. Dentro de los métodos teóricos fueron utilizados el hipotético deductivo, el de análisis y síntesis, y el histórico-lógico; y dentro de los métodos empíricos fueron utilizadas la entrevista y la observación, ya que en esta fase la elaboración de un buen instrumento determina en gran medida la calidad de los datos necesarios para dar respuesta al problema. A continuación se especifica el por qué de la selección de los mismos.

1- El hipotético deductivo ya que a partir de la hipótesis planteada y siguiendo las reglas lógicas de deducción se llega a nuevos conocimientos y predicciones, las que posteriormente son sometidas a verificaciones empíricas.

2- Entrevistas al personal de calidad UCI y al líder del Polo Sistemas Geológicos, para recopilar todo la información necesaria y conocer cómo se desarrolla el proceso de levantamiento de requerimientos en la UCI y en dicho Polo.

3- Observación, este método nos permite conocer la realidad mediante la percepción directa de los fenómenos y objetos, además puede utilizarse en compañía de otras técnicas y procedimientos como la entrevista y el cuestionario.

4- El Método Histórico, para investigar si existe un proceso planteado en la UCI para efectuar el levantamiento de requerimientos, y de ser así ver cómo es que este se lleva a cabo en el desarrollo de software del Polo Sistemas Geológicos.

5- El método de Análisis para comprender y enunciar los principios para la guía del proceso de levantamientos de requerimientos y el de Síntesis para plantear, describir y resumir el proceso que se llevará a cabo a partir de la investigación en el Polo Sistemas Geológicos.

Con el objetivo de lograr resultados que le dieran credibilidad a esta investigación fue necesario seleccionar una población, de la cual se extrajo una muestra que fue analizada.

Población: El conjunto de todos los proyectos productivos (módulos) del Polo Sistemas Geológicos de la UCI.

Muestra: Se tomó como muestra un módulo del Polo Sistemas Geológicos.

Unidad de Estudio: Se tiene como unidad de estudio en este caso a la muestra de la población, un módulo del Polo Sistemas Geológicos.

La selección de esta muestra se llevó a cabo haciendo uso de un muestreo no probabilístico, en este caso: el muestro intencional, ya que este tipo de muestreo posibilita escoger a los integrantes de la muestra, por lo que nos permite seleccionar explícitamente los elementos que son representativos o con posibilidad de brindar mayor información.

El trabajo que se presenta a continuación está estructurado en dos capítulos. El capítulo 1, contempla la fundamentación teórica de esta investigación, en el cual son expuestos los principales conceptos que contribuyen al mejor entendimiento del problema en cuestión, se especifican detalladamente todos los argumentos que esclarecen el objeto de estudio y se explican otras soluciones existentes.

En el capítulo 2, se realiza un análisis del proceso de levantamiento de requerimientos según distintas bibliografías y particularmente cómo se realiza dicho proceso en el Polo Sistemas

Geológicos, donde se hacen necesario llevar a cabo un conjunto de observaciones para realizar el planteamiento de un conjunto de principios y técnicas que contribuyan al desarrollo de la guía a proponer para el proceso de la Ingeniería de Requerimientos en el Polo Sistemas Geológicos. También se establece una clasificación de las técnicas de obtención de requerimientos así como una comparación de sus ventajas y desventajas. Además de una descripción detallada de la herramienta de código abierto para la administración de requisitos.

Para finalizar se presentan las conclusiones, las recomendaciones, las referencias bibliográficas, la bibliografía, los anexos y el glosario de términos.

CAPÍTULO 1

Fundamentación Teórica.

1.1 Introducción.

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 (*Government Account Office*, GAO) realizó un estudio (GAO, 1979) seleccionando 9 proyectos de desarrollo de software para el gobierno norteamericano cuyos contratos sumaban una cantidad de total de 6.800.000 dólares. De esta cantidad, sólo 119.000 dólares correspondían a un proyecto que se había utilizado tal como se había entregado. Dicho proyecto se trataba de 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. El resto de los 6.8 millones de dólares se distribuyeron, como puede verse en la figura 1 en la que pueden destacarse el enorme porcentaje de dinero invertido, en proyectos cancelados o no satisfactorios.

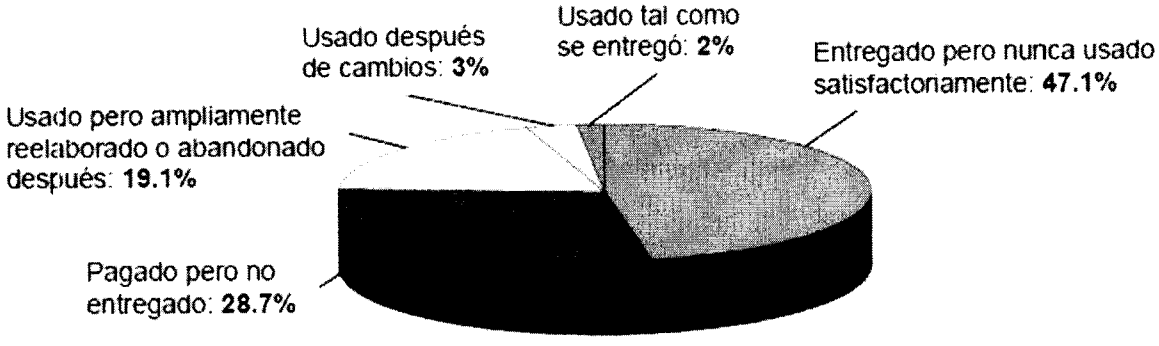

Figura 1: Resultados del Informe GAO.

En 1995, el Grupo Standish realizó un estudio (el informe CHAOS) mucho más amplio y significativo que el del GAO cuyos resultados, a pesar de haber pasado más de 25 años, no reflejaban una mejoría sustancial (TSG 1995). La siguiente figura (Figura 2) muestra el resultado de este estudio.

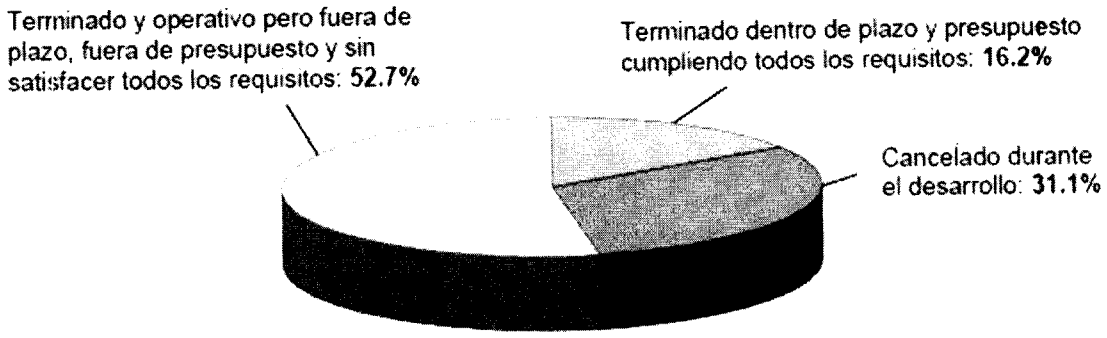

Figura 2: Resultados del Informe CHAOS.

Las encuestas realizadas a los directores de los proyectos que participaron en el estudio indicaron que, en su opinión, los tres principales factores de éxito eran: la implicación de los usuarios, el apoyo de los directivos y el enunciado claro de los requisitos; mientras que los tres principales factores de fracaso eran: falta de información por parte de los usuarios, especificaciones y requisitos incompletos, y especificaciones y requisitos cambiantes. Estos informes ponen de manifiesto el hecho de que, a pesar de que las herramientas para construir software han evolucionado enormemente, se sigue produciendo software que no es satisfactorio para los clientes y los usuarios. Esto indica que los principales problemas que han dado origen a la crisis del software residen en las primeras etapas del desarrollo, cuando hay que decidir las características del producto de software 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 (Roger 1997, Davis 1993), por lo que no es de extrañar que aquellos proyectos en los que no se determinan correctamente los requisitos y cambian frecuentemente durante el desarrollo, superen su presupuesto inicial. Todas estas circunstancias han convencido a la gran parte de la comunidad de la ingeniería del software de la necesidad, cada vez mayor, de una ingeniería de requerimientos (del Toro 2002).

Es precisamente la Ingeniería de Requerimientos el objetivo esencial sobre el cual gira la investigación realizada para el desarrollo del Capítulo 1, ya que constituye la base para el análisis y entendimiento del objeto de estudio que rige la misma, el cual se centra en el proceso de levantamiento de requerimientos del Polo de Sistemas Geológicos de la Universidad de las Ciencias Informáticas. Por tal motivo se hace necesario plantear los fundamentos teóricos que contribuyen a

esclarecer el origen y desarrollo del mismo, hasta llegar al planteamiento de posibles soluciones que son utilizadas en la actualidad.

En este capítulo se hace referencia a un grupo de conceptos, los cuales le brindarán al lector un mayor entendimiento de los temas abordados en el próximo capítulo, conceptos que están relacionados con el objeto de estudio de la investigación, entre ellos están: Ingeniería de Software, Ingeniería de Requerimientos y Requerimiento. Se describen además, los diferentes tipos de requisitos y sus características, las actividades que se llevan a cabo en proceso de la Ingeniería de Requerimientos, algunas herramientas para la administración de requisitos, así como otras metodologías existentes para llevar a cabo el levantamiento de requisitos.

1.2 Conceptos asociados al dominio del problema.

1.2.1 ¿Qué es la Ingeniería de Software?

Entre las diferentes definiciones que existen acerca de la Ingeniería de Software, podemos mencionar.

Ian Sommerville: Disciplina de la Ingeniería que comprende todos los aspectos de la producción de software desde las etapas iniciales de la especificación del sistema, hasta el mantenimiento de este después que se utiliza (Sommerville 2005).

Presman: Es el establecimiento y el uso de principios robustos de la ingeniería a fin de obtener económicamente software que sea fiable y que funcione eficientemente sobre máquinas reales (Presman 2002).

Lawrence: Disciplina que proporciona un conjunto de procedimientos y técnicas para la producción y desarrollo sistematizado, disciplinado y cuantificable de aplicaciones de software, satisfacen los requerimientos de funcionalidad y desempeño (Lawrence 2002).

Podemos resumir que la Ingeniería de Software comprende todo el proceso desde que surge la idea de crear el producto, hasta el mantenimiento del mismo una vez terminado el mismo.

1.2.2 Ingeniería de Requerimientos/Requisitos.

La ingeniería de requisitos es todavía una disciplina inmadura. De hecho no hay una definición universal. Se define como Ingeniería de requisitos:

(a) el proceso de estudiar las necesidades del usuario para llegar a una definición de requisitos de sistemas, hardware o software. (b) El proceso de estudiar y refinar los requisitos de sistemas, hardware o software (del Toro 2002).

Según el Glosario de IEEE 1997: Ingeniería de requerimientos es el proceso en el cual se transforman los requerimientos declarados por el cliente, ya sean hablados o escritos, a especificaciones precisas, no ambiguas, consistentes y completas del comportamiento del sistema, incluyendo funciones, interfaces, rendimiento y limitaciones. Es el proceso mediante el cual se intercambian diferentes puntos de vistas para recopilar y modelar lo que el sistema va a realizar (Richard 1997).

Loucopoulos: La Ingeniería de Requisitos trata con actividades en la cual intenta comprender las necesidades exactas de los usuarios del sistema software, para traducir tales necesidades en instrucciones precisas y no ambiguas las cuales podrían ser posteriormente utilizadas en el desarrollo del sistema (Loucopoulos 1995).

La ingeniería de requerimientos comprende todo el proceso de levantamiento y validación de requerimientos, para modelar lo que el usuario desea.

1.2.3 Requerimiento.

Los requerimientos del software se definen como las necesidades de los clientes, los servicios que el usuario desea que proporcione el sistema y las restricciones sobre las que el software debe operar (Medina 2004).

Según el glosario de la IEEE: (a) Una condición o necesidad de un usuario para resolver un problema o alcanzar un objetivo. (b) Una condición o capacidad que debe estar presente en un sistema o componentes de sistemas para satisfacer un contrato, estándar, especificación u otro documento formal. (c) Una representación documentada de una condición o capacidad como en (a) o (b). (Richard 1997).

Según Sommerville: Un requerimiento es simplemente una declaración abstracta de alto nivel de un servicio que debe proporcionar el sistema o una restricción de éste. (Sommerville 2005).

Un requerimiento no es más que una necesidad o una restricción con la cual el sistema debe cumplir, y que es establecida por el usuario o cliente.

1.3 Objeto de Estudio.

1.3.1 Descripción General.

En la introducción de esta tesis queda reflejado que el objeto de estudio por el cual se lleva a cabo esta investigación, es el proceso de levantamiento de requerimientos del Polo Sistemas Geológicos de la Universidad de las Ciencias Informáticas. Para un mayor entendimiento del mismo, se hará referencias a los conceptos más generales en los cuales está inmerso este proceso hasta caer en un análisis detallado de sus principales características mencionando también algunas de las herramientas que podemos utilizar para llevar a cabo este proceso.

1.3.1.1 Proceso de desarrollo de la Ingeniería de Software.

El proceso de desarrollo de software es una descripción de la construcción del software que contiene actividades organizadas de modo que en conjunto producen código probado. No existe una definición estándar de estas actividades y muchos autores le dan importancia a algunas más que a otras. Generalmente podemos clasificar estas actividades en:

1- **Ingeniería de requerimientos.**

2- Diseño del sistema.

3- Implementación del software.

4- Prueba o validación del software.

5- Mantenimiento

En la Ingeniería de Requerimientos, es necesario conocer la naturaleza del sistema, entender lo que desean los clientes, delimitar el alcance del sistema teniendo en cuenta el tiempo disponible, el presupuesto y el personal asignado. El producto generado de esta etapa, es un documento de especificación de requerimientos en donde se encuentran definidos todos los servicios requeridos del sistema y las restricciones sobre las que debe operar.

El Diseño del Sistema toma como base el documento de especificación de requerimientos, agrega detalles a cada requerimiento, identifica los subsistemas, describe la estructura interna

de los datos y las interfaces entre los componentes del sistema. El diseño del sistema utiliza varios modelos para la representación del sistema desde diferentes perspectivas y niveles de abstracción. El resultado de esta etapa es la especificación precisa de los algoritmos y estructuras de datos que van a implementarse.

En la implementación se lleva a cabo la codificación del sistema, se deben satisfacer los requerimientos de la manera que especifica el diseño detallado. El programador examina los documentos generados en las etapas anteriores para evitar inconsistencias entre los documentos. Se toman en cuenta los estándares de programación, así como los lenguajes de programación.

Las pruebas y validación se utilizan para demostrar que el sistema cumple con su especificación y satisface las necesidades del usuario. De cualquier forma, es necesario llevar a cabo un proceso de verificación por medio de inspecciones y revisiones desde la especificación de requerimientos hasta la puesta en marcha del sistema. El propósito de realizar las pruebas no es demostrar que una aplicación es factible, sino determinar con firmeza en qué parte no lo es. Las pruebas nos permiten mostrar la presencia de defectos en el sistema.

El mantenimiento de software consiste en las actividades realizadas sobre el sistema una vez entregado y puesto en marcha.

Como se observa, el resultado de la Ingeniería de Software consiste en mucho más que el código del sistema, es decir, incluye planes, informes, documentos e inclusive programas llamados prototipos que son desechados después de lograr el objetivo por lo que fueron creados.

1.3.1.2 El proceso de la Ingeniería de Requerimientos.

El proceso de la Ingeniería de Requerimientos, involucra a todas las actividades necesarias para crear y mantener el documento de requerimientos del sistema. Para realizar este proceso, es necesario previamente obtener un análisis de factibilidad que indique si es factible continuar con el desarrollo del sistema, además de conocer las necesidades del cliente para tener un contexto general del sistema.

Las actividades que se definen en el proceso de la Ingeniería de Requerimientos son las siguientes:

1- Obtención de los requerimientos.

2- Análisis de requerimientos.

3- Especificación de requerimientos.

4- Validación de documento de especificación.

5- Administración de requerimientos.

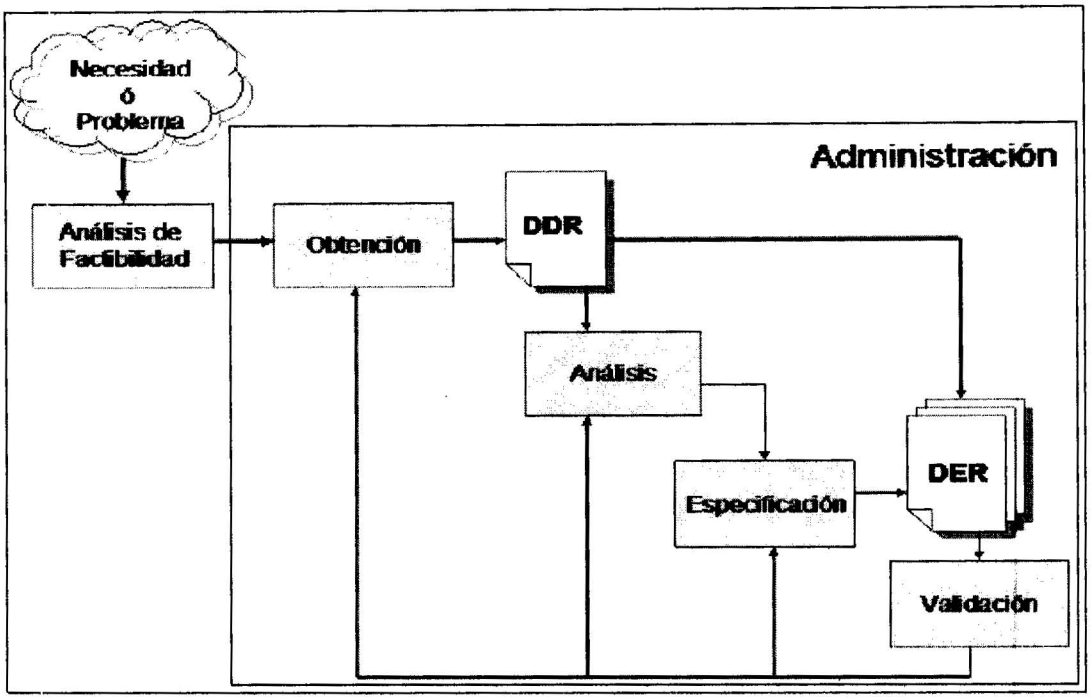

Figura 3: El proceso de la Ingeniería de Requerimientos en el Modelo de Cascada.

En la figura 3 se muestra la interacción que existe entre las diferentes actividades del proceso de Ingeniería de Requerimientos utilizando el paradigma de cascada (Anexo 3), en donde cada actividad tiene un regreso a la actividad anterior. Se parte del hecho de que existe una definición general del problema y de un análisis de factibilidad que indica la continuación del desarrollo del software. De esta manera, el proceso de la Ingeniería de Requerimientos inicia con la actividad de la obtención de requerimientos, mediante la revisión de la información existente en manuales y documentos y entrevistas a los usuarios para descubrir y analizar los requerimientos. La especificación y validación son actividades que pueden ser repetidas cuantas veces sean necesarias hasta lograr el nivel de refinamiento en que se desean especificar los requerimientos. Finalmente cada cambio en los requerimientos, es administrado en versiones del Documento de Especificación de Requerimientos.

1.3.1.3 Actividades del Proceso de Ingeniería de Requerimientos.

El objetivo de la Ingeniería de Requerimientos es obtener un documento formal donde se especifiquen las características que debe cumplir el sistema que se va a desarrollar. Estas características y restricciones son definidas en común acuerdo por el cliente y los usuarios del sistema. La especificación de los requerimientos debe cumplir con ciertas características de calidad, como las siguientes: entendibles, necesarios, consistentes, no ambiguos, completos, verificables, correctos, reales, rastreables y modificables (Medina 2004).

En cada una de las actividades del proceso de la Ingeniería de Requerimientos se llevan a cabo tareas específicas utilizando técnicas de requerimientos, modelos de especificación y herramientas para la administración del documento de requerimientos.

1- Obtención de Requerimientos.

En esta actividad se determina el domino de la aplicación, se especifican los servicios que deben proveer el sistema, la funcionalidad requerida del sistema, y las restricciones de hardware y software. Es indispensable la participación de los usuarios y clientes para la identificación de los requerimientos del sistema.

Se debe obtener como resultado de esta actividad, un **Documento de Definición de los Requerimientos (DDR),** donde se define las necesidades inicialmente encontradas, esto no significa que estos requerimientos sean los definitivos, pueden ser agregados nuevos requerimientos conforme se vayan encontrando o incluso los requerimientos establecidos pueden modificarse o eliminarse. Para lograr la obtención de los requerimientos se definieron las siguientes tareas:

1- **Comprender el problema que se está resolviendo,** es necesario estudiar el dominio o entorno en el que el sistema va a operar.

2- **Buscar y recolectar información** de manuales de información y mantenimiento, de manuales organizacionales y políticas de operación.

3- **Definir los límites y restricciones del sistema,** para determinar con exactitud qué es lo que el sistema va hacer y también especificar lo que no va hacer.

4- **Identificar a las personas o usuarios interesados por el sistema,** ya que ellos conocen el medio ambiente en que operará el sistema y pueden ayudar describiendo sus necesidades.

5- **Recolectar y clasificar requerimientos,** de esta manera los desarrolladores pueden iniciar definiendo un bosquejo del sistema, su funcionamiento básico y estableciendo el alcance del sistema.

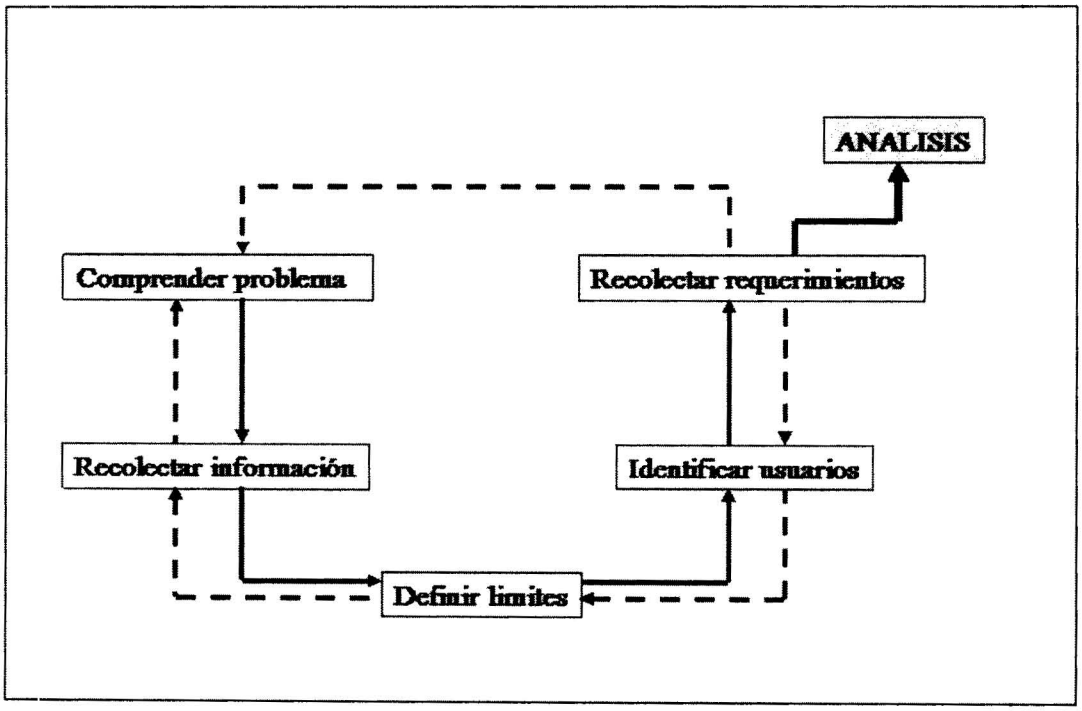

Figura 4: Fase #1. El proceso de Obtención de Requerimientos.

El desarrollo de estas tareas en la obtención de requerimientos es realizado secuencialmente, pero es necesario que cada tarea pueda regresarnos a la anterior, sobre todo si no se descubre la información necesaria en el primer recorrido del diagrama. La figura 4 muestra esta relación. La salida de esta actividad nos conduce hacia el análisis de requerimientos.

En principio, parece bastante simple, pero en realidad es un proceso bastante complicado, Pressman, en su quinta edición (Pressman 2002), identifica una serie de problemas que nos ayudan a comprender por qué la obtención de los requisitos es costosa.

1- **Problemas de alcance:** El límite del sistema está mal definido o los detalles técnicos innecesarios, que han sido aportados por los clientes/usuarios, pueden confundir más que clarificar los objetivos del sistema.

2- **Problemas de comprensión:** Los clientes/usuarios no están completamente seguros de lo que necesitan, tienen una pobre comprensión de las capacidades y limitaciones de su entorno de computación, no existe un total entendimiento del dominio del problema, existen dificultades para comunicar las necesidades al ingeniero del sistema, la omisión de información por considerar que es obvia, especificación de requisitos que están en conflicto con las necesidades de otros clientes/usuarios, o especificar requisitos ambiguos o poco estables.

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

Para ayudar a solucionar estos problemas Pressman sugieren un conjunto de actuaciones para la obtención de requisitos, los cuales quedan descritos en las siguientes tareas:

1- Valorar el impacto en el negocio y la viabilidad técnica del sistema propuesto.

2- Identificar las personas que ayudarán a especificar requisitos y contrastar su papel en la organización.

3- Definir el entorno técnico (arquitectura de computación, sistemas operativos, necesidades de telecomunicaciones) en el sistema o producto a desarrollarse e integrar.

4- Identificar restricciones del dominio (características específicas del entorno del negocio en el dominio de la aplicación) que limiten la funcionalidad y rendimientos del sistema o producto a construir.

5- Definir uno o más métodos de obtención de requisitos (entrevistas, grupos de trabajo, equipos de discusión).

6- Solicitar la participación de muchas personas para que los requisitos se definan desde diferentes puntos de vista, asegurarse de identificar lo fundamental de cada requisito registrado.

7- Identificar requisitos ambiguos como candidatos para el prototipo.

8- Crear escenarios de uso para ayudar a los cliente/usuarios a identificar mejor los requisitos funcionales.

El resultado alcanzado como consecuencia de la identificación de requisitos variará dependiendo del tamaño del sistema o producto a construir; una vez recopilados los requisitos, el producto obtenido configura la base del análisis de requisitos.

2- Análisis de Requerimientos.

Los requerimientos definidos en el documento de definición son analizados detalladamente por el equipo de desarrollo y negociados con el cliente y usuarios, para decidir los requerimientos que serán aceptados y definirlos de manera conjunta con el fin de homogenizar su interpretación.

El análisis del **Documento de Definición de Requerimientos** se lleva a cabo mediante técnicas de revisión y verificación de los criterios de calidad de cada requerimiento definido, este estudio es realizado por los desarrolladores, clientes y usuarios.

En el análisis se llevan a cabo las siguientes actividades:

1- **Priorizar los requerimientos** debido a que pueden existir requerimientos más importantes que otros para los clientes y usuarios, por lo que deben ser clasificados e implementados de acuerdo a su prioridad en el sistema.

2- **Encontrar dependencias entre requerimientos y etiquetar los requerimientos** con un identificador único, con el fin de poderlos identificar o rastrear en el futuro.

3- **Resolver conflictos entre los requerimientos,** se pueden encontrar conflictos entre requerimientos mediante la revisión de los criterios de calidad que debe cumplir cada requerimiento del sistema.

4- **Negociar con flexibilidad** con los demás elementos del tipo que interviene en el proceso de desarrollo de software, para homogenizar su comprensión y de ésta forma tanto desarrolladores como usuarios tengan la misma interpretación al momento de leer el documento de requerimientos.

El resultado del análisis es el **Documento de Definición de Requerimientos (DDR),** donde se encuentran descritos los requerimientos iniciales del sistema que se va a desarrollar en lenguaje natural; este documento sirve como punto de partida hacia la siguiente actividad del proceso de la Ingeniería de Requerimientos, por lo que es necesario que no existan requerimientos en conflictos y estén bien escritos. La figura 5 describe este proceso, muestra la interacción de las tareas realizadas y la salida de esta actividad es la entrada a la actividad de especificación de requerimientos.

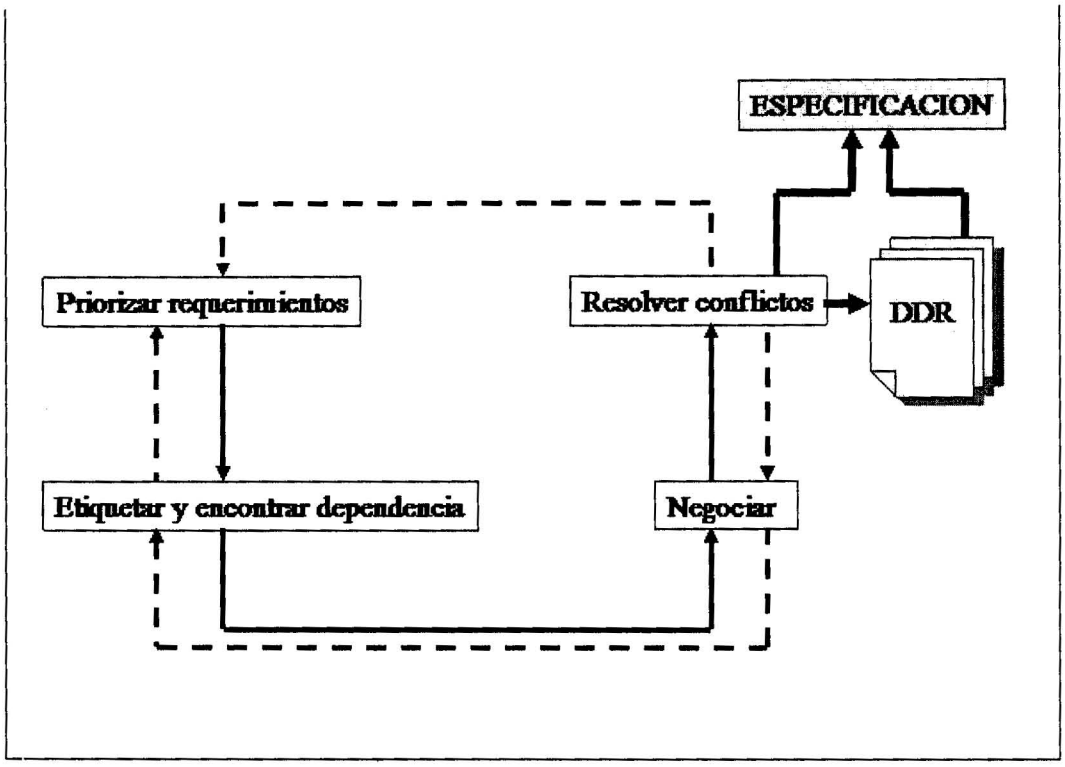

Figura 5: Fase #2. El proceso de Análisis de Requerimientos.

3- Especificación de Requerimientos.

En esta actividad se obtiene como resultado el **Documento de Especificación de Requerimientos (DER).** Este documento contiene la descripción precisa de los requerimientos, incluye modelos de representación que agregan detalles al sistema mostrándolo desde diferentes perspectivas.

Para el desarrollo de esta actividad se realizan las siguientes tareas:

1- **Especificar los requerimientos funcionales y no funcionales,** ambos requerimientos debe ser descrito a detalles para su mejor comprensión.

2- **Determinar el tipo de estándar** a utilizar para la definición del documento de especificación de requerimientos (DER). Se define el esquema del contenido del DER en base al estándar definido por la IEEE.

3- **Elegir la herramienta de especificación,** en caso de utilizar alguna para automatizar el proceso.

4- **Utilizar modelos y diagramas** para explicar con mayor detalle y diferentes perspectivas el comportamiento del sistema.

5- **Utilizar cualquier información de soporte o guía** para etapas posteriores y que amplíen la comprensión del sistema.

Todas estas tareas van encaminadas a obtener el Documento de Especificación de Requerimientos, en donde se especifican las funciones, restricciones o características del sistema en desarrollo. Es necesario contar con toda la información disponible para formar un documento completo y que sirva como base de partida hacia las otras etapas del desarrollo del sistema. En la figura 6 se muestran las actividades de la especificación de los requerimientos del sistema.

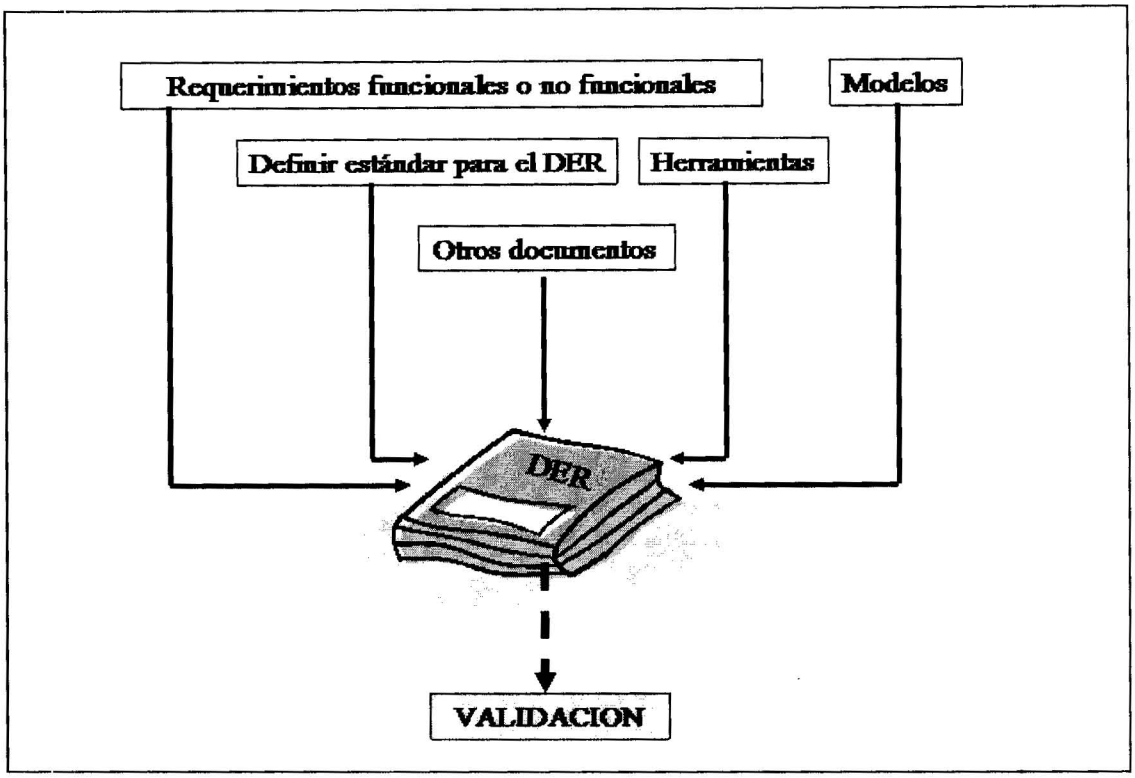

Figura 6: Fase #3. El proceso de Especificación de Requerimientos.

4- Validación de requerimientos.

Según Pressman (Pressman 2002) la validación de requisitos examina las especificaciones para asegurar que todos los requisitos del sistema han sido establecidos sin ambigüedad, sin

inconsistencias, sin omisiones, que los errores detectados hayan sido corregidos, y que el resultado del trabajo se ajusta a los estándares establecidos para el proceso, el proyecto y el producto. La validación permite detectar problemas en el documento de requerimientos, y se lleva a cabo verificando cuidadosamente la consistencia y completitud del DER, antes de que sea usado como base en etapas posteriores del proceso de desarrollo del sistema. Es un proceso importante debido a que permite detectar errores en el documento de requerimientos, de forma que pueden ser corregidos a tiempo. Estos errores si no son descubiertos en esta actividad pueden conducir a costos excesivos, ya que producen que se repita el trabajo cuando los errores sean descubiertos en etapas posteriores del desarrollo del sistema o en caso drástico, si llegan a ser detectados cuando el sistema esté en servicio.

Los criterios que son revisados para validar el documento de especificación de requerimientos son los siguientes:

1- **Verificar validez**: Los requerimientos satisfacen a las necesidades de los usuarios y son necesarios.

2- **Verificar consistencia**: Los requerimientos en el documento no deben contradecirse, ya que esto generaría conflictos en el momento de implementarse.

3- **Verificar integridad**: Es decir, deben estar todos los requerimientos que describan todas las funciones y las restricciones propuestas por el usuario del sistema.

4- **Verificar realismo**: Asegurar que los requerimientos puedan implementarse con la tecnología existente, considerando también el presupuesto y la calendarización del desarrollo del sistema.

5- **Generar casos de pruebas**: para los requisitos.

Todos estos criterios deben ser orientados hacia la aplicación en el documento de especificación de requerimientos, a fin de lograr su certificación y validación. Esta actividad es importante y debe tomarse en cuenta en cualquier proceso de desarrollo de software porque garantiza que lo expresado en el documento, describe las características del sistema que se va a desarrollar y puede servir como base para las etapas de diseño, implementación y pruebas del software. En la siguiente figura (Figura 7) se muestra esta actividad.

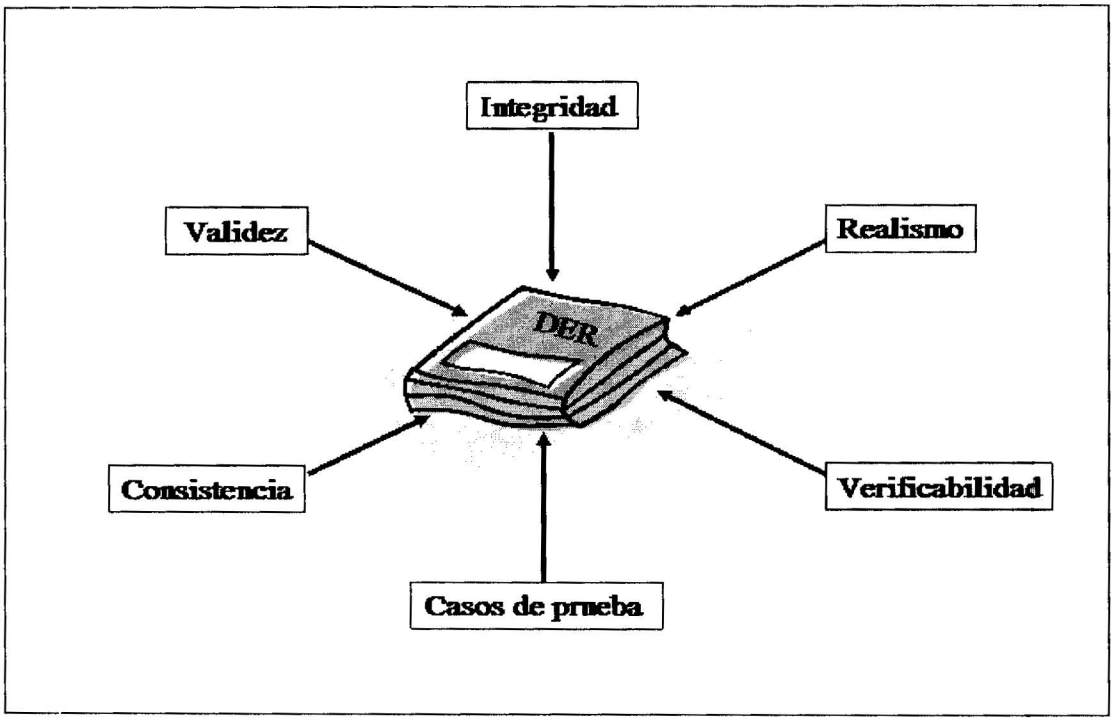

Figura 7: Fase #4. El proceso de Validación de Requerimientos.

5- Administración de requerimientos.

La administración de los requerimientos describe el proceso de comprender y controlar los cambios en los requerimientos del sistema. Esta actividad comienza en cuanto existe la primera versión del documento de requerimientos ó cuando se planea darle mantenimiento a un sistema existente. Específicamente, en esta actividad se requiere lo siguiente:

1- **Administrar los cambios de los requerimientos:** Esto implica el análisis del problema y su propuesta de cambio, valorar el costo del cambio propuesto y finalmente implementar los cambios en los documentos que lo requieren.

2- **Establecer políticas de rastreo:** Es necesario conocer el origen de los requerimientos o los efectos que tendrán los cambios en los requerimientos relacionados.

3- **Control de versiones del documento de requerimientos:** cuando implique algún cambio para referencias futuras.

1.3.1.4 Importancia de la Ingeniería de Requerimientos.

Los principales beneficios que se obtienen de la Ingeniería de Requerimientos son:

- **Permiten gestionar las necesidades del proyecto en forma estructurada:** Cada actividad de la IR consiste en una serie de pasos organizados y bien definidos.
- **Mejora la capacidad de predecir cronogramas de proyectos, así como sus resultados:** La IR proporciona un punto de partida para controles subsecuentes y actividades de mantenimiento, tales como estimación de costos, tiempo y recursos necesarios.
- **Disminuye los costos y retrasos del proyecto:** Muchos estudios han demostrado que reparar errores pues un mal desarrollo no descubierto a tiempo, es sumamente caro.
- **Mejora la calidad del software:** La calidad en el software tiene que ver con cumplir un conjunto de requerimientos (funcionalidad, facilidad de uso, confiabilidad, desempeño, etc.).
- **Mejora la comunicación entre equipos:** La especificación de requerimientos representa una forma de consenso entre clientes y desarrolladores. Si este consenso no ocurre, el proyecto no será exitoso.
- **Evita rechazos de usuarios finales:** La ingeniería de requerimientos obliga al cliente a considerar sus requerimientos cuidadosamente y revisarlos dentro del marco del problema, por lo que se involucra durante todo el desarrollo del proyecto.

La ingeniería de requerimientos es una de las disciplinas de la Ingeniería de Software de mayor importancia pues la misma depende de una intensa comunicación entre clientes y analistas de requerimientos. La Ingeniería se encarga de establecer y mantener un acuerdo en qué el sistema debe hacer. Proporciona al equipo de desarrollo un entendimiento de los requisitos, hasta definir los límites del sistema. Además se controlan los cambios a los requisitos.

1.3.1.5 Requerimientos.

Tipos de requerimientos.

Los requerimientos de software son las descripciones de los servicios y restricciones del sistema en desarrollo. Existen diferentes maneras de clasificarlos, generalmente se le agrupan de acuerdo a la audiencia a quienes van dirigidos y las características del sistema.

Tipos de requerimientos de acuerdo a la audiencia.

Existen diferentes niveles de descripción de requerimientos, que permiten orientar los requerimientos a diversos usuarios. Es distinto explicar los alcances de un sistema nuevo a clientes usuarios que no van a utilizar el sistema, pero que de alguna forma están involucrados en el desarrollo del sistema, que explicárselos a los desarrolladores o arquitectos de software que se encargarán de la implementación del sistema. Por tal motivo, se requiere de diferentes niveles de descripción y por lo tanto de diferentes tipos de descripción de requerimientos. (Sommerville 2005).

1- **Los Requerimientos del Usuario.** Son expresados en lenguaje natural utilizando diagramas fáciles de comprender, de los servicios que espera que el sistema provea y de las restricciones bajo las cuales el sistema debe funcionar, también son conocidos como *Requerimientos del Cliente o Requerimientos C.*

2- **Los Requerimientos del Sistema.** En estos requerimientos se establecen con más detalles los servicios y restricciones del sistema. También se les conoce como *especificación funcional, Requerimientos del Desarrollador o Requerimientos D.*

3- **La Especificación del Diseño del Software.** Es una descripción abstracta del diseño del software que se utiliza como una base para un diseño e implementación detallados.

Los distintos niveles de descripción de requerimientos son necesarios debido a que informan de manera clara a diferentes tipos de usuarios. Por ejemplo, los requerimientos abstractos de alto nivel más conocidos como **requerimientos de usuario,** como están expresados en un lenguaje claro, fácil y sin detalles funcionales, están dirigidos a clientes administradores, usuarios finales del sistema, clientes ingenieros, contratistas administradores y arquitectos del sistema. Por otro lado, los requerimientos que describen lo que el sistema debe hacer ó **requerimientos del sistemas** son dirigidos para usuarios técnicos del sistema, clientes ingenieros, arquitectos del sistema y desarrolladores del software. Debido a su especificación funcional dentro del sistema, es necesario de un cierto grado de conocimiento técnico o especialización en el área. Y por último, la **especificación del diseño de software** está dirigida a clientes ingenieros, arquitectos del sistema y desarrolladores del software.

Tipo de requerimientos de acuerdo a sus características.

Esta clasificación se realiza en función de la naturaleza de las características del sistema que se desarrolla, generalmente los requerimientos del sistemas de software se clasifican en funcionales, o no funcionales.

1- **Requerimientos funcionales.** Los requerimientos funcionales son declaraciones de los servicios que proveerá el sistema y comprenden la interacción entre el sistema y su ambiente, la reacción a entradas particulares de datos y su comportamiento en condiciones específicas. En algunos casos también declaran lo que el sistema no debe hacer.

2- **Requerimientos no funcionales.** Los requerimientos no funcionales, son especificaciones de las restricciones de los servicios o funciones ofrecidas por el sistema, restricciones sobre el sistema que limitan nuestras elecciones en la solución a un problema.

Los **requerimientos no funcionales** no se refieren directamente a las funciones específicas que entrega el sistema, sino a sus propiedades emergentes. Existen muchas maneras de clasificarlos, Sommerville (Sommerville 2005) propone la siguiente clasificación (Ver Figura 8).

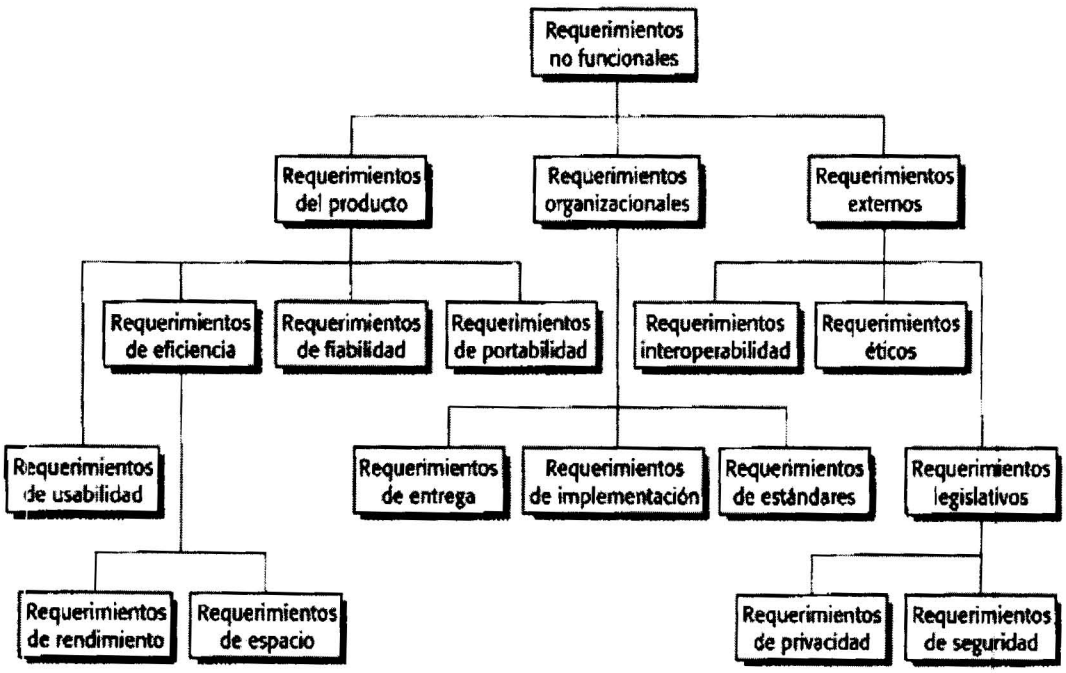

Figura 8: Tipos de Requerimientos no Funcionales.

2.1- Requerimientos del producto: Especifican el comportamiento del producto, por ejemplo el desempeño en la rapidez de ejecución del sistema, espacio de memoria requerida, la fiabilidad ó tasa de fallas para que el sistema sea aceptable, portabilidad y usabilidad.

2.2- Requerimientos organizacionales: Se derivan de las políticas y procedimientos existentes en la organización del cliente y desarrollador. Son ejemplos de estos los estándares que deben utilizarse en los procesos, lenguajes de programación en la implementación o métodos de diseño, los requerimientos que se especifican cuando se entrega el producto y documentación.

2.3- Requerimientos externos: Se refieren aquellos que se derivan de los factores externos al sistema y de su proceso de desarrollo. Por ejemplo, los requerimientos de interoperabilidad que establecen la manera en que el sistema interactúa con otros sistemas de la organización, requerimientos legales que aseguran que el sistema opere dentro de la ley y los principios éticos que aseguran que el sistema será aceptado por el usuario y el público en general.

Los requerimientos no funcionales tienen que ver con características que de una u otra forma pueden limitar el sistema, como por ejemplo, el rendimiento (en tiempo y espacio), interfaces de usuario, fiabilidad (robustez del sistema, disponibilidad de equipo), mantenimiento, seguridad, portabilidad, etc.

Algunas de las **propiedades** de los **requerimientos no funcionales** que hacen al producto atractivo, usable, rápido o confiable, son las siguientes:

- **Requerimientos de apariencia o interfaz externa:**

  Ejemplo: Muy legible, simple de usar, profesional o tipo ejecutivo.

- **Requerimiento de usabilidad:**

  Ejemplo: Facilidad de uso por personas que hablen otros idiomas distintos al del país donde el producto fue creado, accesibilidad para personas discapacitadas, consistencia en la interfaz de usuario, documentación de usuario.

- **Requerimientos de rendimiento:**

  Ejemplo: Velocidad de procesamiento o cálculo, eficiencia, disponibilidad, tiempo de respuesta.

- **Requerimientos de soporte:**

Ejemplo: Adaptabilidad, mantenimiento.

- **Requerimientos de Portabilidad:**

  Ejemplo: El producto podrá ser usado bajo el sistema operativo Linux.

- **Requerimientos de seguridad:**

  1- **Confidencialidad**: La información manejada por el sistema está protegida de acceso no autorizado y divulgación.

  2- **Integridad:** La información manejada por el sistema será objeto de cuidadosa protección contra la corrupción y estados inconsistentes.

  3- **Disponibilidad:** Significa que los usuarios autorizados se les garantizará el acceso a la información y que los dispositivos o mecanismos utilizados para lograr la seguridad no ocultarán o retrasarán a los usuarios para obtener los datos deseados en un momento dado.

- **Requerimientos de confiabilidad:**

  Frecuencia y severidad de los fallos, protección contra fallos, recuperación, predicción de fallos, tiempo medio entre fallos.

- **Requerimientos de software:**

  Ejemplo: Sistema Operativo Windows 95 o Superior; máquina virtual de java versión 1.3 o superior; etc.

- **Requerimientos de hardware:**

  Ejemplo: Se requiere disponer de un MÓDEN estándar o una tarjeta digitalizadora de video, etc.

Otros tipos de requerimientos.

Existen otras clasificaciones de los requerimientos en la bibliografía de Ingeniería de Requerimientos y que permite definir de manera específica las necesidades de los usuarios, estos tipos de requerimientos son los siguientes:

1- **Requerimientos de dominio:** Los requerimientos de dominio, son requerimientos que provienen del dominio de aplicación del sistema y reflejan las características de este dominio.

2- **Requerimientos de datos:** Los requerimientos de datos definen las estructuras de datos requeridas en el sistema.

3- **Requerimiento de interfaz:** Define las características y parámetros de la comunicación del sistema a desarrollar con otros sistemas dentro de la empresa, o incluso de los subsistemas.

Los requerimientos evolucionan o tienden a cambiar de acuerdo a las necesidades del negocio o el desarrollo del software puede llevar tiempo sobre todo si se trata de sistemas grandes. Desde esta perspectiva los requerimientos se clasifican en:

1- **Requerimientos duraderos:** Estos tipos de requerimientos son relativamente estables debido a que se derivan de la actividad principal de la organización y porque están relacionados directamente con el dominio del sistema.

2- **Requerimientos volátiles:** Estos requerimientos sufrirán cambios probablemente durante el desarrollo del sistema o después de que este se haya puesto en operación.

Entre los requerimientos volátiles podemos encontrar:

1- **Requerimientos mutantes:** Son requerimientos que cambian debido a los cambios del ambiente en el que opera la organización.

2- **Requerimientos emergentes:** Son requerimientos que surgen al incrementarse la comprensión del cliente en el desarrollo del sistema. El proceso de diseño puede producir nuevos requerimientos emergentes.

3- **Requerimientos consecutivos:** Estos requerimientos, son el resultado de la introducción del sistema. Esta introducción puede cambiar los procesos de la organización y abrir nuevas formas de trabajar que generan nuevos requerimientos del sistema.

4- **Requerimientos de compatibilidad:** Son requerimientos que dependen de sistemas particulares o procesos de negocios dentro de la organización.

A pesar de de las diferentes características que nos brindan los requerimientos, existen dificultades para recolectar los requisitos, las cuales no nos permiten elegir los requerimientos con la calidad necesaria; ya que estos pueden relacionarse unos con otros y a su vez con otras partes del proceso. Pero aun así, se plantea que sin el levantamiento de requisitos no se podrían desarrollar procesos que son de vital importancia para el desarrollo de software. Los

requisitos constituyen el enlace entre las necesidades reales de los clientes, usuarios y otros participantes vinculados al sistema.

Características de calidad de los requerimientos.

Los requerimientos no sólo describen el flujo de información de entrada y salida del sistema, y la transformación de los datos dentro del sistema, sino también las restricciones de rendimiento del sistema. De la misma manera, los requerimientos pueden servir a los siguientes propósitos:

1- Permiten que los desarrolladores expliquen cómo han entendido lo que el cliente necesita del sistema.

2- Indican a los diseñadores que funcionalidad y características va a tener el software resultante.

3- Indican al equipo de prueba las demostraciones o pruebas que realizarán para convencer al cliente de que el sistema que se le entrega es lo que ha pedido.

La calidad de los requerimientos está dada de acuerdo a las características que tengan y que permitan garantizar a los desarrolladores y clientes su entendimiento y utilización. Estas características son:

1. **Entendible:** Un requerimiento es entendible si es fácil de leer y entender. Su redacción debe ser simple y clara para aquellos que vallan a consultarlo en un futuro.
2. **Necesario:** Un requerimiento es necesario si su omisión provoca una deficiencia en el sistema a construir, y además su capacidad, características físicas o factor de calidad no pueden ser reemplazados por otras capacidades del producto o del proceso.
3. **Consistente:** Un requerimiento es consistente si no es contradictorio con otro requerimiento.
4. **No ambiguo:** Un requerimiento no es ambiguo cuando tiene una sola interpretación. El lenguaje usado para su especificación, no debe causar confusión al lector.
5. **Completo:** Un requerimiento está completo si no necesita ampliar detalles en su redacción, por lo que se proporciona toda la información suficiente para su comprensión.

6. **Verificable:** Es verificable cuando puede ser cuantificado de manera que nos permita hacer uso de métodos de verificación (inspección, análisis, demostración o pruebas).
7. **Correctos:** Los requerimientos deben ser revisados por el cliente y desarrollador, para asegurar que no tienen errores.
8. **Reales:** Todos los requerimientos deben ser revisados para asegurar que son posibles de implementar en el sistema
9. **Rastreables:** Los requerimientos pueden ser rastreables hacia atrás y hacia adelante. Rastreables hacia atrás significa que para cada requerimiento se conoce su origen. Rastreable hacia delante significa que todo requerimiento está escrito de tal forma que facilita la referencia a los requerimientos con los que se relaciona.
10. **Modificable:** Un requerimiento es modificable si el lenguaje usado para su definición permite realizar cambios de manera fácil, completa y consistentemente.

Dificultades para definir los requerimientos del software.

La Ingeniería de Requerimientos es un proceso muy complejo debido a la naturaleza y a la diversidad de ambiente de los sistemas, por lo que existe una gran variedad de situaciones que ilustran las dificultades que se enfrentan para lograr la obtención y especificación de los requerimientos del sistema. Estas dificultades son las siguientes:

1. Los usuarios o clientes del sistema no siempre pueden describir con exactitud lo que desean o necesitan.
2. Las descripciones de sus necesidades son expresadas utilizando sus propios términos y suposiciones con las cuales los desarrolladores pueden no estar familiarizados.
3. Diferentes usuarios tendrán distintas percepción de un mismo requerimiento, una visión contradictoria o con distinta prioridad.
4. Los usuarios conocen su trabajo y su sistema actual, pero no siempre pueden describir sus problemas a personas externas.
5. Los usuarios con frecuencia no conocen realmente lo que desean del sistema, y piden demandas irreales debido a que ignoran el costo de sus peticiones.
6. Los clientes encargados de hablar con los desarrolladores pueden no tener experiencia directa en hacer el trabajo que hacen los usuarios del sistema actual.

7. El equipo de desarrollo tiene que descubrir todas las fuentes potenciales de requerimientos, así como las partes comunes y en conflicto, debido a que diferentes usuarios tienen distintos requerimientos y podrían expresarlos de diferentes formas.
8. Los desarrolladores conocen soluciones al sistema, pero no siempre saben como afectará las soluciones posibles a las actividades del negocio de los clientes.
9. Los desarrolladores tienen también su propio lenguaje, y en la mayoría de las veces creen estar hablando en los mismos términos con el cliente, cuando en realidad dan significados diferentes a la misma cosa.
10. Si la comunicación no está cuidadosamente organizada y fomentada entre el equipo de desarrollo y los clientes, puede suceder que los requerimientos sean mal entendidos o queden incompletos.
11. El entorno económico y de negocio en el que se lleva a cabo el análisis es dinámico y la importancia de ciertos requerimientos puede cambiar

Personal involucrado en la Ingeniería de Requerimientos.

Realmente, son muchas las personas involucradas en el desarrollo de los requerimientos de un sistema. Es importante saber que cada una de esas personas tiene diversos intereses y juegan roles específicos dentro de la planificación del proyecto; el conocimiento de cada papel desempeñado, asegura que se involucren a las personas correctas en las diferentes fases del ciclo de vida, y en las diferentes actividades de la Ingeniería de Requerimientos. No conocer estos intereses puede ocasionar una comunicación poco afectiva entre clientes y desarrolladores, que a la vez traería impactos negativos tanto en tiempo como en presupuesto.

Los roles mas importantes pueden clasificarse como sigue:

Usuario final: Son las personas que usarán el sistema desarrollado. Ellos están relacionados con la usabilidad, la disponibilidad y la fiabilidad del sistema; están familiarizados con los procesos específicos que debe realizar el software, dentro de los parámetros de su ambiente laboral. Serán quienes utilicen las interfaces y los manuales de usuarios.

Usuario líder: Son los individuos que comprenden el ambiente del sistema o el dominio del problema en donde será empleado el software desarrollado. Ellos proporcionan al equipo técnico los detalles y requerimientos de las interfaces del sistema.

Personal de mantenimiento: Para proyectos que requieran un mantenimiento eventual, estas personas son las responsables de la administración de cambios, de la implementación y resolución de anomalías. Su trabajo consiste en revisar y mejorar los procesos del producto ya finalizado.

Analistas y programadores: Son los responsables del desarrollo del producto en si; ellos interactúan directamente con el cliente.

Personal de pruebas: Se encargan de elaborar y ejecutar el plan de pruebas para asegurar que las condiciones presentadas por el sistema son las adecuadas. Son quienes van a validar si los requerimientos satisfacen las necesidades del cliente.

Otras personas: Que pueden estar involucradas, dependiendo de la magnitud del proyecto, pueden ser: administradores del proyecto, documentadores, diseñadores de bases de datos, entre otros.

1.3.1.6 Herramientas de Software para la Administración de los Requerimientos.

Cada vez es más común el uso de las herramientas de software en el desarrollo de sistemas, estas herramientas pueden integrarse y constituir una herramienta CASE (Ingeniería de Software Asistida por Computadora) compleja que puede abarcar todo el proceso de desarrollo de un sistema. A continuación se relacionan algunas de las herramientas de software para la administración de los requerimientos. Es preciso aclarar que aunque todas las herramientas de administración de los requerimientos conservan un propósito común, también existen diferencias entre una y otra herramienta que resultan en ventajas y desventajas para su aplicación en los diferentes contextos de desarrollo. En general, las herramientas de software para la administración de los requerimientos, se concentran en administrar y producir el documento de especificación de requerimientos del sistema.

Rational RequisitePro.

Es una herramienta ofrecida por Rational Software para la administración de requerimientos (Racional Co. 2007), esta herramienta proporciona un mayor control para los requerimientos planteados por los usuarios. Mejora la administración de los cambios en los requerimientos y facilita la adición de nuevos requerimientos que surgen en este proceso. Además, con la instalación del Rational RequisitePro en un entorno de red, permite la integración y

comunicación de los miembros del equipo de desarrollo, asegurando la integridad y consistencia de los requerimientos del sistema. De esta manera se reducen los riesgos en el proyecto. Es una solución fácil de usar, es una herramienta de administración de requerimientos que le permite al equipo crear y compartir sus requerimientos utilizando métodos familiares basados en documentos potenciados por la aplicación de las capacidades de una base de datos, tales como la trazabilidad y análisis de impacto. El resultados es una mejor comunicación y administración de requerimientos con una mayor probabilidad de completar los proyectos en tiempo, dentro del presupuesto y superando las expectativas. Es una herramienta centrada en documentos, que almacena los requisitos asociándolos a documentos (aunque también permite guardarlos directamente en la bases de datos), mientras que las otras herramientas están orientadas a requisitos. Permite el uso de Oracle sobre Unix o Windows y también soporta SQL Server sobre Windows. En RiquisitePro los requerimientos se encuentran documentados bajo un esquema organizado de documentos; estos esquemas cumplen completamente con los estándares requeridos por algunas de las instituciones a nivel mundial más reconocidas en el desarrollo de software, tales como: IEEE (Instituto de Ingenieros Eléctricos y Electrónicos), ISO, CMM (Modelo de Capacidad de madurez) y por el RUP (Proceso Unificado Racional), (Arias 2005).

Entre sus beneficios podemos encontrar, (RequisitePro 2007):

- Permite la clasificación de requerimientos, en base a las necesidades de cada empresa.
- Define atributos para todos los tipos de requerimientos especificados.
- Ayuda a manipular el alcance del proyecto mediante la asignación de prioridad de desarrollo a cada uno de los requerimientos planteados.
- Permite características avanzadas de rastreo, por medio de matrices, que permiten visualizar las dependencias entre requerimientos dentro de un proyecto o en diferentes proyectos.
- Administración de cambios mediante el rastreo y la visualización histórica de los cambios efectuados al requerimiento, cuándo y quién lo hizo.
- Manejo de plantillas creadas por el usuario, o creadas por otras empresas.
- Interactúa con los demás productos Rational para el ciclo de vida, así como con herramientas de Microsoft Office.

- Ayuda a determinar en forma automatizada cuántos requerimientos tiene el proyecto.
- Ayuda a determinar responsables y actores en cada uno de los requerimientos.
- RequisitePro, permite organizar los requerimientos, establecer y mantener relaciones padre/hijo entre ellos.

Visual Paradigm.

Es una herramienta CASE que da soporte al modelado visual con UML 2.0, permitiendo representar gráficamente el sistema software, resaltando los detalles más importantes. Se centra en cómo los componentes del sistema interactúan entre ellos, sin entrar en detalles excesivos, de esta forma mejora la comunicación entre los miembros del equipo usando un lenguaje gráfico.

Esta herramienta nos proporciona:

- Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa (versión profesional) e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de múltiples versiones, para cada necesidad.
- Disponibilidad en múltiples plataformas.

El uso de esta herramienta proporciona:

- Ahorro en costes de especificación y de desarrollo minimizando el impacto de errores.
- Mejora la calidad mediante un adecuado análisis y gestión de los requisitos.
- Facilita la reutilización real.
- Mejora la productividad facilitando la reutilización real desde la especificación.
- Reduce las no conformidades del sistema.
- Permite controlar la especificación.
- Permite administrar más fácilmente la especificación.

- Ayuda a cumplir con estándares de calidad.
- Permite centralizar toda la información del problema.
- Permite especificar sistemas de una forma estructurada y gráfica.
- Proporciona una trazabilidad completa de la especificación, etc.

Otras herramientas para la administración de requerimientos.

Las herramientas seleccionadas proporcionan casi todas las necesidades básicas exigibles. Además, son herramientas ampliamente difundidas y muy reconocidas. Existen otras herramientas, aunque no muy reconocidas, son utilizadas también para la administración de requerimientos, entre ellas están:

VORDTool.

La herramienta VordTool fue desarrollada por **Ian Sommerville y Katonya** en el año de 1996 (Medina 2004) y se basa en la especificación de requerimientos orientado a puntos de vistas. El modelo adoptado por VordTool (VORDTools 1996), es orientado a servicios donde los puntos de vistas son análogos a clientes en un sistema cliente-servidor. El sistema capta los servicios como puntos de vista y los puntos de vistas pasan información de control al sistema. Los puntos de vistas mapean clases de usuarios finales de un sistema o las interfaces hacia otros sistemas. Los puntos de vista que constituyen la base del modelo son conocidos como puntos de vistas directos. Los puntos de vistas que respectan a otros sistemas que tiene influencia en el dominio de la aplicación también son considerados y son llamados puntos de vistas indirectos. Ambos puntos de vistas representan a los requerimientos funcionales y no funcionales respectivamente.

Catalyze Interprise:

La herramienta ofrecida por **Steel Trace** (Medina 2004), utiliza el paradigma orientado a objetos con un enfoque basado en casos de uso. Los casos de uso proporcionan un mecanismo para capturar requerimientos funcionales de una manera comprensible y estructurada. Los casos de uso son ejecutados por actores. Un caso de uso es un hilo de funcionalidad que el sistema podría realizar, una colección de casos de uso pueden expresar el comportamiento entero del sistema. Esta herramienta proporciona una manera sencilla de definir los términos utilizados en el proyecto, de esta forma reduce la ambigüedad para los desarrolladores y aumenta la comprensibilidad de los revisores y analistas del sistema.

Integral Requisite Analizer (IRqA)

El IRqA es una herramienta de software desarrollada por **TCP Sistemas e Ingeniería.** Esta herramienta es especialmente diseñada para dar soporte a los analistas y a los ingenieros de software en el proceso de ingeniería de requerimientos. Proporciona la funcionalidad no solo de administrar los requerimientos y producir una especificación, sino también la posibilidad de formular estos requerimientos dentro del contexto del sistema. En IRqA el ciclo de especificación completo incluye la captura de requisitos, análisis, especificación de sistema, validación y la organización de requisitos es soportada por modelos estándares (Mcdonald Landazuri 2005).

OSMRT 1.3

El Grupo de Trabajo de la Base de Datos de Herramientas (TDWG) del Consejo Internacional de Ingeniería de Sistemas (INCOSE) realizó una encuesta (INCOSE 2006) a los proveedores de herramientas para la gestión de requisitos. INCOSE de un total de 37 herramientas analizadas en la encuesta, 10 respondían a la característica de correr en entornos GNU-Linux: CARE 3.2, Cradle 5.2, Focal Point, IRqA 3.3, PACE, SoftREQ, UGS Teamcenter 2005 (Requirements Management), Banyan2.2, Contour y OSRMT 1_3. De ellas solo OSMRT 1.3 es libre lo que facilita su descarga. La "Open Source Requirements Management Tool" (OSRMT), como su nombre lo indica es una herramienta de fuente abierta que corre en múltiples plataformas. Soporta todas las actividades definidas en el proceso gestión de requisitos de manera eficiente. Es una herramienta multi-usuario, independiente de la plataforma, contiene 5 bases de datos, la versión 1.3 añade un árbol de trazabilidad para facilitar la auditoría y la localización de los artefactos, además de un módulo de gestión de los cambios, contiene numerosas correcciones de errores, entre otras características. (OSMRT 2006). Esta herramienta servirá de propuesta en este trabajo de tesis, como solución a la administración de requerimientos utilizando herramientas libres, ya que constituye una alternativa para el proceso de migración, en el cual se encuentra la Universidad de las Ciencias Informáticas, hacia plataformas libres y en particular en Polo Sistemas Geológicos de la facultad 9.

1.3.2 Descripción actual del dominio del problema

La facultad #9 de la Universidad de las Ciencias Informáticas ha decidido llevar a cabo una investigación acerca del proceso de desarrollo del software en el Polo Sistemas Geológicos, y dentro de este proceso, investigar en específico cómo se desarrolla la etapa de levantamientos de requisitos,

la cual teniendo en cuenta los resultados obtenidos, no se ha desarrollado de la forma más óptima posible y es por ello que se hace necesario darle solución a este problema. Haciendo un análisis de esta etapa en este Polo de la facultad 9, se llegó a la conclusión, que la falta de experiencia y de personal capacitado para esta actividad (Ver Anexo 2), es el principal problema, debido a que se utilizan muy pocas técnicas de obtención y validación de requerimientos para la elaboración y desarrollo de un buen producto, además de darse la situación que en muchos casos no se logra obtener en su totalidad lo que el usuario desea para su sistema, trayendo como consecuencia retraso en la elaboración del producto. Es por ello que se hace necesaria la elaboración de una nueva estrategia a seguir, en la cual se apliquen nuevas técnicas para llevar a cabo un proceso de levantamiento de requerimientos más óptimo que el actualmente existente.

1.3.3 Situación Problémica.

La Universidad de las Ciencias Informáticas, desde sus inicios ha estado dividida en facultades las cuales a la par de la docencia llevan a cabo la producción de software, como parte de la formación de futuros profesionales más capacitados. El proceso de elaboración de software para que sea desarrollado de una forma óptima y en el tiempo estimado debe de llevarse a cabo cumpliendo con una serie de etapas incluidas en el proceso de la Ingeniería de Software, las cuales son: Ingeniería de Requerimientos, Diseño del Sistema, Implementación, Validación y Mantenimiento. Una etapa de vital importancia en este proceso es la Ingeniería de Requerimientos, donde se lleva a cabo el proceso de descubrir, analizar, escribir y verificar los servicios y restricciones del sistema de software que se desea producir; este proceso se realiza mediante la obtención, el análisis, la especificación, la validación y la administración de los requerimientos del software. Esta etapa es muy importante ya que de la definición de los requerimientos dependerá la definición de las etapas subsecuentes del desarrollo del software, es decir, que si no se descubren los requerimientos que se encuentran en el ambiente del sistema ó son encontrados en una etapa avanzada del desarrollo del sistema, se tendrá que retroceder nuevamente a la etapa de requerimientos y esto provocaría cambios en el sistema y consecuentemente retraso en la entrega del mismo; constituyendo este el principal problema encontrado en el Polo Sistemas Geológicos de la facultad #9 de la UCI. Este problema está dado debido a la falta de experiencia y de personal capacitado para llevar a cabo un buen proceso de levantamiento de requisitos (Ver Anexo 2), debido a la utilización de pocas técnicas de obtención de requerimientos, en este caso la entrevista, la encuesta y el análisis de documentos, en la etapa de Obtención de Requerimientos, que no son lo suficientemente abarcadoras como para que los desarrolladores y analistas del sistemas logren capturar todos los requerimientos que los usuarios y

clientes desean que lleve el sistema a realizar, ya que en muchos casos no se logra entender lo que estos quieren. Según el análisis llevado a cabo en el proceso de investigación (Ver Anexo 4), se llegó a la conclusión de que los principales problemas detectados en el proceso de Ingeniería de Requerimientos de dicho Polo, radican en la etapa de Obtención de Requerimientos. Es por ello que se hace necesaria la elaboración de una guía, en donde se expongan nuevas técnicas de obtención de requerimientos como son: Puntos de Vista, Escenarios, Análisis de Protocolos, Enfoque Grupal, Prototipos, entre otras, que serán objeto de análisis en el capítulo 2 como posible solución al proceso de levantamiento de requerimientos en el Polo Sistemas Geológicos de la UCI; ya que el resultado de esta primera etapa de la Ingeniería de Software, la Ingeniería de Requerimientos, constituye la base para el diseño, la implementación, la evaluación del software y el mantenimiento del software.

1.4 Análisis de otras soluciones existentes.

Algunas de las metodologías que se representan a continuación tratan de abarcar todas las actividades del proceso de Ingeniería de Requerimientos, otras sólo comprenden ciertas actividades. En general, las metodologías nos indican una manera de cómo llevar a cabo sistemáticamente un proceso, para este caso, sería el proceso de la Ingeniería de Requerimientos.

1.4.1 Joint Application Design (JAD).

El Diseño Conjunto de Aplicaciones es una metodología cooperativa orientada al usuario, fue desarrollada por Chuck Morris y Tony Crawford en IBM en el año de 1977. La efectividad de esta metodología consiste en que el trabajo de obtención de requerimientos se realiza en una sesión de trabajo en la que participan todos los interesados, con el fin de presentar sus puntos de vistas, escuchar a los demás, negociar y ponerse de acuerdo en una solución mutuamente aceptable (Maciaszek 2001). El resultado de esta sesión es el documento JAD final, es un documento de especificación del sistema completo donde se incluyen definiciones de los datos, flujo de trabajo y pantallas de interfaz.

1.4.2 Cooperative Requirements Capture (CRC).

La técnica de captura cooperativa de requerimientos, esta metodología consiste en realizar una sesión de trabajo en la que participan todos los usuarios interesados, cada usuario debe presentar sus puntos de vistas, discutir con los demás para acordar una solución si es que existen diferencias entre los puntos de vistas de los demás participantes. En este aspecto es similar al JAD en su enfoque en

sesiones de grupos de personas, pero con diferencias a JAD porque requiere la participación explícitamente de los interesados (stakeholders) quienes no pueden de otra manera estar involucrados en el desarrollo del sistema. Los participantes de las sesiones de CRC podrían incluir individuos con intereses financieros o de regulación en el producto final (Macaulay 1996).

1.4.3 Objectory.

La metodología Racional Objectory Process, es el resultado de la integración de tres metodologías orientadas a objetos, Rumbaugh 1991, Booch 1994 y Jacobson 1996; (Jacobson, Booch, Rumbaugh 2000). Abarca todo el proceso de desarrollo, aunque en este caso importa lo referente al análisis de requerimientos. El objetivo de la captura de requerimientos en Objectory es decir **qué** debe hacer el sistema de software, definiendo el entorno y el comportamiento del sistema. La primera parte de la captura de requerimientos se realiza utilizando los Casos de Uso, para obtener un modelo en el que clientes y usuarios vean reflejados los servicios que el sistema va a proveer. La notación utilizada en los casos de uso es sencilla y clara. Los modelos de diagramas de casos de uso dan una visión general del sistema, pero se puede agregar detalles utilizando escenarios que en Objectory se modelan aplicando diagramas de secuencia o de colaboración. Para cada caso de uso de este modelo es necesario elaborar varios escenarios, cada uno de ellos representará una situación particular de un caso de uso.

Es importante señalar que en Objectory todo el proceso de desarrollo de un nuevo software será manejado por los casos de uso, esto es, los requerimientos estarán presentes en todo momento, evitando con ello realizar productos de software incorrectos, es decir que no coincidan con las necesidades del usuario.

1.4.4 COHERENCE.

Fue desarrollado en la Universidad de Lancaster, Gran Bretaña (Viller 1998), su enfoque es humano-social, integra el análisis etnográfico para la integración social, y Objectory para el modelado de los requerimientos. En Coherente se produce un enfoque sistemático que integra el análisis social con el análisis orientado a objetos para el diseño de sistemas de cómputo. El objetivo de Coherente es evitar que el usuario tenga que aprender nuevos elementos que ya han cubiertos otras metodologías e integrar los elementos faltantes, como es el análisis social. En Coherente los puntos de vista representan el análisis social y son conectados con los elementos de Objectory (casos de uso, clases, actores y objetos).

1.4.5 SSM (Soft System Methodology).

La metodología SSM fue desarrollada en Inglaterra por Peter Chekland en 1970, cuyo enfoque estudia el entorno en el que el sistema reside, investiga el por qué existe o podría existir el sistema, y cómo adaptarlo en un entorno general. Una vez que el sistema haya sido propuesto, las diferencias entre el sistema actual y el propuesto son examinadas. La metodología SSM se enfoca en un cambio revolucionario entre el sistema actual y el sistema futuro, determinando el estado final y cómo lograrlo, antes de estudiar las mejoras incrementales que podrían realizarse en el sistema actual. Aunque la notación es variada y flexible se hace hincapié en el uso de gráficos para lograr una rica imagen de situaciones (Checkland, Scholes 1990).

1.4.6 Metodología de DSED.

La metodología de DSED (Desarrollo de Sistemas Estructurados de Datos) o de Warnier-Orr fue comenzada por Warnier en 1974 y continuada por Orr en 1977, su enfoque es sobre los datos y lo que busca es tener una herramienta para representar la jerarquía de la información y el flujo que mantiene la información en el dominio del problema (Roger 1998).

Esta metodología comprende los siguientes pasos:

1- El análisis del contexto de la aplicación: Se busca definir los elementos de información, los productores y consumidores de esa información y los puntos de vistas sobre la información de cada productor o consumidor.

2- La definición de las funciones de la aplicación: Se definen las funciones que deben implementarse en el sistema.

3- Definición de los resultados de la aplicación: Se incluye la construcción de un prototipo en papel de las salidas deseadas del sistema y estas deben corresponder a los diagramas de Warnier definidos en el paso uno.

4- Requerimientos físicos: Se hace una lista de los requerimientos del comportamiento del sistema divididos en requerimientos de: comportamiento, fiabilidad, seguridad, hardware e interfaces de comunicación.

1.4.7 COLOR-X

Es una herramienta automática y a su vez una metodología basado en lingüística computacional y el análisis orientado a objetos. Comprende varias etapas de la Ingeniería de Requerimientos: especificación, validación y verificación de requerimientos. COLOR-X es un acrónimo de **C**onceptual **L**inguistically based **O**bject oriented **R**epresentation language for information and communication systems, más una X que agrega el autor J.F.M Burg. La metodología fue realizada como una tesis doctoral en la Universidad de Vrije, Holanda (Burg 1997).

1.4.8 RARE- IDIOM.

Esta metodología presenta el enfoque de reuso; se basa en las propiedades estructurales de los documentos de requerimientos y los procesos realizados para su producción. La metodología propone varias formas de reusar los requerimientos y sus artefactos asociados, todo descrito en términos de patrones de reuso de requerimientos. RARE-IDIOM fue elaborada por Jacob Cybulski en la Universidad de Melboune en Australia (Medina 2004) y es un acrónimo de **R**euse-**A**ssisted **R**equirement **E**ngineering with **I**nformal **D**ocument **I**nterpreter **O**rganizer and **M**anager, su idea es reutilizar en todo momento a partir de que el sistema es solicitado por el cliente. Además propone un nuevo ciclo de vida de Ingeniería de Requerimientos que introduce el reuso.

1.5 Conclusiones Parciales.

En este capítulo hemos visto como la necesidad de una guía para el proceso de levantamiento de requerimiento en los software, ha conducido a la elaboración de la misma, como referencia fundamental para el análisis de posibles soluciones que contribuyan a la recopilación de un conjunto de método y técnicas para el desarrollo del proceso de Ingeniería de Requerimientos en los software del Polo Sistemas Geológicos de la UCI.

El estudio del marco conceptual, que rodea el objeto de estudio que rige esta investigación, ha permitido que se tenga un conocimiento general de la situación actual bajo la cual se desarrolla el proceso de la Ingeniería de Requerimientos en dicho Polo, proceso que es bastante pobre en cuanto al uso de técnicas para la obtención de requerimientos, es por ello que la integración de nuevas técnicas, de obtención de requerimientos, a este proceso, formará parte de la propuesta que será presentada en el próximo capítulo.

CAPÍTULO 2

Propuesta para el levantamiento de requerimientos en el Polo Sistemas Geológico de la UCI.

2.1 Introducción.

A lo largo de la investigación realizada, en el Polo Sistemas Geológicos de la Universidad de las Ciencias Informáticas, para poder desarrollar este trabajo de tesis, se pudo observar que el principal problema que existe con el levantamiento de requerimientos, a la hora de aplicar las técnicas, radica en la primera etapa (Ver Anexo 4), es decir la etapa de obtención de requerimientos; etapa en la cual no se utilizan muchas técnicas de obtención de requerimientos, no sucediendo así en las restantes etapas de la Ingeniería de Requerimientos en las cuales se llevan a cabo una serie de procedimientos y métodos que son necesarios y dan cumplimiento a cada fase de este proceso. Dichas etapas serán mencionadas y explicadas a lo largo del desarrollo de este capítulo como parte de la solución a proponer.

Como posible solución a este problema, en este capítulo se expondrán una serie de técnicas que se llevan a cabo en el proceso de obtención de requerimientos y que ayudarán al equipo de trabajo del Polo de Sistema Geológicos a desarrollar un trabajo de mayor calidad. Se hará mayor énfasis en la primera etapa, etapa de Obtención de Requerimientos, ya que según los análisis realizados, es la que presenta más problemas; se hará también mención de las ventajas y desventajas de cada técnica a la hora de ser utilizadas así como la propuesta de una herramienta de software libre para el proceso de administración de requerimientos, debido a que la Universidad de las Ciencias Informáticas se encuentra inmersa en un proceso de migración al software libre.

2.2 Técnicas para el proceso de Ingeniería de Requerimientos en el Polo Sistemas Geológicos.

2.2.1 Técnicas de Obtención de Requerimientos.

Las técnicas de obtención de requerimientos son aquellas que nos permiten comprender el dominio del sistema, buscar y recolectar información para definir sus límites y restricciones e identificar a las personas interesadas y usuarios involucrados. El resultado permite obtener una colección y clasificación de los requerimientos del sistema, mediante la participación de los clientes y usuarios (Medina 2004). En esta etapa de la Ingeniería de Requerimientos es muy importante la utilización de diversas técnicas para llevar a cabo un buen proceso de Obtención de Requerimientos, técnicas que no necesariamente implican un orden en su realización. Entre las técnicas más utilizadas y que forman parte de la propuesta de este trabajo de tesis además de la entrevista, la encuesta y el análisis de documentos, técnicas utilizadas hasta el momento en el Polo Sistemas Geológicos, podemos encontrar también las siguientes: Joint Application Development (JAD), Puntos de Vistas, La Observación, Análisis de Usuarios Interesados (Stakeholders) entre otras que también son de gran importancia para llevar a cabo un proceso de levantamiento de requerimientos con calidad y que serán explicadas a lo largo del desarrollo de este capítulo.

2.2.1.1 Entrevista.

La entrevista es una técnica para obtener información mediante una conversación dirigida por los analistas a clientes y usuarios, con el objetivo de entender el dominio del problema y sus necesidades. Se basa en un formato de preguntas y respuestas. El desarrollador buscará obtener las opiniones de los clientes o usuarios entrevistados, sus opiniones del sistema, sus metas personales, de la organización y de los procedimientos informales.

Existen cinco puntos que deben tomarse en cuenta para preparar una entrevista (Kendall 1991).

1. **Lectura de antecedentes.** Se debe hacer un previo estudio de antecedentes de los entrevistados y su ambiente de trabajo, con ello se logrará conocer el lenguaje empleado dentro de la empresa.
2. **Establecer objetivos de la entrevista.** Se establecen los objetivos de la entrevista en base a los antecedentes y la experiencia personal.

3. **Selección de los entrevistados.** Se entrevista a gente clave de todos los niveles del sistema.
4. **Preparación del entrevistado.** Se debe avisar a la persona que se entrevistará con tiempo para que pueda reflexionar acerca de la entrevista.
5. **Selección del tipo y la estructura de las preguntas.** Escribir las preguntas que cubran los aspectos fundamentales del objetivo de la entrevista, eligiendo el tipo y la estructura adecuada de las preguntas de la entrevista.

Las preguntas tienen ciertas estructuras básicas que se deben conocer. Los dos tipos básicos de preguntas son abiertas y cerradas.

1. **Las preguntas abiertas** permiten al entrevistado expresar de manera flexible y en consideración de su experiencia responder a lo que se le pregunta.
2. **Las preguntas cerradas** limitan las respuestas disponibles de los entrevistados mediante una lista de opciones o alternativas de respuestas.

La entrevista puede estructurarse de las siguientes maneras:

1. **Estructura de pirámides.** En este caso, el entrevistador inicia con preguntas detalladas, del tipo de preguntas cerradas, y luego va haciendo preguntas abiertas, de respuestas más generalizadas.
2. **Estructura de embudo.** El entrevistador comienza haciendo preguntas abiertas de carácter general y más adelante va utilizando preguntas cerradas.
3. **Estructura de diamantes.** Combina las estructuras anteriores, el entrevistador comienza de una manera muy específica, luego examina aspectos muy generales y finalmente llega a una conclusión muy específica.

Las entrevistas deben registrarse ya sea por un cuaderno de notas o utilizando una grabación. Es importante que al final de la entrevista se escriba un informe, con el fin de asegurar la calidad de los datos de la entrevista.

Entre las **ventajas** que esta técnica nos proporciona tenemos:

1. Es utilizada como punto de partida en la obtención de los requerimientos.
2. Se logra conocer la opinión del cliente y los usuarios.

3. Se conoce la organización de la empresa.
4. Obtienen las metas del sistema.
5. Se logra un entendimiento general del problema.
6. Puede combinarse con otras técnicas.

Como **desventajas** de la aplicación de esta técnica podemos encontrar:

1. No existe confianza entre los usuarios y los desarrolladores.
2. Utilizan diferente terminología.
3. Difícil de expresar.
4. Sabotear la entrevista por falta de cooperación.
5. Se da mucha información irrelevante, por parte del cliente o usuario.

2.2.1.2 Cuestionarios.

Es una técnica la cual constituye una manera de obtener información que permite a los desarrolladores del sistema recopilar opiniones, posturas, conductas y características de los diversos usuarios que son encuestados y que se encuentran involucrados en la operación del sistema actual o en la implantación de un sistema nuevo. Los cuestionarios son eficaces si la gente de la organización se encuentra dispersa o si está involucrada mucha gente en el proyecto de sistema (Kendall 1991). Al igual que la entrevista, un cuestionario puede redactarse usando preguntas abiertas o preguntas cerradas, además es importante usar un vocabulario que refleje el lenguaje de los involucrados en la organización. El uso de preguntas cerradas en los cuestionarios permite a los analistas medir los resultados mediante gráficas. Esta técnica suele combinarse con las entrevistas para mejorar la obtención de información del sistema.

Entre las **ventajas** que podemos citar acerca del uso de los cuestionarios tenemos:

1. Se obtiene una gran cantidad de información a través del usuario.
2. Son flexibles.
3. Permite obtener datos que pueden medir resultados de forma escalar.
4. Permiten combinarse con otras técnicas.

Como **desventajas** podemos encontrar:

1. Se obtiene un gran volumen de información que requiere de análisis.
2. Se obtiene información redundante y en ocasiones incompletas.
3. Se requiere de experiencia en el dominio para tratar de comprender el comportamiento de los encuestados.

2.2.1.3 Joint Application Development.

La técnica denominada JAD (*Joint Application Development, Desarrollo Conjunto de Aplicaciones*), desarrollada por IBM en 1977, es una alternativa a las entrevistas individuales que se desarrolla a lo largo de un conjunto de reuniones en grupo durante un período de 2 a 4 días. En estas reuniones se ayuda a los clientes y usuarios a formular problemas y explorar posibles soluciones, involucrándolos y haciéndolos sentirse partícipes del desarrollo.

Esta técnica se basa en cuatro principios: dinámica de grupo, el uso de ayudas visuales para mejorar la comunicación (diagramas, transparencias, multimedia, herramientas CASE, etc.), 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)*, por lo que durante las reuniones se trabaja directamente sobre los documentos a generar.

2.2.1.4 Puntos de Vistas.

En cualquier sistema existen varios tipos de usuarios que tienen diferentes intereses en los requerimientos del sistema, es decir, existen diferentes puntos de vista para un problema. Los enfoques orientados a puntos de vista toman en cuenta estos diferentes puntos de vista y los utilizan para organizar y estructurar tanto el proceso de obtención como los requerimientos (Sommerville 1998). Se toma en cuenta la existencia de diferentes perspectivas y proporciona un esquema para descubrir problemas con los requerimientos propuestos por diferentes usuarios.

Los puntos de vista pueden ser considerados como:

1. **Una fuente o consumidor de datos,** son responsables de producir o consumir datos.
2. **Un marco de trabajo de la representación,** se considera un tipo particular del modelo del sistema.
3. **Un receptor de servicios,** son externos al sistema y reciben servicios de él.

Las lluvias de ideas y los puntos de vista orientados a eventos son técnicas que utiliza este enfoque.

Lluvias de Ideas

En esta técnica se identifican los servicios potenciales y las entidades que interactúan con el sistema. Se lleva a cabo mediante una reunión entre el equipo de desarrollo y la empresa que solicita el software. Las personas involucradas generalmente son los usuarios del sistema y los administradores. En esta reunión todos expresan sus ideas sobre el problema y la posible solución. Cada persona participa sin ser interrumpida por otra, las ideas son anotadas dentro de un diagrama de burbuja, y no pueden ser criticadas por los demás participantes. Al finalizar la sesión, se realiza una recolección de ideas sin duplicidad (Sommerville 1998).

Haciendo uso de este tipo de técnica obtenemos como **ventajas**:

1. Permite estimular la aportación de ideas.
2. Se crea un ambiente de confianza.
3. Puede combinarse con otras técnicas.

Pero además existen **desventajas** que pueden frenar el proceso de captura de requerimientos.

1. Existen personas que no les gusta hablar en público y que posiblemente tengan buenas ideas.
2. Se requiere de un buen facilitador que asegure el éxito de la técnica.
3. Puede haber requerimientos externos no encontrados.

2.2.1.5 Observación.

Es una técnica en donde el desarrollador se sumerge en el ambiente de trabajo de los usuarios, para observar el trabajo diario que estos realizan anotando los aspectos importantes, observando factores sociales y organizacionales que afecten el sistema. Existen dos enfoques de esta técnica y son la etnografía y grabar lo observado (Kotonya, Sommerville 2000).

Etnografía:

Generalmente los sistemas de información se encuentran dentro de un contexto social y organizacional, donde los requerimientos del sistema se derivan y se restringen conforme a este contexto. La etnografía es una técnica que se utiliza para entender estos tipos de

requerimientos sociales y organizacionales. El desarrollador se involucra en el entorno laboral donde se utilizará el sistema, para observar y anotar las tares reales que se llevan a cabo. Esta técnica descubre requerimientos implícitos que reflejan los procesos reales más que los formales, en los que la gente está involucrada, y permite descubrir los factores sociales y organizacionales que puedan afectar al sistema. Una desventaja de este enfoque es que no siempre puede identificar nuevas propiedades o agregar otras al sistema (Sommerville 1998).

Estas técnicas son muy eficaces ya que nos permiten:

1. Contacto directo con el medio ambiente.
2. Se detectan problemas.
3. Pueden combinarse con otras técnicas como el prototipado y los Casos de Uso.

Pero tiene como **desventajas**:

1. Tiempo prolongado y difícil de permanecer en calidad de observador.
2. Los usuarios presentan incomodidad al ser observados.

2.2.1.6 Escenarios.

Los escenarios son representaciones de cómo el usuario interactúa con el sistema. En este esquema se representan gráficamente las entradas, salidas, el flujo de datos y el comportamiento del sistema. Los escenarios pueden agregar detalles a un bosquejo de acción. Los escenarios son ejemplos de secciones de interacción de una simple operación entre el usuario y el sistema, el proceso de desarrollo de un escenario puede ayudar a entender los requerimientos (Kotonya, Sommerville 2000). Los escenarios son parte básica de algunos métodos de análisis orientados a objetos. Existen dos esquemas para esta representación: utilizando escenarios de eventos ó casos de uso.

Escenarios de Eventos:

Este enfoque se utiliza para documentar el comportamiento del sistema ante eventos específicos. Los escenarios de eventos incluyen una descripción del flujo de datos y las acciones del sistema, documenta las excepciones que pueden ocurrir. Un escenario de evento es una instancia de un caso de uso, es decir, nos representa una secuencia de sucesos que podrían ocurrir en una situación dada (Sommerville 1998).

Casos de Uso:

Los casos de uso es una técnica que se basa en escenarios para la obtención de requerimientos (Sommerville 1998). En los casos de uso se especifica una secuencia de acciones, incluyendo variantes que el sistema puede llevar a cabo y que producen resultados observables de valor para un actor determinado (Jacobson, Booch, Rumbaugh 2000). En un modelo de caso de uso se puede identificar a un conjunto de actores involucrados y su relación con varios casos de uso. Un conjunto de casos de uso representan todas las posibles interacciones que ocurrirán en los requerimientos del sistema. Los actores pueden tener las siguientes representaciones: personas, sistemas o hardware y cualquier de las tres representaciones puede estar relacionada con un caso de uso. Los requerimientos funcionales pueden estar representados mediante casos de uso y muchos de los requerimientos no funcionales pueden estar asociados a ellos.

El uso de esta técnica nos brinda las siguientes **ventajas**:

1. Fácil de comprender y criticar un escenario.
2. Describe el estado inicial del escenario.
3. Describe el flujo de eventos.
4. Descubre excepciones y soluciones.
5. Descubre cómo llevar a cabo situaciones paralelas.
6. Describe el estado final del escenario.

Entre sus **desventajas** podemos mencionar.

1. Toman tiempo de acuerdo a la complejidad del sistema
2. Las personan que modelen el sistema deben conocerlo previamente.

2.2.1.7 Talleres de Trabajo basados en Casos de Uso (Run Use Case WorkShop).

Los talleres de trabajo basados en casos de uso se realizan entre cliente/usuario y el equipo de desarrolladores (Craig 1999). Esta técnica consiste en dos tareas principales:

- **Generar los escenarios,** mediante la aplicación de los casos de uso se puede conversar con los usuarios para extraerles los detalles que suceden cuando realizan un evento determinado. De esta manera se podrán definir los actores que participan y definir los pasos que se realizan para el caso de uso en cuestión. Se debe verificar con los usuarios si los pasos están bien descritos, o pueden ser modificados y mejorados.
- Al término de la etapa anterior el equipo de desarrolladores **describe y deduce los requerimientos** a partir del conocimiento previamente adquirido.

Como **ventajas** de la aplicación de esta técnica tenemos:

1. Permite la participación directa de los usuarios para definir escenarios.
2. Permite crear un ambiente adecuado para la participación de los demás usuarios.

Como **desventajas** podemos citar:

1. Existe el riesgo de caer en grandes diferencias de interpretación o contradicciones.
2. Los talleres pueden caer en ser pesados a los usuarios y generar desaliento y poca participación.
3. Se pueden deducir requerimientos con diferente percepción.

2.2.1.8 Análisis de Usuarios Interesados (Stakeholders).

Los ***stakeholders*** son las personas involucradas de alguna manera en el sistema y que permiten asegurar el éxito del proyecto, por ejemplo los usuarios finales y sus administradores, clientes y dueños del negocio, ingenieros de sistemas y programadores (Lauesen 2002). Es importante encontrar a todos los grupos de stakeholders y conocer sus intereses. Durante el análisis de stakeholders se debe tratar de encontrar respuesta a las siguientes preguntas (Kotonya, Sommerville 2000).

- ¿Quiénes son los stakeholders?
- ¿Qué objetivos visualizan para el sistema?
- ¿Cómo podrían contribuir en el sistema?
- ¿Qué riesgos y costos ven?
- ¿Qué tipo de soluciones observan, para el sistema?

Hay varias maneras de obtener esta información, puede obtenerse mediante una reunión grupal, donde todos los stakeholders conocidos estén representados ó pueden citarse grupos pequeños de stakeholders a reuniones. En caso de que esto fallara, puede entrevistarse a uno por uno del grupo de stakeholders identificados.

Con la aplicación de esta técnica:

1. Se conoce la jerarquía organizacional del dominio.
2. Se obtiene un gran número de requerimientos desde diferentes puntos de vistas.
3. Se logra la participación de diferentes sectores del negocio.

Pero la misma tiene como **desventajas**:

1. Diferentes percepciones de los requerimientos requiere de secciones de negociación y homogenización.
2. Se debe establecer una agenda acordada que no afecte a ningún participante.
3. Requiere de tiempo si la empresa es grande organizacionalmente.

2.2.1.9 Demostración de Tareas.

La técnica de demostración de tareas es una variante de la entrevista y la observación, y consiste en preguntar a los usuarios cómo realizan una tarea específica del sistema. Es útil cuando los usuarios no pueden explicar lo que hacen en su trabajo diario, pero si pueden demostrar cómo realizan tareas específicas. La demostración de tareas es también una manera poco frecuente de observar tareas críticas. A los usuarios se les hace más fácil demostrar cómo realizan sus tareas, que explicar con palabras cómo se desarrollan las cosas incluso antes de que sean introducidas al sistema actual. Otra aplicación de la demostración de tarea es la identificación de problemas de utilidad, para la identificación de problemas de utilidad en un sistema nuevo es necesario algún tipo de demostración (Lauesen 2002).

Entre las **ventajas** de la aplicación de esta técnica tenemos:

1. Al usuario le es más fácil demostrar cómo hacen sus tareas.
2. Se logra una mayor interacción con el usuario.
3. Se pueden encontrar detalles omitidos u olvidados por los usuarios en otras técnicas.
4. Se logra un mayor entendimiento del sistema.

Como **desventajas** tenemos.

1. Se requiere de la participación voluntaria del usuario.
2. El usuario se incomoda cuando se le cuestiona sobre su trabajo.
3. Requiere demasiado tiempo.
4. El usuario trata de hacer lo mejor posible su tarea y omite posibles defectos.

2.2.1.10 Enfoque Grupal (Focus Groups).

Esta técnica consiste en reuniones grupales donde se fomenta la discusión abierta entre los participantes. La discusión proporciona al analista ideas de cómo los usuarios piensan y sienten acerca de los aspectos del sistema. Este enfoque puede ser usado en el desarrollo del proceso de la Ingeniería de Requerimientos, es decir, en la etapa inicial del proyecto, el enfoque grupal puede ser utilizado para obtener requerimientos basándose en lo que los usuarios piensan que el sistema podría abarcar y durante el desarrollo podría ser usado para evaluar prototipos y proporcionar validación y verificación del producto (Templeton 1994).

Entre las principales **ventajas** que esta técnica nos proporciona tenemos:

1. Se detectan los sentimientos de los usuarios hacia el sistema.
2. Provee un medio para hacer surgir ideas de los demás participantes.
3. Se logra obtener los límites y restricciones del sistema.

Como **desventajas** tenemos:

1. Se requiere de un buen moderador para que la reunión tenga éxito.
2. Puede tomar tiempo si no se dirige la reunión hacia objetivos específicos.
3. Pueden encontrarse grupos políticos de usuarios que afecten sus intereses personales, más a los intereses de la empresa.

2.2.1.11 Talleres Futuros (Future Workshops).

Esta técnica es similar al enfoque grupal, pero es más estructurada y muy concreta en su alcance. Consiste en que los participantes definan el estado futuro deseado del ambiente del

sistema. Los facilitadores se aseguran de que el alcance del sistema sea definido y respetado, además permite a los participantes discutir el estado final deseado del sistema (Macaulay 1996).

Los talleres futuros tienen como **ventajas**:

1. Provee un medio para hacer surgir ideas de los demás participantes.
2. Se logra obtener los límites y restricciones del sistema.
3. Permite visualizar el estado final del nuevo sistema y prever las afectaciones o restricciones que tendrá.

Entre sus **desventajas** tenemos:

1. Se puede exagerar en la visualización de las funciones que tendrá el futuro sistema.
2. Existe el riesgo de que los usuarios tengan otra percepción del sistema a como lo delimitan los desarrolladores.

2.2.1.12 Estudio de Documentos/Datos.

Esta técnica consiste en estudiar los documentos existentes tales como formas, archivos de datos, documentación y base de datos existente. Se puede analizar las pantallas desplegadas por el sistema en uso, si son impresas en papel. Otra función de esta técnica, es la de verificar la información recabada en las entrevistas realizadas a los usuarios y clientes del sistema. En general lo que se desea lograr es determinar el contexto y la información detallada del sistema en la que los requerimientos están basados (Morris, Tamm 1993).

Entre las **ventajas** de la aplicación de esta técnica tenemos:

1. Permite realizar un planteamiento del sistema en base de formas, archivos y datos que son utilizados en el sistema actual.
2. Requiere solo de la participación del desarrollador sin afectar a los usuarios en sus tareas.

Como **desventajas** podemos citar:

1. No hay participación activa del usuario.
2. Se corre el riesgo por suposiciones de los desarrolladores en el análisis de documentos y formas.

2.2.2 Técnicas de Análisis de Requerimientos.

Los requerimientos deben ser analizados detalladamente tanto por el equipo de desarrollo como por los usuarios para resolver conflictos de ambigüedad y consistencia entre requerimientos, priorizando los requerimientos que han de ser inicialmente implementados en el sistema, además de negociarlos con flexibilidad para homogeneizar su comprensión y de esta forma obtener una lista consistente de requerimientos. Para realizar el análisis se cuenta con diversas técnicas, como las que se mencionan a continuación.

2.2.2.1 La identificación de requerimientos.

Se debe identificar a cada requerimiento de manera única, de tal forma que puedan permitir una referencia cruzada con otros requerimientos, y así utilizarse en las evaluaciones de rastreo. Para identificar cada requerimiento se pueden usar las letras iniciales del tipo del requerimiento y un número de referencia, por ejemplo si se tratara de un requerimiento funcional puede usarse la siguiente estructura RFn, donde n es un número entero consecutivo para cada requerimiento y para los requerimientos no funcionales RNFn.

2.2.2.2 Listas de Verificación.

Esta técnica es muy fácil de utilizar y proporciona una gran utilidad. En general, es una lista de preguntas que el analista debe usar para evaluar cada requerimiento. Los analistas deben verificar y marcar los puntos de esta lista mientras leen el documento de requerimientos, cuando se descubran problemas potenciales, deberán ser anotados, ya sea en los márgenes del documento, ó en una lista de verificación. Las listas de verificación son útiles porque brindan un recordatorio de lo que se debe buscar y reducen la posibilidad de pasar por alto alguna verificación importante. No sólo son útiles para verificar requerimientos, sino también se puede aplicar con los casos de uso. El proceso de verificación se puede implementar con una hoja de cálculo en donde las filas representan, por ejemplo, los distintos requerimientos, y las columnas representan los puntos a verificar ó los criterios que deben cumplir (Ver Anexo 5). Entonces se completan las intersecciones que correspondan con los comentarios sobre los problemas potenciales.

2.2.2.3 Matriz de dependencia.

Suponiendo que los requerimientos sean claramente identificados y enumerados, entonces podemos utilizar una matriz de dependencia de requerimientos, donde se listen en orden todos los requerimientos identificados, como se muestra en la siguiente tabla (Tabla 1).

Requerimientos	R1	R2	R3	R4
R1	X	X	X	X
R2	Conflicto	X	X	X
R3			X	X
R4		Intercalado	Intercalado	X

Tabla 1: Matriz de Dependencia entre Requerimientos.

Las celdas de la matriz son marcadas con una X si dos requerimientos tienen dependencia o se escribe en las celdas si están intercalados o en conflicto. Una celda vacía indica que los requerimientos son independientes. Los requerimientos en conflicto o intercalados podrán ser discutidos con los clientes para después ser replanteados y de este modo resolver los conflictos o la intercalación entre estos. La matriz de dependencia de requerimientos es una técnica simple pero efectiva para encontrar conflictos e intercalaciones cuando el número de requerimientos es relativamente pequeño. Sin embargo, cuando el número de requerimientos es mayor, la técnica aún puede ser usada si los requerimientos se agrupan en categorías, de esta forma los requerimientos pueden ser comparados separadamente en cada categoría.

2.2.2.4 Negociación de Requerimientos.

El propósito de la negociación es resolver conflictos en el desarrollo del sistema. Los conflictos pueden encontrarse entre desarrolladores y clientes, pero los conflictos más serios surgen entre los grupos de stakeholders que existen dentro del ambiente del sistema. Los conflictos dentro del ambiente del sistema pueden presentarse de diferentes maneras. Un miembro de un grupo de discusión debe participar desde alguna posición dentro de las partes en problemas para poder entender el inconveniente y proponer una solución. Para lograr acuerdos, las partes en conflictos deben estar dispuestas a hablar e intentar entender la postura de la parte contraria. Si esto no sucede, el trabajo obtenido podría estar basado desde un punto de vista en particular y no de un común acuerdo de los participantes. De otra manera, pudiera ser resuelto por las decisiones de

personas que se encuentran en niveles superiores en la organización. Existen varias maneras disponibles para resolver los conflictos, por ejemplo que cada parte en conflicto explique que le hace falta a la otra. Desde el punto de vista de los requerimientos, lo más importante es analizar los objetivos de las partes involucradas en el desarrollo del sistema. Generalmente, los conflictos son sobre las soluciones que se plantean, pero mediante la participación de todos los involucrados se pueden ajustar estas soluciones incluso los objetivos mismos. La meta es encontrar soluciones que no entren en conflictos, pero que impliquen todos los objetivos en común.

2.2.3 Técnicas de Especificación de Requerimientos.

En esta actividad se obtiene el documento de especificación de requerimientos que es orientado a diversos tipos de lectores, por lo que es necesario contar con múltiples niveles de descripción. La especificación de los requerimientos debe incluir diferentes modelos para representar el sistema con mayor detalle y desde diversas perspectivas. Es necesario contar con modelos de representación, métodos, metodologías y herramientas que proporcionan la manera de describir los diferentes tipos de requerimientos del sistema en el *documento de especificación*. El documento es redactado para que sea interpretado por los usuarios y por los diseñadores del sistema, recordemos que los usuarios son personas sin conocimientos técnicos de implementación, por otro lado el equipo de diseño son expertos en el área, por lo que se requiere de ambas perspectivas del documento. Existen varias maneras de describir los requerimientos de software y se describen a continuación.

- **Lenguaje Natural.**
- **Notación Gráfica.**
- **Especificación Matemática.**

2.2.3.1 Lenguaje Natural.

El lenguaje natural es utilizado en la mayoría de las veces para escribir los requerimientos del sistema con el fin de que los usuarios puedan comprenderlos sin necesidad de poseer conocimientos técnicos de desarrollo y comprobar que sus necesidades están expresadas en el documento de requerimientos. Por otro lado, aunque el lenguaje natural sea flexible y sencillo puede contener problemas de interpretación. Algunos de los problemas que se pueden encontrar en la descripción de los requerimientos utilizando el lenguaje natural son:

- **Falta de claridad.** Es difícil utilizar el lenguaje de forma precisa y no ambigua.

- **Confusión de requerimientos.** Se dificulta la distinción de los requerimientos funcionales de los no funcionales, las metas del sistema y la información para el diseño.
- **Conjunción de requerimientos.** Varios requerimientos diferentes se expresan de forma conjunta como un único requerimiento.
- **La comprensión del lenguaje natural.** Radica en que los lectores y redactores de la especificación utilicen la misma palabra para el concepto.
- **La especificación del lenguaje natural es demasiado flexible.** Se puede decir lo mismo de formas completamente diferentes.
- **No existe una forma fácil de modularizar los requerimientos.** En el lenguaje natural es difícil exhibir todos los requerimientos relacionados.

Existen dos modificaciones del lenguaje natural que restringen la manera de redactar los requerimientos y son **lenguaje natural estructurado** y **lenguaje de descripción de diseño.**

Lenguaje Natural Estructurado.

Este lenguaje es una forma restringida del lenguaje natural para expresar los requerimientos del sistema asegurando que se imponga un cierto grado de uniformidad a la especificación y manteniendo la expresividad y comprensión del lenguaje natural. Para la notación de este lenguaje es fundamental el uso de plantillas o formas estándar para especificar los requerimientos. Cada forma debe incluir la siguiente información:

- Una descripción de la función o entidad a especificar.
- Una descripción de sus entradas y de dónde provienen.
- Una descripción de sus salidas y hacia dónde van.
- Una descripción de qué otras entidades se utilizan.
- Si se utiliza un enfoque funcional, definir una precondición y poscondición
- Una descripción de los efectos colaterales (si existen) de la operación.

Con esto podemos lograr eliminar algunos problemas y obtener menos variación en la especificación, sin embargo pueden permanecer algunas ambigüedades en la especificación.

Lenguaje de Descripción de Diseño.

Los requerimientos pueden describirse de forma operacional utilizando un Lenguaje de Descripción de Programas (PDL), para anular las ambigüedades inherentes en la especificación en el lenguaje natural. Un PDL contiene construcciones abstractas para incrementar su poder de expresividad, es decir, utiliza un lenguaje similar a un lenguaje de programación pero con características más genéricas. Su ventaja es que se verifica de manera sintáctica y semántica con herramientas de software. Las omisiones de los requerimientos y las inconsistencias se infieren de los resultados de las verificaciones. El uso de PDL nos lleva a la especificación detallada y algunas veces muy cercana a la implementación.

2.2.3.2 Notaciones Gráficas.

Es una manera ilustrativa de generar la especificación de los requerimientos mediante un conjunto de modelos. Los modelos utilizan un lenguaje gráfico, complementado con anotaciones textuales que describen el problema a resolver y el sistema a desarrollar. En muchos casos, los modelos de requerimientos del sistema son más entendibles que la descripción en lenguaje natural, debido a las representaciones gráficas. Son considerados vías directas entre los procesos de requerimientos y diseño.

Los modelos se utilizan en la IR para desarrollar una comprensión del sistema existente a reemplazar, mejorar o para especificar el sistema requerido. Un modelo es una abstracción del sistema que se va a desarrollar, más que una representación alternativa de ese sistema, por lo que una abstracción omite detalles simplificando y seleccionando las características más sobresalientes de la entidad representada (Medina 2004). En contraste, una representación de un sistema debe mantener toda la información. Por lo que existen diferentes tipos de modelos que se basan en varios enfoques de abstracción y representan al sistema desde diferentes perspectivas.

1. **Perspectiva Externa** la que se modela el contexto o entorno del sistema.
    a. *Modelos de Contextos.* Define los límites del sistema, distinguiendo lo que es el sistema y su entorno, como se muestra en la siguiente figura (Figura 9), tomando como punto de partida, un cajero automático.

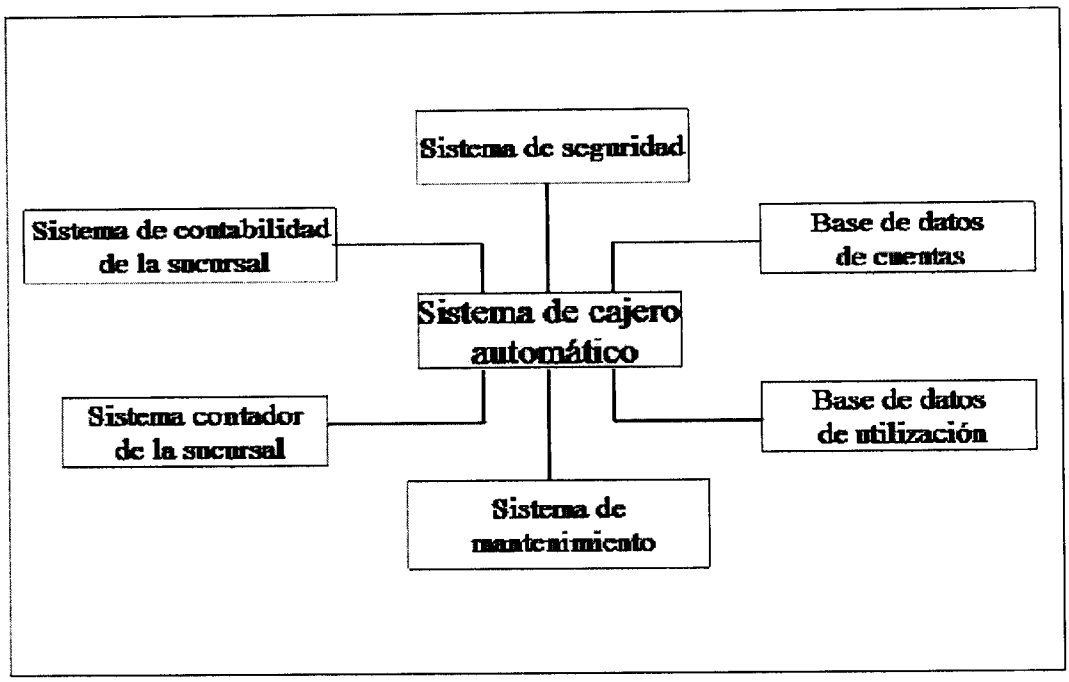

Figura 9: Modelo de contexto de un sistema de cajero automático.

2. **Perspectiva de Comportamiento.** Se utilizan para describir el comportamiento completo del sistema.
    a. *Modelos de Flujos de Datos.* Modelan el comportamiento de datos en el sistema, muestran como fluyen los datos a través de una secuencia de pasos de procesamientos.
    b. *Modelos de Máquinas de Estados.* Modelan el comportamiento de un sistema en respuesta a eventos internos o externos, mostrando los estados del sistema y los eventos que provocan las transiciones de un estado a otro.
3. **Perspectiva Estructural.** Se modela la arquitectura del sistema o la estructura lógica de los datos procesados por este.
    a. *Modelo Entidad-Relación.* Muestran las entidades de los datos y sus atributos asociados y las relaciones entre estas entidades.
    b. *Modelos de Datos Semánticos.* Muestran los modelos de datos en conjunto con los de flujos de datos para describir la estructura de la información que se está procesando.

4. **Perspectiva Orientada a Objetos.** Combina los modelos de comportamiento y el estructural representando los datos mediante objetos y expresar los requerimientos del sistema utilizando casos de uso, realizan el diseño utilizando objetos y desarrollan el sistema en un lenguaje de programación orientado a objetos.

    a. *Modelos de Casos de Uso.* Permite que los desarrolladores de software y los clientes lleguen a un acuerdo sobre los requerimientos, es decir, sobre las condiciones y posibilidades que debe cumplir el sistema. El modelo de casos de uso sirve como acuerdo entre el cliente y desarrolladores y proporciona las bases fundamentales para el análisis, el diseño y las pruebas. Un modelo de casos de uso contiene actores, casos de uso y sus relaciones.

    b. *Modelos de Clases Jerárquicas (herencia).* Identifican las clases de objetos que son importantes en el dominio del sistema. Organiza los objetos mediante una taxonomía. Una taxonomía es un esquema de clasificación que muestra la manera en que una clase de los objetos esta relacionada con otras clases a través de servicios y atributos comunes.

    c. *Modelado de Comportamiento de Objetos (UML).* En UML se utilizan diagramas de secuencias (Anexo 6) y diagramas de colaboración para mostrar el comportamiento e intercambio de mensajes de los objetos del sistema.

5. **CASE.** Es un conjunto de herramientas que ayudan a una fase particular del proceso de desarrollo de software como el diseño, la implementación o las pruebas.

6. **Prototipo.** Un prototipo es una versión inicial de un sistema de software que se utiliza para demostrar los conceptos, probar las opciones de diseño y de forma general enterarse más acerca del problema y sus posibles soluciones.

2.2.3.3 Especificación Matemática.

La especificación de los requerimientos es redactada usando notaciones que se basan en conceptos matemáticos como el de máquinas de estados finitos o conjuntos. Estas especificaciones no ambiguas reducen los argumentos sobre la funcionalidad del sistema entre el cliente y el desarrollador. Sin embargo muchos clientes no comprenden las especificaciones formales y se rehúsan a aceptarlas como contrato del sistema.

2.2.4 Técnicas de Validación de Requerimientos.

La validación es la actividad que permite verificar si los requerimientos obtenidos y analizados son los que definen el sistema que el cliente desea, y se lleva a cabo verificando cuidadosamente algunos criterios de validación que son establecidos, antes de que el documento de requerimientos sea usado como base en etapas posteriores del proceso de desarrollo del sistema. La validación y el análisis de requerimientos son procesos que tiene en común encontrar problemas con los requerimientos. La validación utiliza el documento de requerimientos completo mientras que el análisis trabaja con requerimientos incompletos. En esta actividad se pretende detectar y resolver conflictos entre requerimientos, verificar la consistencia e integridad entre los requerimientos, asegurar que los requerimientos puedan implementarse con la tecnología existente, considerando también el presupuesto y la calendarización del desarrollo del sistema y generar casos de prueba para los requerimientos.

2.2.4.1 Criterios de Validación.

Existen algunos criterios de validación que se deben llevar a cabo en el documento de requerimientos. Estos criterios incluyen:

- ***Verificaciones de consistencia.*** Los requerimientos en el documento no deben contradecirse, es decir, no debe haber restricciones contradictorias o descripciones distintas de la misma función del sistema.
- ***Verificaciones de completitud.*** En el documento de requerimientos deben estar los requerimientos que el usuario especifica, es decir deben contener todas las necesidades de los usuarios.
- ***Verificaciones de ambigüedad.*** Los requerimientos deben verificarse, usando el conocimiento de la tecnología existente, para asegurar que puedan implementarse, considerando también el presupuesto y la calendarización del desarrollo del sistema.
- ***Verificaciones de rastreabilidad.*** Los requerimientos del sistema siempre deben redactarse de tal forma que sea posible rastrearlos hacia su origen o hacia los requerimientos dependientes que lo detallan.
- ***Verificaciones de Correctibilidad.*** Los requerimientos deben estar bien escritos, esto significa que pueden ser revisados por los clientes y desarrolladores.

- ***Verificaciones de Modificabilidad.*** Los requerimientos deben ser especificados utilizando un lenguaje que permitan modificaciones.

Existen varias técnicas de validación de requerimientos que pueden utilizarse en forma individual o combinada.

2.2.4.2 Revisiones del Documento de Requerimientos.

Este es un proceso manual, en el que se involucran varias personas para verificar el documento final de requerimientos, y participan tanto el personal del cliente como de los desarrolladores, en la revisión de anomalías y omisiones. El equipo de revisores debe verificar la consistencia de cada requerimiento y la integridad de estos como un todo, también se comprueban que el documento de requerimientos cumpla con los criterios establecidos de validación.

Los conflictos, contradicciones, errores y omisiones deben señalarse durante la revisión y registrarse formalmente.

2.2.4.3 Escenarios.

Los escenarios son representaciones de cómo el usuario interactúa con el sistema. En este esquema se representan gráficamente las entradas, salidas, el flujo de datos y el comportamiento del sistema. Los escenarios pueden agregar detalles a un bosquejo de acción.

2.2.4.4 Construcción de Prototipos.

En este enfoque de validación se presenta un modelo ejecutable del sistema a los usuarios y clientes, con el fin de que puedan realizar pruebas y comprobar si cumplen sus necesidades reales.

2.2.4.5 Generación de Casos de Pruebas.

Los requerimientos deberán ser probados. Si las pruebas pueden ser consideradas como parte del proceso de validación, esto frecuentemente permite describir los problemas en los requerimientos.

2.2.4.6 Análisis de Consistencia Automático.

Si los requerimientos son expresados de manera automática mediante el uso de herramientas CASE, entonces dichas herramientas permitirán verificar la consistencia del modelo. Un analizador de requerimientos produce un informe de las inconsistencias descubiertas.

2.2.5 Técnicas de Administración del Documento de Requerimientos.

Se conoce como Administración de los Requerimientos de software al proceso de comprender y controlar los cambios en los requerimientos del sistema. Este proceso se lleva a cabo junto con otros procesos de la ingeniería de requerimientos, y su planeación comienza al mismo tiempo que la obtención inicial de los requerimientos. La administración de los requerimientos debe comenzar tan pronto como esté lista la primera versión del documento de requerimientos.

2.2.5.1 Evolución de los Requerimientos.

Existen diversas formas en que los requerimientos evolucionen o sean susceptibles a cambiar. Conforme se va desarrollando la definición de requerimientos se tiene mejor comprensión de las necesidades del usuario, esto hace que el usuario se retroalimente de información y provoque que existan cambios en los requerimientos, más aun cuando se trata de sistemas demasiado grandes. El desarrollo de software puede llevar tiempo, y con esto el entorno del sistema y los objetivos del negocio puedan sufrir cambios, por lo tanto, los requerimientos deben adaptarse para reflejar estos cambios.

2.2.5.2 Administración del Cambio en los Requerimientos.

La administración en el cambio de los requerimientos se aplica a todo los cambios propuestos en los requerimientos, en donde se evalúa el impacto y costo de dichos cambios. En la figura se muestran los pasos a seguir en esta administración. La ventaja de utilizar esta administración, es que todos los cambios propuestos son tratados de forma consistente y que los cambios en el documento de requerimientos se realizan de forma controlada. Como se muestra en la figura 10 existen tres etapas de administración de cambio.

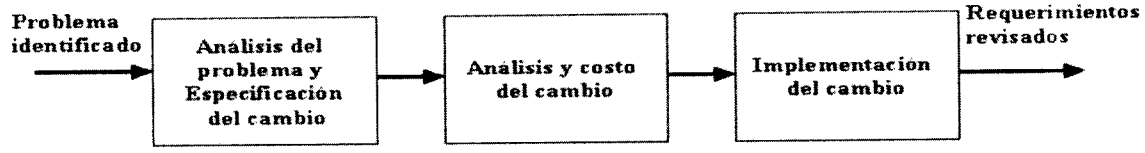

Figura 10: Administración de cambios en los requerimientos.

- **Análisis del problema y especificación de cambios.** Se identifican los problemas en los requerimientos o la propuesta específica de cambios en los requerimientos. En esta etapa, la propuesta de cambio o el problema en los requerimientos, se analizan para verificar que sí es válida, y entonces se realiza una propuesta de cambio de requerimientos más específica.
- **Análisis del cambio y costos.** El efecto de un cambio propuesto es valorado. Para esto, se utiliza la información de rastreo y el conocimiento general de los requerimientos del sistema. El costo de realizar un cambio se estima en términos de las modificaciones hechas al documento de requerimientos y al diseño e implementación del sistema. Una vez completado el análisis, se toma la decisión de proceder o no con el cambio al requerimiento.
- **Implementación de los cambios.** Se modifica el documento de requerimientos y, si es necesario también se modifica el diseño e implementación del sistema. El documento se organiza de tal forma que los cambios puedan llevarse a cabo sin que esto cause la redacción nuevamente del documento.

Si se requiere un cambio de forma urgente en los requerimientos del sistema, se podría modificar el sistema y después modificar el documento de requerimientos. Esto inevitablemente produciría un desfase entre la especificación de requerimientos y la implementación del sistema. Una vez realizados los cambios en el sistema, deben actualizarse los cambios necesarios en el documento de requerimientos, ya que si se deja pasar el tiempo se olvidan o se hacen de forma no consistente con los cambios del sistema.

2.2.5.3 Políticas de Rastreo.

Las políticas de rastreo definen las relaciones entre requerimientos y las relaciones que existen entre los requerimientos con el diseño del sistema. Así mismo, estas políticas definen la forma en que se debe mantener su documentación. Cuando existen cambios en algunos requerimientos, se tiene que rastrear el efecto de estos cambios en otros requerimientos. El rastreo es una propiedad que facilita encontrar requerimientos relacionados, Según Sommerville (Sommerville 1998) existen tres tipos de información de rastreo:

- **La información de rastreo de la fuente.** Relaciona los requerimientos con los usuarios que propusieron los requerimientos. De esta forma cuando un cambio es propuesto, es posible consultar a estos usuarios.
- **La información de rastreo de los requerimientos.** El documento de requerimientos es vinculado con otros requerimientos relacionados. Esta información es utilizada para evaluar de que forma el cambio de un requerimiento afecta a otros.
- **Información de rastreo de diseño.** Vincula los requerimientos a los módulos de diseño. Esta información se utiliza para evaluar el impacto de los cambios hechos a los requerimientos en el diseño e implementación del sistema.

Con frecuencia se han utilizado matrices de rastreos para obtener estos tres tipos de información en el desarrollo de un sistema. En la matriz de rastreo, cada requerimiento está representado por una fila y una columna. Si existe una relación entre requerimientos esta se registra con una letra en una celda en donde se localiza la intersección entre los requerimientos. En la siguiente tabla (Tabla 2) se indica el uso de letras para especificar el tipo de relación. La letra U, indica que el requerimiento en la fila utiliza los recursos del requerimiento indicado en la columna; una R significa que existe una relación débil entre los requerimientos. Las matrices de rastreo son útiles cuando se tiene que administrar un número pequeño de requerimientos, pero son muy difíciles de mantener cuando se trata de sistemas grandes con demasiados requerimientos.

Requerimientos	R1	R2	R3	R4	R5	R6	R7	R8
R1		U	R					
R2			U			R		U
R3	R			R				
R4			R		U			U
R5								U
R6		R		U				
R7								R
R8							R	

Tabla 2: Matriz de Rastreo.

2.2.5.4 Soporte de Herramientas CASE.

La administración de requerimientos comprende el procesamiento de grandes cantidades de información. Las herramientas que se utilizan van desde sistemas especializados de administración de requerimientos, hasta hojas de cálculo y sistemas de bases de datos sencillas. En el mundo existen gran cantidad de herramientas para llevar a cabo la administración de requisitos, entre ellas podemos mencionar Rational RequisitePro y Visual Paradigm, herramientas muy usadas debido a que poseen características que responden a un buen control y administración de los requerimientos (ver epígrafe 1.3.1.5), pero que a la vez constituyen herramientas de software propietario y debido a las características de este tipo de software y teniendo en cuenta que la Universidad de las Ciencias Informáticas, y en específico el Polo Sistemas Geológicos, se encuentra en un proceso de cambio hacia plataformas libres; en este trabajo de tesis investigativa se hará la propuesta de una herramienta de software libre, para llevar a cabo el proceso de administración de requerimientos. La OSRMT v1.3 ("Open Source Requirements Management Tool"), (ver epígrafe 1.3.1.5), es una herramienta de software libre, llevada a cabo por un único desarrollador (aron-smith@hotmail.com) y que no ofrece ningún soporte empresarial (VULCANO 2006).

Esta herramienta la podemos encontrar como propuesta en el documento que corresponde al entregable D6. "Estudio de herramientas de certificación de madurez de procesos de desarrollo" correspondiente al subproyecto 3 "Creación de una suite de herramientas integradas para facilitar la implantación de procesos de desarrollo de software" del Proyecto Vulcano (VULCANO 2006). El objetivo de este subproyecto es facilitar a las empresas españolas un conjunto de herramientas, entre ellas OSRMT, basadas en sistemas *open source (código abierto)*, que mediante mecanismos

de interoperabilidad de sistemas estén integradas y faciliten la implantación de proceso de desarrollo de software.

Se trata de una herramienta de gestión de requisitos, que permite la descripción avanzada de diversos tipos de requisitos y garantiza la trazabilidad entre todos los documentos relacionados con la ingeniería de requisitos (funcionalidades, requisitos, casos de uso, casos de prueba).

***- Características y funcionalidad básica.**

Los diversos módulos integrados en la herramienta son:

- Administración y Configuración.
- Gestión de documentos de ingeniería de requisitos (funcionalidades, requisitos, casos de uso, casos de pruebas).
- Trazabilidad entre documentos de trabajo.
- Informes y estadísticas.

Las funcionalidades ofrecidas por la herramienta son:

- Gestión de requisitos, diferenciando entre:
  - Requisitos.
  - Funcionalidades.
  - Requisitos técnicos.
  - Casos de prueba.
- Trazabilidad entre todos los documentos de trabajo.
  - Requisito-Requisito (control de versiones).
  - Requisito-Requisito (dependencia entre requisitos).
  - Requisito-Funcionalidad.
  - Requisito-Caso de Prueba.
  - Visualización de la matriz de trazabilidad.
  - Árbol de trazabilidad para facilitar las auditorías.

- Gráfico de dependencia entre documentos de trabajo para poder determinar el impacto de un cambio.

- Personalización y configuración.
  - Definición de los atributos de una funcionalidad.
  - Definición de los atributos de un requisito.
  - Definición de los atributos de un caso de prueba.
  - Valores predefinidos para cada usuario (prioridades por defecto, estado por defecto, etc.).
  - Personalización de vistas.
- Representación jerárquica de los documentos de trabajo.
- Definición de casos de prueba mediante pruebas para cada uno de los pasos del caso de uso.
- Descripción de un requisito mediante la secuencia de pasos de su caso de uso.
- Gestión de la configuración.
  - Almacenamiento de quién y cuándo cambia qué.
  - Versionado de los documentos de trabajo.
- Posibilidad de almacenamiento de ficheros binarios adjuntos o hipervínculos.
- Gestión de usuarios.
- Acceso restringido a usuarios registrados.
  - Gestión de privilegios para determinadas tareas.
  - Autenticación LDAP.
- Búsqueda avanzada (filtros, órdenes) sobre los documentos de trabajo registrados.
- Informes y estadísticos.
  - Básicos.
  - Específicos creados por el usuario.
  - A partir de los resultados de búsquedas avanzadas.

- Exportados a HTML o PDF.

- Importar información en XML y mediante líneas de comandos.
- Exportar información en XML y HTML y mediante línea de comandos.
- Herramientas de migración para los diversos cambios de versiones.
- Múltiples idiomas (importación y exportación para dar soporte a diversos idiomas).

*- Integrabilidad Web o con el IDE Eclipse mediante plugin.

Hasta la fecha no es posible acceder a ningún formulario de la herramienta vía Web, ni existe ningún plugin para Eclipse disponible para esta herramienta.

*- Ventajas e Inconvenientes

La herramienta presenta las siguientes ventajas:

- La visualización de requisitos en forma jerárquica es intuitiva y fácil de manejar.
- Existen diversas distribuciones, tanto para un equipo en local como para un servidor de aplicaciones J2EE para permitir desarrollo colaborativo.
- Su licencia es GPL.
- Es un desarrollo basado en Java, por lo que es multiplataforma.
- Las nuevas versiones incorporan un cliente Web para permitir accesos desde internet.
- Como herramienta *open source (código abierto)* de gestión de requisitos no tiene mucha competencia en cuanto a la funcionalidad ofrecida.
- Tiene una buena documentación pese a tratarse de una herramienta muy reciente.
- A pesar de ser llevada a cabo por un único desarrollador, el ritmo de mejoras y nuevas versiones es constante.
- Existen muchas opciones para configurar y personalizar la herramienta a las necesidades concretas de una organización.
- Lleva incorporado un sistema de gestión de la configuración que permite definir líneas base.
- Existe un gran soporte para mantener la trazabilidad entre los documentos.
- Existen mecanismos que facilitan la importación y exportación de la información en XML.

Los principales inconvenientes que se han observado son los siguientes:

- Es un desarrollo llevado a cabo por una persona individual, por lo que existe el riesgo de que no sea sostenible a lo largo del tiempo.
- No existe un soporte empresarial.
- Las nuevas versiones no están planificadas ni se anuncian claramente las mejoras que serán incorporadas. Es posible que las nuevas versiones no sean compatibles con las anteriores.
- No es posible generar automáticamente un documento de requisitos para entregar al cliente.
- Algunas funcionalidades no han sido desarrolladas completamente y están a medias.
- La interfaz de usuario es en ocasiones lenta.
- Se ofrecen pocos mensajes de confirmación y aviso al usuario (la interacción con el usuario es pobre).

Esta herramienta, dentro de las herramientas "open source" que abarca la ingeniería de requisitos, se trata sin duda de una de las mejores funcionalidades que ofrece. Posee muchas opciones de configuración, que permite personalizar la herramienta a las necesidades concretas de una organización, esta aplicación cubre los principales aspectos relacionados con la gestión de requisitos: su registro, definición, categorización, seguimiento y trazabilidad con el resto de documentos de trabajo.

Su principal inconveniente radica en que se trata de un proyecto llevado a cabo de forma individual por un único desarrollador, por lo que se hace preciso de un entorno empresarial que permita dar soporte al proyecto.

2.3 Clasificación de las técnicas de Obtención de Requerimientos.

La obtención de los requerimientos es una tarea compleja que requiere de un gran esfuerzo por parte de los desarrolladores para lograr comprender el sistema y las necesidades de los usuarios. Por tal motivo se hace necesaria la participación de los usuarios del sistema para lograr identificar el mayor número de requerimientos del sistema y aproximarse a una descripción completa del sistema. La tarea fundamental para lograr una completa especificación de requerimientos, es la obtención de los requerimientos, por lo que se deben seleccionar adecuadamente las técnicas apropiadas para este fin.

Haciendo uso de un estudio realizado por Juan Carlos Medina Martínez del Centro de Investigación y de Estudios Avanzados del IPN en México, para establecer una clasificación teniendo en cuenta diversos puntos de vistas; la siguiente tabla (Tabla 3) muestra una selección de dicha clasificación, mostrando las clasificaciones de las técnicas de obtención de requerimientos que forman parte de esta propuesta de tesis. La clasificación se realizó tomando en cuenta la participación de los usuarios/clientes del sistema en la obtención de requerimientos, así como, el número de usuarios que participan y las veces que la técnica puede ser aplicada en la actividad de obtención de requerimientos. Las clasificaciones con las siguientes (Medina 2004):

- **Activa:** Los usuarios/clientes participan activamente en la obtención de los requerimientos.
- **Pasiva:** Los usuarios/clientes no participan o son observados por los desarrolladores.
- **General:** Todos los usuarios/clientes participan en la obtención de requerimientos.
- **Parcial:** Solo algunos usuarios participan en la obtención de requerimientos.
- **Iterativas:** La técnica puede ser aplicada varias veces para poder obtener los requerimientos
- **Definitiva:** La técnica es aplicada una sola vez.

Técnica	Activa	Pasiva	General	Parcial	Iterativa	Definitiva
1. Entrevista	X			X	X	
2. Cuestionario	X			X		X
3. Punto de vista	X		X			X
4. Observación		X		X	X	
5. Escenarios	X		X		X	
6. Run Use Case Workshop	X		X			X
7. Análisis de Interesados	X			X		X
8. Demostración de tareas	X			X		X
9. Enfoque grupal	X		X		X	
10. Talleres futuros.	X		X		X	
11. Estudio de Documentos		X			X	X

Tabla 3: Clasificación de las técnicas de obtención de requerimientos.

Otra clasificación que se puede realizar es seleccionando las técnicas de acuerdo a su aplicación en las tareas que se desarrollan en la actividad de obtención de requerimientos. La tabla 4 muestra esta clasificación. Por ejemplo, para tener conocimiento del problema y entender el contexto del sistema a desarrollar se pueden aplicar la observación, entrevista, estudio de documentos o la demostración de tareas. Con un conocimiento de la situación problemática a resolver se continúa con la obtención de información que definan las características del sistema mediante la entrevista, cuestionarios, y el estudio de documentos, estas técnicas se pueden emplear para recolectar suficiente información para lograr una definición inicial de las necesidades del cliente. A partir de la información obtenida se puede definir los límites del sistema mediante la técnica de escenarios, o utilizando técnicas dirigidas a un grupo de usuarios mediante reuniones con el fin de analizar el sistema y definir su alcance mediante las técnicas Run Use Case Workshop, enfoque grupal y talleres futuros. La identificación de usuarios puede llevarse a cabo mediante las técnicas de puntos de vista, análisis de usuarios interesados y demostración de tareas, todas estas técnicas tienen la característica que se aplican a usuarios de manera individual. Obtenida la información suficiente se pueden definir los requerimientos del sistema mediante los puntos de vista, escenarios, Run Use Case Workshop, enfoque grupal y talleres futuros (Medina 2004).

Técnica	Comprender el problema	Recolectar información	Definir límites	Identificar usuarios	Recolectar requerimientos
1. Entrevista	X	X			
2. Cuestionario		X			
3. Punto de vista				X	X
4. Observación	X	X			
5. Escenarios			X		X
6. Run Use Case Workshop			X		X
7. Análisis de Interesados				X	
8. Demostración de tareas	X			X	
9. Enfoque grupal			X		X
10. Talleres futuros.			X		X
11. Estudio de Documentos	X	X			

Tabla 4: Clasificación de las técnicas en las actividades de la obtención de requerimientos.

Analizando las características de cada técnica podemos agruparlas de acuerdo a la similitud en su aplicación y en su estructura interna en seis grupos diferentes y dos más que resultan de la combinación de otros (Medina 2004).

- **Grupo 1:** Puede ser utilizada para cualquier enfoque ya sea funcional u orientado a objeto, la participación de los usuarios no es directamente, los desarrolladores se encargan de observar las tareas que realizan los usuarios.
- **Grupo 2:** Una técnica clásica de obtención de requerimientos y adecuada en todo inicio de un proyecto de software, en la que el desarrollador entabla directamente una comunicación verbal con los interesados del sistema para recolectar información que le ayudará a definir sus necesidades. También suele aplicarse en las etapas restantes del proceso de desarrollo, es decir, el análisis, la validación y especificación de los requerimientos.
- **Grupo 3:** Este grupo de técnicas, se caracteriza por la aplicación a un grupo de usuarios y el éxito depende de la habilidad del facilitador o moderador de la reunión.
- **Grupo 4:** Este grupo de técnicas se utiliza para sistemas que son desarrollados bajo el enfoque orientado a objetos, principalmente por la aplicación de los casos de uso que son la parte esencial de este enfoque, para su aplicación es necesario contar con información recabada sobre la operación del sistema.
- **Grupo 5:** Esta técnica se basa en la obtención de información del sistema mediante la aplicación de cuestionarios, suele combinarse con otras técnicas para lograr resultados importantes. Es útil aplicarla al inicio de todo proceso de desarrollo para obtener un contexto del sistema.
- **Grupo 6:** La participación de los usuarios en este grupo es nula, los desarrolladores son los encargados para lograr obtener un conocimiento del sistema. Esta técnica debe ser utilizada como auxiliar de otras para obtener un mayor conocimiento del contexto del sistema, sobre todo si ya existe un sistema previo y se requiere una actualización o mejora.
- **Grupo 7:** Este grupo combina las características del grupo de uno y dos. Utiliza la observación y mediante preguntas se cuestiona al usuario para aclarar dudas de la realización de sus tareas y de esta manera lograr el conocimiento del contexto del sistema.

- **Grupo 8:** Es el resultado de la combinación de características del grupo dos y tres, suele funcionar de una entrevista dirigida por el desarrollador hacia un grupo de usuarios o de manera particular, en busca de información o datos que le permitan definir el sistema.

Todos los grupos pueden combinarse para lograr una amplia captación de información y datos que ayuden a comprender el funcionamiento del sistema y lograr una definición de sus límites. Es difícil definir que técnica utilizar y es aún más difícil recomendar las técnicas que se deben seguir para el desarrollo de los sistemas, debido a que estos siempre tendrán características diferentes aunque se encuentre dentro de un mismo contexto. Por lo que se recomienda usar una combinación de los grupos mencionados anteriormente pero tomando en consideración la disponibilidad de los usuarios y el ambiente en donde operará el sistema que se va a desarrollar, además, si el sistema será nuevo o una mejora de uno ya existente. En la figura 11 se muestran los agrupamientos de las técnicas de obtención de requerimientos.

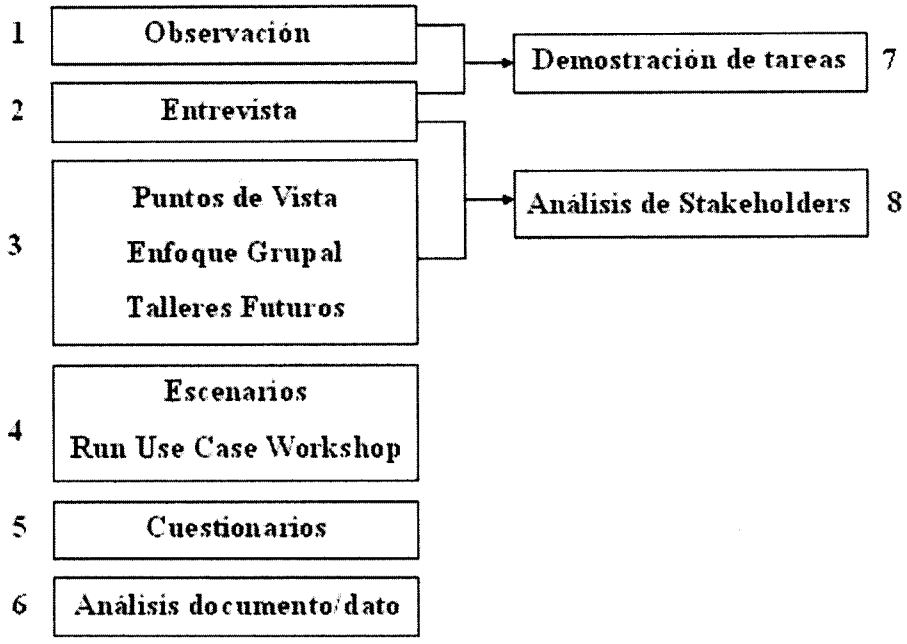

Figura 11: Agrupamiento de técnicas de obtención de requerimientos.

2.4 Análisis comparativo de las Técnicas de Obtención de Requerimientos.

Básicamente la comparación realizada entre las técnicas de obtención de los requerimientos es realizada de acuerdo a las ventajas y desventajas que cada técnica ofrece en la aplicación de sistemas de software interactivos, es decir, en sistemas donde la participación del usuario es necesaria para el funcionamiento del sistema. La mayoría de las técnicas requieren la participación de los usuarios para lograr obtener la mayor parte de los requerimientos del sistema.

Técnica	Ventajas	Desventajas
Entrevista.	• Es utilizada como punto de partida en la obtención de los requerimientos. • Se logra conocer la opinión del cliente y los usuarios. • Se conoce la organización de la empresa. • Obtiene las metas del sistema. • Se obtiene un entendimiento general del problema. • Puede combinarse con otras técnicas.	• Creen perder el poder si revelan lo que saben. • No existe confianza entre los usuarios y los desarrolladores. • Utilizan diferente terminología. • Difícil de expresar. • Sabotear la entrevista por falta de cooperación. • Expresar satisfacción de la forma en que se hacen las cosas y no desea cambios. • Da mucha información irrelevante.
Cuestionario	• Se obtiene una gran cantidad de información a través del usuario. • Son flexibles. • Permite obtener datos que pueden medir resultados de forma escalar. • Permite combinarse con otras técnicas.	• Se obtiene un gran volumen de información que requiere de análisis. • Se obtiene información redundante y en ocasiones incompleta. • Se requiere de experiencia en el dominio para tratar de comprender el comportamiento de los encuestados.
Punto de Vista	• Permite estimular la aprobación de ideas. • Se crea un ambiente de confianza.	• Existen personas que no les gusta hablar en público y que posiblemente tengan buenas ideas.

	• Puede combinarse con otras técnicas.	• Se requiere de un buen facilitador que asegure el éxito de la técnica. • Puede haber requerimientos externos no encontrados.
Observación	• Contacto directo con el medio ambiente. • Se detectan problemas. • Puede combinarse con otras técnicas como el prototipado y los Casos de Uso	• Tiempo prolongado y difícil de permanecer en calidad de observador. • Los usuarios presentan incomodidad al ser observados.
Escenarios	• Fácil de comprender y criticar un escenario. • Describe el estado inicial del escenario. • Describe el flujo de eventos. • Descubre excepciones y soluciones. • Descubre como llevar a cabo situaciones paralelas. • Describe el estado final del escenario.	• Toman tiempo de acuerdo a la complejidad del sistema. • Las personas que modelen el sistema deben conocerlo previamente.
Run Use Case Workshop	• Permite la participación directa de los usuarios para definir escenarios. • Permite crear un ambiente adecuado para la participación de los demás usuarios.	• Existe el riesgo de caer en grandes diferencias de interpretación o contradicciones. • Los talleres pueden caer en ser pesados a los usuarios y generar desaliento y poca participación. • Se pueden deducir requerimientos con diferente percepción.
Análisis de Stakeholders	• Se conoce la jerarquía organizacional del dominio. • Se obtienen un gran número de requerimientos desde diferentes	• Diferentes percepciones de los requerimientos requiere de secciones de negociación y homogenización.

	puntos de vistas. • Se logra la participación de diferentes sectores del negocio.	• Se debe establecer una agenda acordada que no afecte a ningún participante. • Requiere de tiempo si la empresa es grande organizacionalmente.
Demostración de tareas.	• Al usuario le es más fácil demostrar como hacen sus tareas. • Se logra una mayor interacción con el usuario. • Se pueden encontrar detalles omitidos u olvidados por los usuarios en otras técnicas. • Se logra un mayor entendimiento del sistema.	• Se requiere de la participación voluntaria del usuario. • El usuario se incomoda cuando se le cuestiona sobre su trabajo. • Requiere demasiado tiempo. • El usuario trata de hacer lo mejor posible su tarea y omite posibles defectos.
Enfoque grupal.	• Se detectan los sentimientos de los usuarios hacia el sistema. • Provee un medio para hacer surgir ideas de los demás participantes. • Se logra obtener los límites y restricciones del sistema.	• Se requiere de un buen moderador para que la reunión tenga éxito. • Puede tomar tiempo si no se dirige la reunión hacia objetivos específicos. • Pueden encontrarse grupos políticos de usuarios que afecten sus intereses personales, más que a los interesados de la empresa.
Talleres futuros	• Provee un medio para hacer surgir ideas de los demás participantes. • Se logra obtener los límites y restricciones del sistema. • Permite visualizar el estado final del nuevo sistema y prever las afectaciones o restricciones que tendrá.	• Se puede exagerar en la visualización de las funciones que tendrá el futuro sistema. • Existe el riesgo de que los usuarios tengan otra percepción del sistema a como lo delimitan los desarrolladores
Análisis de	• Permite realizar un planteamiento	• No hay participación activa del

documentos/datos.	del sistema en base de formas, archivos y datos que son utilizados en el sistema actual. • Requiere solo de la participación del desarrollador sin afectar a los usuarios en sus tareas.	usuario. • Se corre el riesgo por suposiciones de los desarrolladores en el análisis de documentos y formas.

Tabla 5: Cuadro comparativo de las técnicas de obtención de requerimientos.

La tabla 4 muestra las ventajas y desventajas que cada técnica ofrece en la actividad de obtención de los requerimientos del sistema, de esta forma permite seleccionar de manera rápida la técnica o técnicas a aplicar en el desarrollo de software, considerando el contexto actual del sistema y las características de los usuarios involucrados en el desarrollo.

2.5 Conclusiones Parciales.

En el desarrollo de este capítulo han quedado expuestas una serie de técnicas extraídas de diversas bibliografías, las cuales fueron el resultado de la investigación desarrollada para la elaboración de este trabajo de tesis. La comparación establecida entre dicha recopilación de técnicas y las técnicas utilizadas en el Polo Sistemas Geológicos de la UCI, contribuyó en gran medida al planteamiento de una nueva guía para desarrollar de una forma más óptima el Proceso de levantamiento de Requisitos en dicho Polo, dando cumplimiento al objetivo por el cual se ha llevado a cabo este trabajo de tesis.

Como parte de esta investigación en la encuesta realizada a líderes del Polo (Ver Anexo 4), se obtuvo como resultado un pobre uso de las técnicas relacionadas con la etapa de Obtención de Requerimientos, lo que constituyó la base para el planteamiento de nuevas técnicas que fueron planteadas como solución de este trabajo de diploma en dicho Polo con el objetivo de lograr un proceso de desarrollo de software con mayor calidad.

CONCLUSIONES

A lo largo de la investigación realizada con el objetivo de elaborar una nueva guía para el proceso de levantamientos de requerimientos en el Polo Sistemas Geológicos se llegó a la conclusión de que:

- Existe un vago concepto de la importancia de la ingeniería de requerimientos, ya que no se le da la importancia requerida y en la mayoría de las situaciones éste proceso no se desarrolla con la calidad debida. Situación que debe mantenerse si no se hace un llamado a la toma de conciencia para que esta etapa sea desarrollada de la manera más óptima posible.
- La falta de personal capacitado y la mala capacitación del personal unido a la falta de experiencia hacen que esta etapa de la Ingeniería de Software no se desarrolle de la forma debida, trayendo consigo retraso en la entrega del producto, debido a un continuo incompleto levantamiento de requerimientos.
- El estudio riguroso de diferentes técnicas y metodologías que pudieran ser aplicadas a dicho Polo permitió el planteamiento de nuevos principios que forman parte de la guía a aplicar en el Polo Sistemas Geológicos de la UCI, la cual constituye el objetivo fundamental que rigió esta investigación. Las técnicas que forma parte de esta guía quedaron registradas bajo las siglas IR: GEPSiG (Ingeniería de Requerimientos: Guía Enfocada al Polo Sistemas Geológico).

RECOMENDACIONES

Después de un análisis detallado de diversas fuentes de información que contribuyeron a la recopilación de un conjunto de técnicas y metodologías que guiarán el proceso de levantamiento de requerimientos en el Polo Sistemas Geológicos de la Facultad 9 de la UCI, se hace necesario que se listen un grupo de recomendaciones que permitan la mejora continua del proceso de levantamiento de requerimientos en dicho Polo.

- Llevar a cabo una capacitación más profunda a líderes y personal del equipo de trabajo del Polo Sistema Geológicos sobre todo lo relacionado con el proceso de levantamiento y validación de requerimientos, con el fin de que posean una cultura general sobre el tema y se tome conciencia de la importancia de este proceso para poder desarrollar un producto con la calidad requerida.
- Realizar un estudio más detallado de las técnicas y metodologías existentes para el levantamiento de requerimiento con el objetivo de plantear otras técnicas que no hayan sido reflejadas en la propuesta de solución y que puedan contribuir a la disminución del tiempo de desarrollo y aumento de la calidad del producto resultante.
- Analizar y aplicar consecuentemente las técnicas planteadas en IR: GEPSiG en proyectos de dicho Polo.
- Realizar un seguimiento del trabajo de estos proyectos bajo este nuevo enfoque y velar por los resultados de su aplicación.
- De resultar exitosa la aplicación de IR: GEPSiG en estos proyectos, proponer su extensión a todos los proyectos productivos que conforman el Polo.
- Estudiar la posible aplicación de esta guía en otros Polos y proyectos de la Universidad.

REFERENCIAS BIBLIOGRÁFICAS

(RequisitePro 2007) RequisitePro, Rational Co., http://www.rational.com.

(VORDTools 1996) VODRTools, Ian Sommerville, http://www.comp.lancs.ac.uk/computing/resources/SE6

(Viller 1998) Viller S., Social Analysis for Software Engineering, Technical Report 1998, http://www.comp.lancs.ac.uk/computing/research/cseg/98_rep.html

(Burg 1997) Burg J. F. M., Van de Riet R. P., Análisis Informal Requirements Specifications: A First step towards Conceptual Modelling, http://www.cs.vu.nl/jmfburg/publications.html

(OSRMT 2006) OSRMT 1.3, Open Source Requirements Management Tool, http://www.osrmt.com.

(TSG 1995) TSG. *The CHAOS Report*. The Standish Group, 1995.

(Arias 2005) Michael Arias Chávez. La Ingeniería de Requerimientos y su importancia en el desarrollo de proyectos de software, Revista InterSedes, 2005, Volumen IV (Número 10), 1-13. Disponible en www.intersedes.ucr.ac.cr

(INCOSE 2006) INCOSE, INCOSE Requirements Management Tools Survey, http://www.paper-review.com/tools/rms/read.php

(VULCANO 2006) ATOS, ITA, TID, GERMINUS, UPM, YACO, ANDAGO. Estudio de herramientas de certificación de madurez de proceso de desarrollo. 2006. Disponible en http://www.ines.org.es/vulcano/wp-content/uploads/2007/09/d6-estudio-de-herramientas-de-certificacion-tid.pdf

(Sommerville 2005) Ian Sommerville, Ingeniería del Software. Séptima Edición. Madrid, Pearson Education Limited, 2005.Pág. 6,108, 122, 126 (712).

(Pressman 2002) R. S. Pressman, Ingeniería del Software, un enfoque práctico. Quinta Edición. Mc Graw Hill/Interamericana de España, S.A, 2002. Pág. 14, 171, 174.

(Lawrence 2002) Shari Lawrence Pfleegeer, Ingeniería de Software, teoría y práctica, Prentice Hall, 2002.

(Medina 2004) Juan Carlos Medina Martínez. Análisis Comparativo de Técnicas, Metodologías y Herramientas de Ingeniería de Requerimientos. Tesis Maestría. Centro de Investigación de Estudios

Avanzados del IPN. México, D.F. 2004. Pág. 25. Disponible en http://www.cs.cinvestav.mx/Estudiantes/TesisGraduados/2004/.

(Del Toro 2002) José Miguel del Toro Bonilla. Un entorno metodológico de Ingeniería de Requisitos para Sistemas de Información. Tesis Doctoral. Universidad de Sevilla. Sevilla 2002. Pág 9.

(Richard 1997) Ricard, IEEE Software Requirement Engineering, Second Edition. Thayer y Merlin Dorfman, IEEE Computing Society, New York, NY. 1997.

(Loucopoulos 1995) Loucopoulos, P; Karakostas, V. (1995); System Requirements Engineering McGraw-Hill, 1995.

(Mcdonald Landazuri 2005) Bárbara A. Mcdonald Landazuri. Definición de Perfiles en Herramientas de Gestión de Requisitos. Tesis Doctoral. Universidad Politécnica de Madrid. Madrid, 2005.

(Macaulay 1996) Macaulay L. A., Requirements Engineering, Springer-Verlag, 1996.

(Jacobson, Booch, Rumbaugh 2000) Jacobson Ivar, Booch Grady, Rumbaugh James, El Proceso Unificado de Desarrollo de Software, Addison Wesley, 2000.

(Kendall 1991) Kendall E. Kenenth, Kendall E. Julie, Análisis y Diseño de Sistemas, primera edición, Prentice Hall Hispanoamericana, 1991.

(Sommerville 1998) Ian Sommerville, Software Engineering, Wiley, 1998.

(Kotonya, Sommerville 2000) Kotonya Gerald, Sommerville Ian, Requirements Engineering, processes and techniques, Wiley, 2000.

(Craig 1999) Craig L., UML y Patrones, Introducción al Análisis y Diseño Orientado a Objetos, Pearson 1999.

(Lauesen 2002) Lauesen Soren, Software Requirements, Styles and Techniques, first edition, Eddison Wesley a Pearson Education Book, 2002.

(Templeton 1994) Templeton J.F., The Focus Group: A strategic Guide to Organizing, Conducting and Analyzing the Focus Group Interview, Probus publishing Co., 1994.

(Morris, Tamm 1993) Morris D., Tamm B., Concise Encyclopedia of Software Engineering, Pergamon Press. 1993.

BIBLIOGRAFÍA

(Sommerville 2005) Ian Sommerville, Ingeniería del Software. Séptima Edición. Madrid, Pearson Education Limited, 2005.

(Pressman 2002) R. S. Pressman, Ingeniería del Software, un enfoque práctico. Quinta Edición. Mc Graw Hill/Interamericana de España, S.A, 2002.

(Lawrence 2002) Shari Lawrence Pfleegeer, Ingeniería de Software, teoría y práctica, Prentice Hall, 2002.

(Del Toro 2002) José Miguel del Toro Bonilla. Un entorno metodológico de Ingeniería de Requisitos para Sistemas de Información. Tesis Doctoral. Universidad de Sevilla. Sevilla 2002.

(Richard 1997) Ricard, IEEE Software Requirement Engineering, Second Edition. Thayer y Merlin Dorfman, IEEE Computing Society, New York, NY. 1997.

(Medina 2004) Juan Carlos Medina Martínez. Análisis Comparativo de Técnicas, Metodologías y Herramientas de Ingeniería de Requerimientos. Tesis Maestría. Centro de Investigación de Estudios Avanzados del IPN. México, D.F. 2004. Disponible en http://www.cs.cinvestav.mx/Estudiantes/TesisGraduados/2004/.

(Mcdonald Landazuri 2005) Bárbara A. Mcdonald Landazuri. Definición de Perfiles en Herramientas de Gestión de Requisitos. Tesis Doctoral. Universidad Politécnica de Madrid. Madrid, 2005.

(Maciaszek 2001) Maciaszek A. Leszek, Requirements Análysis and System Design, Developing Information Systems with UML, First Edition, Addison Wesley, 2001.

(Macaulay 1996) Macaulay L. A., Requirements Engineering, Springer-Verlag, 1996.

(Jacobson, Booch, Rumbaugh 2000) Jacobson Ivar, Booch Grady, Rumbaugh James, El Proceso Unificado de Desarrollo de Software, Addison Wesley, 2000.

(Checkland, Scholes 1990) Checkland P., Scholes J., Soft System Methodology in Action, Wiley & Sons, 1990.

(Roger 1997) P. S. Roger, Ingeniería del software, un enfoque práctico, Cuarta Edición, McGraw Hill, 1997

(Burg 1997) Burg J. F. M., Linguistics Instruments in Requirements Engineering, Thesis Doctoral, Vrije Universiteit, IOS Press, 1997.

(Loucopoulos 1995) Loucopoulos, P; Karakostas, V. (1995); System Requirements Engineering McGraw-Hill, 1995.

(GAO 1979) Contracting for Computer Software Development: Serious Problems Require Management Attention to Avoid Wasting Additional Millions. Report FGMSD–80–4, U. S. Government Account Office, Noviembre 1979. Este documento es difícil de encontrar, pero está comentado y referenciado, entre otros, en (Davis 1993), (Christel y Kang 1992) y (Goguen 1994).

(Goguen 1994) J. A. Goguen. Requirements Engineering as the Reconciliation of Social and Technical Issues. En *Requirements Engineering: Social and Technical Issues*, páginas 165–199. Academic Press, 1994.

(Davis 1993) A. M. Davis. *Software Requirements: Objects, Functions and States*. Prentice–Hall, 2a edición, 1993.

(Christel y Kang 1992) M. G. Christel y K. C. Kang. Issues in Requirements Elicitation. Technical Report CMU/SEI–92–TR–12, Software Engineering Institute, Carnegie Mellon University, 1992. Disponible en http://www.sei.cmu.edu.

(Kendall 1991) Kendall E. Kenenth, Kendall E. Julie, Análisis y Diseño de Sistemas, primera edición, Prentice Hall Hispanoamericana, 1991.

(Sommerville 1998) Ian Sommerville, Software Engineering, Wiley, 1998.

(Kotonya, Sommerville 2000) Kotonya Gerald, Sommerville Ian, Requirements Engineering, processes and techniques, Wiley, 2000.

(Craig 1999) Craig L., UML y Patrones, Introducción al Análisis y Diseño Orientado a Objetos, Pearson 1999.

(Lauesen 2002) Lauesen Soren, Software Requirements, Styles and Techniques, first edition, Eddison Wesley a Pearson Education Book, 2002.

(Templeton 1994) Templeton J.F., The Focus Group: A strategic Guide to Organizing, Conducting and Analyzing the Focus Group Interview, Probus publishing Co., 1994.

(Morris, Tamm 1993) Morris D., Tamm B., Concise Encyclopedia of Software Engineering, Pergamon Press. 1993.

(VULCANO 2006) ATOS, ITA, TID, GERMINUS, UPM, YACO, ANDAGO. Estudio de herramientas de certificación de madurez de proceso de desarrollo. 2006. Disponible en http://www.ines.org.es/vulcano/wp-content/uploads/2007/09/d6-estudio-de-herramientas-de-certificacion-tid.pdf

ANEXOS

Anexo 1

Entrevista a personal de Calidad UCI y líderes del Polo Sistemas Geológico:

1- ¿Cómo se debe desarrollar el proceso de la Ingeniería de Requerimientos en la UCI?

2- ¿Se desarrolla este proceso de la forma correcta?

3- ¿Qué documentación se utiliza?

4- ¿Quiénes son los encargados de desarrollar el proceso de levantamiento de requisitos?

5- ¿Cómo se lleva a cabo la capacitación del personal involucrado en este proceso?

6- ¿Se le asigna el tiempo estimado a este proceso?

7- ¿Cuáles son los principales problemas detectados a la hora de realizar el levantamiento de requerimientos?

8- ¿Se utiliza alguna herramienta CASE para efectuar la administración de requerimientos?

Anexo 2

ENTREVISTA

1- ¿Cómo se lleva a cabo el proceso de la Ingeniería de Requerimientos en el Polo?

2- ¿Cuáles son lo principales problemas detectados a la hora de realizar el levantamiento de requisitos?

3- ¿Cuáles cree usted que sean las causas que provocan estos problemas?

4- ¿Qué consecuencias ha traído para el Polo estos problemas detectados?

Anexo 3

El modelo en cascada, define un enfoque secuencial del proceso de desarrollo, separado en etapas y cada etapa es fundamental en el desarrollo (ver figura 1). Cada etapa debe completarse antes de comenzar la siguiente etapa. El problema que presenta este modelo es debido a la separación de las actividades en etapas, en cada fase el resultado es uno o más documentos aprobados. En la práctica estas actividades del desarrollo de un producto de software están entrelazadas y requieren de continuas iteraciones, además si un error no es detectado al principio de la etapa, puede ser desastroso encontrarlo en etapas posteriores. Es difícil dejar definidos los requerimientos en la primera etapa del desarrollo, como lo expresa el modelo, esto podría generar problemas a la hora de encontrar nuevos requerimientos en las últimas etapas y no fuera posible regresar a la primera etapa para agregarlos. Para el modelo en cascada el cliente podría tener una versión operativa del sistema sólo hasta alcanzar las etapas finales del desarrollo (Sommerville 1998).

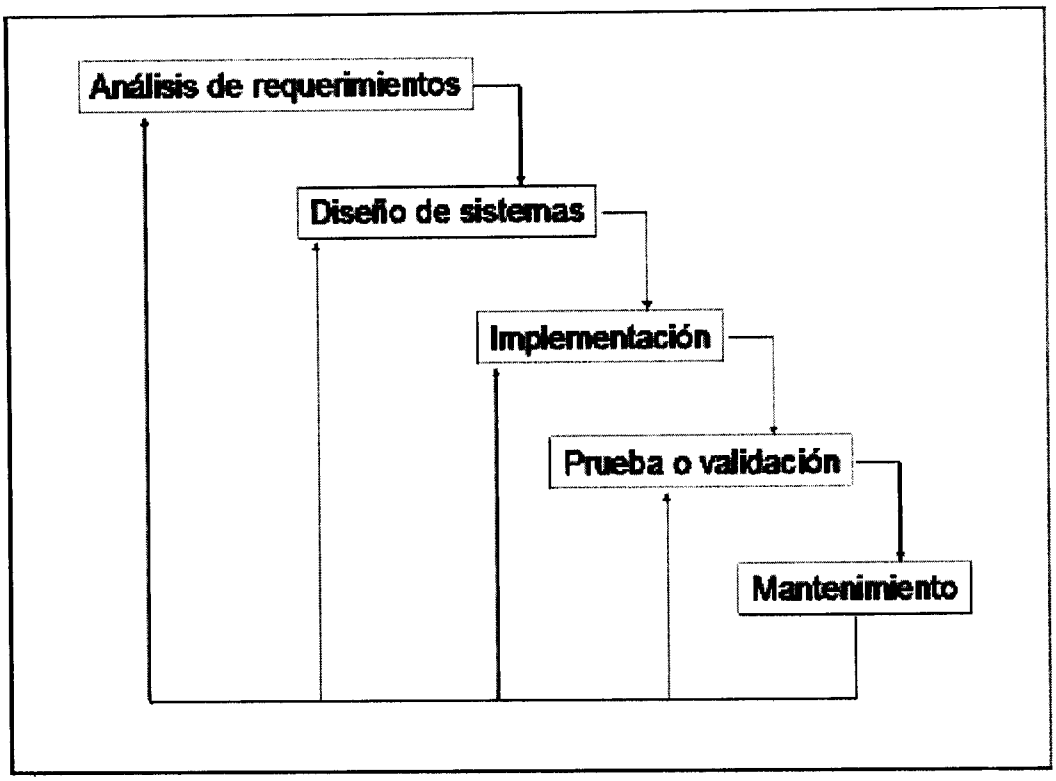

Figura 1. Modelo en Cascada.

Anexo 4

ENCUESTA

De las técnicas relacionadas a continuación, marque con una "X" las utilizadas en el Polo para desarrollar el proceso de Ingeniería de Requerimientos.

1- Obtención de Requerimientos.

_X_ Entrevistas.

_X_ Cuestionarios.

__ Metodología Joint Application Development.

__Puntos de Vistas.

_X_Observación.

__Escenarios.

__Talleres de Trabajo basados en Casos de Uso (Run Use Case WorkShop).

__Análisis de usuarios interesados (Stakeholders).

__Demostración de tareas.

__Enfoque Grupal (Focus Goups).

__Talleres Futuros (Future Workshops).

_X_Estudio de Documentos/Datos.

2- Análisis de Requerimientos.

_X_Identificación de requerimientos.

_X_Lista de verificación.

_X_Matriz de dependencia.

_X_Negociación de requerimientos.

3- Especificación de Requerimientos.

_X_Lenguaje Natural.

_X_Notaciones Gráficas.

__Especificaciones matemáticas.

4- Validación de Requerimientos.

_X_Criterios de validación.

_X_Revisiones del documento de requerimientos.

_X_Escenarios.

_X_Construcción de prototipos.

_X_Generación de casos de pruebas.

_X_Análisis de consistencia automático.

5- Administración de Requerimientos.

_X_Evolución de los requerimientos.

_X_Administración del cambio en los requerimientos.

_X_Políticas de rastreo.

_X_Soporte de herramientas CASE.

Anexo 5

Ejemplo de lista de verificación.

ID	Entendible	Consistente	No ambiguo	Completo	Verificable	Correcto	Modificable
RF1	si	si	si	no	si	no	Si
RF1.1	si	si	si	si	si	si	Si
RF1.2	si	si	si	si	si	si	Si
RF2	si	Si	si	no	si	si	Si
RF2.1	no	Si	no	no	si	si	Si
RF2.2	no	Si	no	no	si	si	Si
RF2.3	no	Si	si	si	si	si	Si

Figura 2. Lista de verificación de calidad en los requerimientos.

Anexo 6

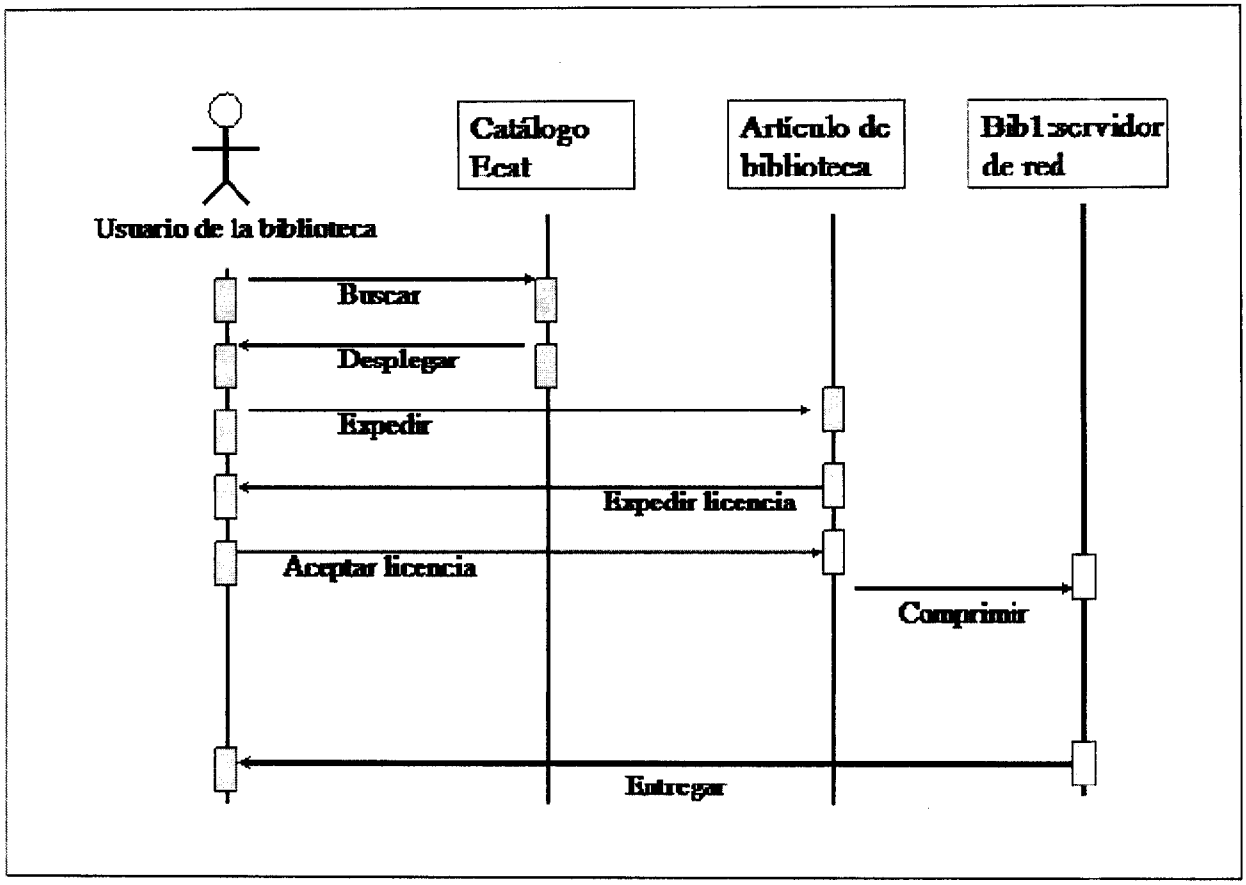

Figura 3. Diagrama de secuencia de un sistema de expedición de artículos electrónicos.

GLOSARIO

Análisis orientado a objetos: es un método de análisis que examina los requisitos desde la perspectiva de las clases y objetos que se encuentran en el vocabulario del dominio del problema.

CASE: Ingeniería de Software Asistida por Computadora.

Calidad: Cumplir sistemáticamente con los requerimientos, para satisfacer las necesidades y expectativas de los clientes y usuarios.

Cliente: Persona que utiliza con asiduidad los servicios de un profesional. Todo aquel que es impactado, recibe, utiliza o se beneficia con las actividades o los servicios de un determinado producto.

DER (Documento de especificación de requerimientos): Este documento contiene la descripción precisa de los requerimientos, incluye modelos de representación que agregan detalles al sistema mostrándolos desde diferentes perspectivas.

DDR (Documento de definición de requerimientos): Se encuentran descritos los requerimientos iniciales del sistema que se va a desarrollar en lenguaje natural.

Estándares: Normas de desempeño definidas para una actividad, un proceso, un producto o un servicio.

Fase: Período de tiempo entre dos hitos principales de un proceso de desarrollo.

GPL (General Public License- Licencia Pública General): La GNU está diseñada para garantizar la libertad entre los usuarios entre los usuarios de compartir y modificar en software. Esta licencia se aplica en la mayoría de los programas realizados por la Free Software Fundation (FSF Fundación del Software Libre) y en cualquier otro programa en los que los autores quieran aplicarla.

Hardware: Conjunto de los componentes que integran la parte material de una computadora.

IEEE: Instituto de Ingenieros Eléctricos y Electrónicos.

Interfaces: Conexión física y funcional entre dos aparatos o sistemas independientes. Artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual, una interacción amigable con un sistema informático.

J2EE: Es una plataforma de programación para desarrollar y ejecutar software de aplicación en lenguaje de programación Java con arquitectura de N niveles distribuidos, basándose ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones.

LDAP (Lightweight Directory Access Protocol): Protocolo de acceso a directorios ligeros; protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red.

Metodología: Se refiere a los métodos de investigación que se siguen para alcanzar una gama de objetivos en una ciencia.

Producto: Software de computadora que diseñan y construyen los ingenieros del software. Esto abarca programas que se ejecutan dentro de una computadora de cualquier tamaño y arquitectura, documentos que comprenden formularios virtuales e impresos y datos que combinan números y texto y también incluyen representaciones de audio y video e imágenes.

Proceso: Conjunto de pasos ordenados y relacionados entre sí a través de los cuales se convierten los insumos en producto o resultados. Marco de trabajo de las tareas que se requieren para construir software de alta calidad. Proporciona una interacción entre los usuarios y diseñadores, entre los usuarios y las herramientas de desarrollo, y entre los diseñadores y las herramientas de desarrollo (tecnología).

Proyecto: Conjunto de actividades interrelacionadas, con un inicio y una finalización definida, que utiliza recursos limitados para lograr un objetivo deseado.

Sistemas: Conjunto de cosas que relacionadas entre sí ordenadamente contribuyen a determinado objeto.

Software: Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora. Es un transformador de información, produciendo, gestionando, modificando, adquiriendo, mostrando o transmitiendo información que puede ser tan simple como un bit, o tan complejo como una presentación en multimedia. Como vehículo utilizado para hacer del producto, el software actúa como la base del control de la computadora (sistemas operativos), la comunicación de información (redes), y la creación y control de otros programas (herramientas de software y entornos).

Técnica: Procedimiento o grupo de procedimientos que tienen el fin de obtener un resultado específico sin importar el campo en donde nos estemos desenvolviendo (arte, tecnología o ciencia).

COBOL (COmmon Budiness – Oriented Language- Lenguaje común orientado a objetos): Lenguaje creado en el año 1960 con el objetivo de crear un lenguaje de programación universal que pudiera ser usado en cualquier ordenador.

UML (Unified Modeling Language- Lenguaje de modelado unificado.**):** Lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por OMG (Object Management Group). Lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software.

Usuario: Individuo u organización que interactúa con un sistema.