Universidad Nacional Autónoma de México

Facultad de Contaduría y Administración

La Ingeniería de Requerimientos como factor clave para el éxito de los proyectos de desarrollo de software

Tesis Profesional que para obtener el título de
Licenciado en Informática presenta

Ma. Teresa Ventura Miranda

Director de Tesis
Dr. Ricardo Rivera Soler

México, D.F., 2002

INTRODUCCIÓN

El presente trabajo está organizado en seis capítulos. El primero de ellos: Marco Problemático, ofrece un panorama de la situación actual de los requerimientos en el desarrollo de sistemas, y que justifica en gran parte la realización y utilidad de este trabajo. En esta parte se presentan los resultados de un cuestionario aplicado a personas relacionadas con el desarrollo de software en México, y en el cual se obtuvieron interesantes resultados.

El segundo capítulo, Marco Teórico, brinda un resumen de las principales fuentes de información utilizadas en el trabajo, las cuales fueron de diversos tipos y conforman la sustentación teórica. Como se podrá observar fueron consultadas diversas fuentes como son: libros, revistas, material de Internet, seminarios, cursos, entre otras.

EL Marco Conceptual, tercer capítulo presenta conceptos e información relacionada con el tema de Ingeniería de Requerimientos, su problemática, las técnicas existentes, así como el punto de vista de diversos autores al respecto. Es resultado del análisis del material comprendido en el Marco Teórico. Además se integraron otras técnicas y recomendaciones que se aplican en la práctica y que se consideraron de gran aportación en este trabajo.

La cuarta parte, Marco Metodológico comprende la comprobación de la hipótesis planteada en el trabajo, con base en el análisis de los resultados obtenidos con el instrumento de recopilación de información que permitió obtener el punto de vista de diversos encuestados. En este capítulo se presentan estadísticas y conclusiones de interés con respecto a la hipótesis.

Como parte de la implementación de soluciones, el Marco Instrumental presenta una serie de acciones que coadyuvarán a la difusión del tema, y en la medida de lo posible a disminuir la problemática relacionada con los requerimientos.

Por último, de gran valor y como resultado de este trabajo, se presentan una serie de conclusiones, las cuales se agrupan por capítulo, además de las generales de toda la investigación.

Esperando resulte de interés el presente trabajo para los lectores, se invita a la lectura del mismo.

ÍNDICE

1 MARCO PROBLEMÁTICO

1.1 Justificación

Diariamente tenemos contacto con sistemas de información y con software, en las diversas actividades que realizamos y en los servicios a los que tenemos acceso. Como usuarios de esos sistemas deseamos que nos ofrezcan determinada funcionalidad de acuerdo a nuestras necesidades, que sean fáciles de usar, que nos automaticen ciertas tareas, que nos proporcionen información útil, que respondan adecuadamente a nuestras peticiones y que sean seguros, estables, entre otras.

Como parte de un equipo de desarrollo de sistemas –ya sea como líder de proyecto, analista o programador– nos interesa producir sistemas de información útiles, que cubran adecuadamente las expectativas de los usuarios, que les permita resolver ciertos problemas o que les ofrezcan una ventaja competitiva, y que a nosotros nos generen un beneficio económico y profesional.

Sin embargo, es común por un lado, que los usuarios no sean especialistas en cómputo y sistemas, por lo que no expresan en términos informáticos sus necesidades, no saben lo que realmente requieren o peor aún no identifican sus verdaderos problemas. Por otro lado, el equipo de desarrollo esta preparado para ofrecer soluciones adecuadas desde el punto de vista informático, mas no es experto en los procesos de negocio de los usuarios.

Los requerimientos representan el elemento primordial en el desarrollo de un sistema de información; si no existen requerimientos no hay sistema por construir. A pesar de esto, muchas veces no se le da la importancia adecuada a la Ingeniería de Requerimientos (IR), lo que representa un problema común y actual en las organizaciones que desarrollan software. Esto no es nuevo, de hecho, uno de los motivos de la llamada "crisis del software" en los años 60's fue precisamente la insuficiente especificación y falta de control en los requerimientos.

¿Cómo lograr entonces que se identifiquen y especifiquen adecuadamente los requerimientos para desarrollar los sistemas correctos? Es necesario que el equipo responsable del desarrollo comprenda y especifique los requerimientos que debe cubrir dicho sistema, considerando que los clientes y usuarios no siempre son especialistas informáticos y aplicando adecuadamente las técnicas y herramientas de la Ingeniería de Requerimientos que le permitan realizar estas actividades de manera eficaz, ya que una de las funciones del Licenciado en Informática como analista, es precisamente ser el intermediario entre los usuarios de un sistema de información y el equipo técnico de desarrollo.

1.2 Identificación del problema

El conjunto de requerimientos define lo que el sistema va a hacer por lo que si no se identifican de manera correcta, el software no proporciona al usuario la funcionalidad esperada; además si no se determinan de manera completa y clara no se conocerá el alcance ni será posible estimar la dimensión real del proyecto.

Es común que los requerimientos no se expresen de manera clara ni se documenten de manera apropiada; aunque existen diversas técnicas, notaciones y métodos, no son utilizados de forma correcta, son complejas, llegan a ser incomprensibles para los usuarios, no representan un estándar entre los grupos involucrados en el desarrollo y algunas veces no reflejan la realidad.

Es un hecho además que los requerimientos cambian, no se actualiza la documentación relacionada y los cambios no son comunicados a todos los grupos involucrados en el desarrollo. Los cambios en requerimientos impactan de manera importante la planeación y arquitectura del proyecto.

De acuerdo con la naturaleza de cada proyecto, existen distintos tipos de requerimientos que a su vez tienen niveles de detalle. La cantidad de requerimientos de un proyecto de software puede ser muy grande y difícil de controlar.

Además como se mencionó anteriormente, se requiere una alta participación del usuario en el proceso de identificar y validar los requerimientos, para garantizar que sean los correctos y cubran sus necesidades, participación que muchas veces no se tiene.

1.3 Demarcación del fenómeno

La problemática expuesta en el punto anterior tiene lugar en las organizaciones o áreas relacionadas con el desarrollo de software, por ejemplo consultoras en informática, fábricas de software, centros de cómputo y áreas internas de las organizaciones que realizan proyectos de desarrollo de sistemas informáticos, y que no cuentan con procesos definidos de ingeniería de requerimientos.

1.4 Conocimiento y opiniones en el medio

Con el fin de recopilar las opiniones de otras personas con conocimientos prácticos o teóricos sobre el tema, se les aplicó un cuestionario. A continuación se detalla este herramienta.

1.4.1 Objetivo del Cuestionario

Realizar un sondeo de la situación actual de la ingeniería de requerimientos en las áreas relacionadas con el desarrollo de sistemas.

1.4.2 Contenido del Cuestionario

1. ¿Cuál considera que sea el factor más importante para el éxito en el desarrollo de un sistema de información? **Objetivo:** Identificar el principal factor de éxito en el software de un sistema informático. **Respuesta esperada:** Que el sistema haga lo que el usuario necesita.
2. ¿Conoce algún proyecto de desarrollo de sistemas que haya fracasado por una mala o insuficiente definición de requerimientos? **Objetivo:** Comprobar si la inadecuada definición de requerimientos es una causa real en el fracaso de un proyecto informático. **Respuesta esperada:** Sí, porque: no se acordaron todos los requerimientos por lo que no se limitó el alcance real del proyecto y porque sobrepasó el tiempo y los recursos estimados.
3. En su experiencia en el desarrollo de sistemas, ¿se documentan los requerimientos de alguna manera? **Objetivo:** Si los requerimientos son documentados de alguna manera, conocer las técnicas utilizadas para la especificación de éstos. **Respuesta esperada:** No se documentan, o sólo como texto libre.

4. ¿El cliente y/o usuario participa en la definición y validación de requerimientos?

Objetivo: Reconocer la interacción con el usuario en la fase de definición de requerimientos
Respuesta esperada: Sí.

5. ¿Qué técnicas se utilizan en la obtención de requerimientos?

Objetivo: Conocer las técnicas más utilizadas para la obtención de requerimientos.
Respuesta esperada: Entrevistas con el usuario, cuestionarios, llenado de formatos específicos para recabar requerimientos, pláticas informales, documentos recopilados y experiencia de sistemas anteriores.

6. Cuando un requerimiento cambia, ¿se actualiza la documentación para comunicarlos a los grupos involucrados en el desarrollo de sistemas?

Objetivo: Conocer si se documenta y controlan los cambios de requerimientos.
Respuesta esperada: No se documentan los cambios.

7. ¿Los requerimientos son comprendidos claramente y sin ambigüedad por los usuarios y por el grupo de desarrollo?

Objetivo: Confirmar si la forma en que se especifican requerimientos es clara, precisa y comprensible por todos los involucrados.
Respuesta esperada: Sí, sólo por el equipo de desarrollo.

8. ¿Qué porcentaje de tiempo promedio de un proyecto se destina a las fases de análisis y diseño conceptual (antes de codificar) en un proyecto de desarrollo de software?

Objetivo: Estimar el tiempo promedio invertido en las fases previas a la codificación.
Respuesta esperada: Menos del 20%.

9. ¿Los requerimientos se usan como base para la planeación, seguimiento y control del proyecto? **Objetivo:** Conocer la importancia de los requerimientos en la planeación del proyecto. **Respuesta esperada**: Sí.
10. ¿Los requerimientos son ponderados y clasificados de acuerdo a su importancia, características y complejidad? **Objetivo:** Reconocer si se identifican y analizan los distintos tipos de requerimientos. **Respuesta esperada**: No.
11. ¿Cuáles son las dos causas más frecuentes por las que un proyecto se sobrepasa del tiempo estimado? **Objetivo:** Conocer si los requerimientos son determinantes para terminar un proyecto en el tiempo estimado. **Respuesta esperada**: Porque surgen nuevos requerimientos.
12. ¿Considera que el no realizar adecuadamente las actividades relacionadas con la obtención y especificación de requerimientos constituye un problema en los proyectos de desarrollo de software? **Objetivo:** Comprobar si las actividades de la ingeniería de requerimientos influyen en el éxito o fracaso de los proyectos de software. **Respuesta esperada**: Sí.

1.4.3 Conocimiento empírico: personas entrevistadas

Los seleccionados fueron básicamente egresados de la Licenciatura de Informática, Ingeniería de Computación y Ciencias de la Computación, así como personas que trabajan en diversas organizaciones en las áreas de cómputo y desarrollo de sistemas:

Nombre	Puesto	Organización
Adrián Galindo Hernández	Ingeniero de Software	MPSNet
Alejandro Vergara Torres	Arquitecto de Software	Instituto Federal Electoral UNICOM
Alma Rosa García Martínez	Líder de Proyecto	Dirección de Sistemas, UNAM
Bryan Andrik De La Torre Cerón	Jefe del Centro de Cómputo	Instituto de Investigaciones Filosóficas, UNAM
Carina Guadalupe Sarabia Delgado	Líder de Proyecto	Dirección de Sistemas, UNAM
Francisco Israel Gerardo Herros	Gerente de Proyectos	Milestone
Grigori Sidorov	Investigador	Centro de Investigación en Computación, IPN
Gustavo A. Cortez Aguilar	Analista-Programador	Dirección de Sistemas, UNAM
Héctor Valdivia Rosas	Líder de proyecto	Dual Hi-Tech
José de Jesús Hernández Suárez	Líder de Proyecto	Dirección de Sistemas, UNA.
José Ventura Miranda	Gerente de Sistemas	Seguros Comercial América
Lilia Alias Ruiz	Analista-Programador	Dirección de Sistemas, UNAM
Nancy Hernández R.	Analista Sr.	IDS
Nora Elizabeth Tapia Ruiz	Jefa del Departamento de Control de Calidad y Auditoria Informática	Dirección de Sistemas, UNAM
Norman Miguel Maza Magnussen	Líder de Proyecto	Comisión Federal de Electricidad
Ricardo Aguillón Romero	Líder de Proyecto	Comisión Federal de Electricidad
Ricardo Rojas Castellanos	Líder de Proyecto	PEMEX Exploración y Producción
Rogelio Ventura Miranda	Gerente	PEMEX Exploración y Producción
Sara De Jesús Sánchez	Analista Sr.	Ingeniería Aplicada
Víctor Hugo de la Rosa Solís	Gerente de Proyecto	VIsion Consulting

1.4.4 Opiniones profesionales: personas entrevistadas

Los seleccionados son personas especialistas en el desarrollo de sistemas, expertos en ingeniería de software y profesionistas con amplia experiencia, cuyo medio de contacto principalmente fue la Lista de Calidad de Software de la AMCIS.

Nombre	Puesto	Organización
Adolfo Guzmán Arenas	Investigador	Centro de Investigación en Computación, IPN.
Alejandro Pérez Hernández	Gerente de Sistemas	Qualitas Compañía de Seguros, S.A. de C.V.
Carlos Marco Macías Medina	Gerente de Proyectos	Milestone Consulting, S.C.
Fernando Gamboa	Investigador	Centro de Instrumentos, UNAM
Josefina Rodríguez Jacobo	Investigadora	Centro de Investigación Científica y de Educación Superior de Ensenada, IPN.
Marco Dorantes Martínez	Gerente de Proyecto	Microsoft Consulting
Mónica Alegría Vázquez	Consultora en Administración de Requerimientos	ITERA
Sandra Minora Orantes Jiménez	Administrador de Proyectos	Centro de Investigación en Computación, IPN.

1.4.5 Resultados del Cuestionario

Las respuestas del cuestionario fueron computadas para obtener los porcentajes mostrados a continuación, algunos los cuales no suman el 100% debido a que era posible seleccionar más de una respuesta.

	¿Cuál considera que sea el factor más importante para el éxito en el desarrollo de un sistema de información?	
1	Que el sistema haga lo que el usuario necesita	90%
	Que no tenga errores ni fallas	28%
	Que se termine a tiempo y bajo el presupuesto estimado	27%
	Que sea amigable al usuario	27%

	Conclusión: Si un sistema no proporciona la funcionalidad requerida, no es una solución útil para el usuario. Si bien es importante que un sistema de información no tenga errores ni fallas y que se termine a tiempo, lo principal es que éste realice lo que el usuario necesita para lograr sus objetivos.

2	**¿Conoce algún proyecto de desarrollo de sistemas que haya fracasado por una mala o insuficiente definición de requerimientos?**	
	Sí	82%
	No	18%
	Sí, porque : No se acordaron todos los requerimientos, por lo que no se limitó el alcance del proyecto	73%
	Sí, porque : Sobrepasó el tiempo y los recursos estimados	46%
	Sí, porque : La funcionalidad del sistema no correspondía con lo que el usuario pidió	36%
	Sí, porque : El grupo de desarrollo no sabía lo que el sistema debía hacer	36%
	Conclusión : Existe un alto porcentaje de proyectos de desarrollo de sistemas que han fracasado por no definir de manera adecuada y suficiente los requerimientos; esto repercute principalmente en tener un proyecto sin límites precisos que no se finaliza con los recursos estimados.	

3	**En su experiencia en el desarrollo de sistemas, ¿se documentan los requerimientos de alguna manera?**	
	Sí	82%
	No	18%
	Sí, con: Diagramas de Flujo de Datos DFD (Yourdon)	64%
	Sí, con: Texto libre	55%
	Sí, con: Casos de Uso (UML)	36%
	Conclusión: En la mayoría de los proyectos de desarrollo de software los requerimientos son documentados de alguna manera, pero no se usa una notación estándar o por lo menos no se observó una notación principalmente utilizada.	

4	**¿El cliente y/o usuario participa en la definición y validación de requerimientos?** Sí 64% No 36% **Conclusión:** Sólo en un 64% de las respuestas obtenidas se confirma que el cliente participa en la definición y validación de requerimientos, aspecto que es muy importante para lograr que la funcionalidad de un sistema sea adecuada a sus necesidades, garantizar una mayor calidad del software y minimizar así algunos riesgos que se pueden presentar.

5	**¿Qué técnicas se utilizan en la obtención de requerimientos?** Sí 64% No 36% Entrevistas con el usuario y/o cliente 100% Pláticas informales 73% Cuestionarios 45% Llenado de formatos específicos para recabar requerimientos 45% Documentos recopilados 45% **Conclusión:** Las entrevistas con los usuarios y/o clientes son la fuente principal para obtener los requerimientos de un sistema. Representan una técnica de recopilación de información sencilla que utilizan todos los encuestados, no obstante algunas veces es difícil contar con la disposición y tiempo de los clientes/usuarios. Existen otras fuentes de información como las pláticas informales, los cuestionarios y los documentos recopilados.

6	**Cuando un requerimiento cambia, ¿se actualiza la documentación para comunicarlos a los grupos involucrados en el desarrollo de sistemas?** Si se documentan los cambios 73% No se documentan los cambios 36% No cambian los requerimientos 0%

	Conclusión: El cambio en los requerimientos de un sistema es ineludible. El 73% de los encuestados afirman que sí documentan de alguna manera los cambios en los requerimientos para que sean comunicados a los grupos involucrados en el desarrollo.

7	**¿Los requerimientos son comprendidos claramente y sin ambigüedad por los usuarios y por el grupo de desarrollo?** A veces 67% Sí, por ambos 33% **Conclusión:** Solo un 33% de las veces, los requerimientos son comprendidos de manera clara y precisa tanto por los usuarios como por el grupo de desarrollo. Si el cliente no comprende los requerimientos del sistema, no puede validarlos adecuadamente; si el equipo de desarrollo no los comprende, el sistema no tendrá la funcionalidad apropiada.

8	**¿Qué porcentaje de tiempo promedio de un proyecto se destina a las fases de análisis y diseño conceptual (antes de codificar) en un proyecto de sistemas?** Menos del 20% 55% Del 20% al 40% 45% Más del 40% 18% **Conclusión:** El tiempo que se dedica a las fases de análisis con respecto al desarrollo es muy poco, considerando que estas etapas son trascendentales para garantizar que se cumplan plenamente los requerimientos, cubrir las necesidades y expectativas de los usuarios y minimizar los errores y fallas.

9	**¿Los requerimientos se usan como base para la planeación, seguimiento y control del proyecto?** SI 91% NO 9%

	Conclusión: En su mayoría los requerimientos son un elemento importante para la planeación, seguimiento y control de los proyectos de software.

10	**¿Los requerimientos son ponderados y clasificados de acuerdo a su importancia, características y complejidad?** SI 73% NO 27% **Conclusión:** En un buen número de proyectos, los requerimientos son ponderados y clasificados de acuerdo a su importancia, no obstante no existen estándares o lineamientos claros para ello.

11	**¿Cuáles son las dos causas más frecuentes por las que un proyecto se sobrepasa del tiempo estimado?** Surgen nuevos requerimientos 100% El usuario y/o cliente no proporciona la información requerida 64% Falta de personal para el desarrollo 38% Falta de capacitación en el personal 36% Falta de recursos/infraestructura para desarrollar 18% **Conclusión:** Siempre surgen nuevos requerimientos en un proyecto de desarrollo y es la causa principal por lo que no se cumplen los tiempos estimados inicialmente. No obstante otra causa muy importante es que el cliente no proporciona la información requerida a tiempo.

12	**¿Considera que el no realizar adecuadamente las actividades relacionadas con la obtención y especificación de requerimientos constituye un problema en los proyectos de desarrollo de software?** SI 100% NO 0%

	Conclusión: La problemática existe y todos los encuestados están de acuerdo. Definitivamente los requerimientos son fundamentales para llevar a cabo un proyecto de desarrollo de software y para que el producto resultante satisfaga las necesidades y cumpla las expectativas del usuario.

1.5 Hipótesis preliminar

Las relaciones hipotéticas preliminares se presentan a continuación:

CAUSA (S)	EFECTO (S)
▪ Los requerimientos se identifican de manera incorrecta	▪ Los sistemas no realizan lo que los usuarios necesitan o no tienen las características deseadas.
▪ Los requerimientos no se documentan	▪ No existe un acuerdo formal entre el cliente y el equipo de desarrollo de lo que el sistema debe realizar. ▪ Los grupos relacionados en el desarrollo no tienen una referencia formal para consultar y validar las características que debe tener el sistema.
▪ No se especifican todos los requerimientos de un sistema ▪ No se controlan adecuadamente los cambios en requerimientos	▪ No se realiza una planeación real y adecuada de los proyectos. ▪ Los proyectos de desarrollo de software no se terminan a tiempo.
▪ Los requerimientos no son expresados de una manera clara y estándar	▪ Los requerimientos no son comprendidos por todos lo grupos involucrados en el desarrollo ni por grupos externos.
▪ Identificar adecuadamente los requerimientos de un sistema	▪ El sistema hace lo que el usuario necesita. ▪ Se estima adecuadamente la dimensión de un proyecto. ▪ Los involucrados conocen exactamente las características del sistema a desarrollar.

▪ Especificar de manera clara y estándar los requerimientos de un sistema	▪ Existe una referencia formal de lo que el sistema debe hacer. ▪ El equipo de desarrollo sabe qué debe hacer. ▪ Permite comunicar adecuadamente los cambios en requerimientos. ▪ El usuario puede validar los documentos y modelos de requerimientos.
▪ Contar con las técnicas y herramientas adecuadas para realizar la Ing. De Requerimientos en el desarrollo de sistemas	▪ Permite obtener, analizar, especificar y validar los requerimientos de una manera sencilla y eficaz.
▪ La realización de la Ingeniería de Requerimientos mediante el uso de diversas técnicas y herramientas, en el desarrollo de un sistema permite	▪ Mayor control del proyecto. ▪ Mejor estimación de la dimensión del proyecto. ▪ Mayor calidad del software. ▪ Mejor comunicación con clientes y usuarios. ▪ Desarrollar sistemas que realicen la funcionalidad requerida por el usuario. ▪ Mejorar la comunicación entre los equipos involucrados en el desarrollo. ▪ Alcanzar un mayor nivel de competitividad en el desarrollo de software. ▪ Cubrir los principales aspectos de los modelos de madurez de procesos.
▪ La realización de las actividades de la ingeniería de requerimientos, mediante el uso de técnicas y herramientas en un proyecto de desarrollo de software	▪ Permite especificar de manera clara, precisa, y completa los requerimientos de un sistema informático. ▪ Permite desarrollar sistemas con la funcionalidad y características requeridas por el usuario. ▪ Coadyuva a terminar los proyectos dentro de los recursos estimados. ▪ Constituye un elemento para lograr una mayor calidad del software. ▪ Permite cumplir con los lineamiento de los modelos de madurez de procesos.

1.6 Objetivos

1.6.1 Personales

- Graduarse de acuerdo al Reglamento General de Exámenes Profesionales de la Facultad de Contaduría y Administración. (Artículos 3,7, 44-56).

- Especializarse en el área de Ingeniería de Requerimientos para participar en la mejora de procesos y el desarrollo de software en las organizaciones mexicanas.

1.6.2 Generales

- Proponer los elementos para realizar una adecuada Ingeniería de Requerimientos, integrando las técnicas y herramientas existentes que permitan mejorar el desarrollo de software en las organizaciones y áreas de sistemas.

- Difundir la Ingeniería de Requerimientos como un conjunto de actividades importantes en el desarrollo de software que permita lograr una mayor calidad y competitividad en las organizaciones mexicanas.

2 MARCO TEÓRICO

2.1 Acopio de Libros

2.1.1 Libros de Estudio

Requirements Engineering ***(Ingeniería de Requerimientos)***
Ian Sommerville. Peter Sawyer.
Editorial *John Wiley & Sons*. 1ª Edición, 1997.
Biblioteca de la DGSCA, UNAM. Clasificación: QA76.6 S6453. ISBN 0-471-974444-7.

Capítulo	Contenido
1. Introducción	Conceptos relacionados con la Ingeniería de Requerimientos como: requerimiento, Ingeniería de Requerimientos, documento de requerimientos, requerimientos funcionales y no funcionales, proceso de Ingeniería de Requerimientos y usuarios que intervienen en los requerimientos. Aspectos generales sobre mejoramiento del proceso de ingeniería de requerimientos, ISO 9000.
2. Mejoramiento del proceso práctico	Generalidades sobre madurez, evaluación y mejoramiento del proceso de Ingeniería de Requerimientos. Propone una serie de lineamientos importantes a considerar en un proceso de mejoramiento de Ingeniería de Requerimientos.
3. El documento de requerimientos	El documento de requerimientos que es usado para comunicar los requerimientos del sistema a los clientes y usuarios, así como a los administradores y desarrolladores del sistema. En el este capítulo se sugiere cómo mejorar la efectividad de este documento, mediante una estructura y contenido estándar. Señala los beneficios de este documento, así como los costos y problemas de introducirlo en la organización.
4. Identificación de requerimientos	La identificación de requerimientos es el proceso de descubrir los requerimientos para un sistema como resultado de la comunicación con los clientes, usuarios y otras personas relacionadas con el sistema. Se requieren conocimientos tanto del dominio del problema como de la organización. En el capítulo se apuntan una serie de lineamientos a considerar en este proceso como: evaluar la viabilidad del

	sistema, tomar en cuenta consideraciones organizacionales y políticas, identificar y consultar a los involucrados, registrar fuentes de requerimientos, definir el ambiente operativo del sistema, identificar restricciones, tomar en cuenta múltiples puntos de vista, usar escenarios, desarrollar prototipos, definir procesos operacionales y reutilizar requerimientos. De cada uno de estos lineamientos se presentan beneficios, implementación, así como su costo y dificultad.
5. Análisis y negociación de requerimientos	Una vez que se ha descubierto un conjunto inicial de requerimientos, deben ser analizados para identificar conflictos, duplicidad, omisiones e inconsistencias. A partir de esto se acuerda con los involucrados un conjunto de requerimientos. Se deben resolver los conflictos y los requerimientos deben ser ponderados. Los lineamientos sugeridos son: definir límites del sistema, usar listas de verificación, planear la resolución de conflictos, priorizar los requerimientos, clasificar los requerimientos de manera multidimensional, usar matrices de requerimientos, evaluar el riesgo de los requerimientos. De cada uno de estos lineamientos se presentan beneficios, su implementación, así como su costo y dificultad.
6. Descripción de requerimientos	La descripción de cada requerimiento debe ser concisa, comprensible y sin ambigüedad. Las sugerencias que se proponen en este capítulo son: definir plantillas estándar para describir requerimientos; usar un lenguaje simple, consistente y conciso; usar diagramas apropiadamente; complementar con lenguaje natural otras formas de describir requerimientos, especificar requerimientos de manera cuantitativa. De cada uno de estos lineamientos se presentan beneficios, su implementación, así como su costo y dificultad.
7. Modelado de sistemas	Se presentan aspectos relacionados con el desarrollo de modelos abstractos del sistema los cuales pueden formar parte de una especificación detallada del sistema, por ejemplo: un modelo del ambiente del sistema y un modelo de la arquitectura del sistema. Las recomendaciones de este punto son: desarrollar modelos complementarios del sistema, modelar el ambiente del sistema, modelar la

	arquitectura del sistema, usar métodos estructurados para el modelado del sistema, usar un diccionario de datos, documentar las relaciones entre los requerimientos de los involucrados y los modelos complementarios del sistema.
8. Validación de requerimientos	Después que se ha producido el documento de requerimientos, los requerimientos deben ser validados formalmente. Este proceso de validación consiste en asegurar que los requerimientos sigan estándares de calidad. Los lineamientos para la validación de requerimientos son: verificar que el documento de requerimientos cumpla con el estándar, organizar inspecciones formales de los requerimientos, usar equipos multidisciplinarios para revisar requerimientos, definir listas de aspectos por validar, usar prototipos para representar requerimientos, escribir un borrador del manual de usuario y realizar casos de prueba para los requerimientos.
9. Administración de requerimientos	La administración de requerimientos involucra los procesos relacionados con los cambios de los requerimientos del sistema. En el capítulo se proponen lineamientos afines con la administración de requerimientos: identificar de manera única cada requerimiento, definir políticas para la administración de requerimientos, definir políticas de rastreo, usar bases de datos para administrar requerimientos, definir políticas de administración de cambios, identificar requerimientos volátiles del sistema y registrar requerimientos rechazados.
10. Ingeniería de requerimientos para sistemas críticos	Los sistemas críticos son sistemas los cuales tienen fiabilidad, disponibilidad y seguridad rigurosas. El costo de una falla del sistema es muy alto y la ingeniería de requerimientos y el proceso de desarrollo del sistema deben asegurar la confiabilidad del sistema. Los lineamientos propuestos para este punto son: crear listas de verificación de requerimientos de seguridad, involucrar revisores externos en la validación del proceso, identificar y analizar riesgos, verificar requerimientos funcionales y operacionales contra requerimientos de seguridad, utilizar especificaciones formales, investigar y aprender de experiencias en incidentes ocurridos, establecer una

	cultura de seguridad organizacional, entre otros.
11. Modelado de sistemas con métodos estructurados	En el capítulo se describe el uso de los métodos estructurados más comunes para el modelado de sistemas. Algunos de los métodos tratados son: SADT, Ward-Mellor, SSADM, Booch, OMT, OOA.
12. Especificación formal	En el capítulo se discuten los pros y contras de las técnicas formales de especificación que generalmente se utilizan en los sistemas críticos.

Software Requirements: objects, functions, and states.
(***Requerimientos de Software: objetos, funciones y estados)***
Alan M. Davis.
Editorial *Prentice Hall*. 1ª Edición, 1993.
Biblioteca del IIMAS, UNAM. Clasificación: QA76 .76D47 D393. ISBN 0-13-805763-X.

Capítulo	Contenido
1. Introducción	Anteriormente el costo de los sistemas de información se centraba en el hardware, esta situación se ha ido revirtiendo; ahora el mayor costo se relaciona con el desarrollo de software. Existen dos aspectos a considerar durante la fase de requerimientos: análisis del problema y descripción del producto. La Ingeniería de Software es la aplicación de principios científicos en el desarrollo y mantenimiento de software. Se presenta un diagrama del ciclo de vida de la Ingeniería de Software, el cual incluye las siguientes etapas: a)Requerimientos de software, b)Diseño preliminar, c)Diseño detallado, d)Codificación, e)Pruebas unitarias, f)Pruebas de integración, g)Pruebas del sistema, h)Liberación, producción y distribución, i)Mantenimiento y crecimiento, j)Planeación de pruebas del sistema de software, k)Planeación de pruebas de integración, l)Planeación de pruebas unitarias. En esta sección también se definen conceptos como: requerimientos, SRS, objetos, funciones y estados; además se señala y justifica la importancia de los requerimientos en el desarrollo de software.

2. Análisis del problema	El análisis del problema es la actividad que implica conocer el problema a resolver, comprender las necesidades de los usuarios, identificar los usuarios reales y comprender las restricciones de la solución. En este capítulo se describen, comparan, contrastan y aplican diversas técnicas para el análisis del problema como son: • Análisis del Problema Orientado a Objetos - Análisis Orientado a Objetos de Coad (COOA) - Desarrollo de Sistemas Jackson (JSD) • Análisis del Problema Orientado a Funciones - Diagramas de Flujo de Datos (DFD) - Diccionario de Datos - Definición Estructurada de Requerimientos (SRD) - Técnica Estructurada de Análisis y Diseño (SADT) - Especificación Estructurada de Análisis y Sistemas (SASS) - Análisis Estructurado Moderno - Lenguaje/Analizador de Declaración del Problema (PSL/PSA) • Análisis del Problema Orientado a Estados
3. La Especificación de requerimientos de software	El comportamiento externo del producto de software se debe especificar en un documento (especificación de requerimientos de software-ERS). La fase de requerimientos no está completa hasta que dicha especificación ha sido escrita. En el capítulo se mencionan los atributos que debe tener una ERS bien escrita, así como la estructura de un documento de ERS de acuerdo a diversos estándares.
4. Especificando requerimientos de comportamiento	Los requerimientos de comportamiento definen precisamente qué entradas son esperadas por el software, qué salidas serán generadas y los detalles de las relaciones que existen entre las entradas y las salidas. En el capítulo se discuten una variedad de técnicas de especificación de requerimientos de comportamiento y sus modelos fundamentales como son: • Orientadas a Estados - Máquina de Estados Finitos

	- Diagramas de Estado - Sistema de Validación de Ingeniería de Requerimientos (REVS) - Redes de Petri • Orientadas a Funciones - Tablas de decisión y árboles de decisión - Lenguaje de Diseño de Programas (PDL) - Procesador de Lenguaje de Requerimientos (RLP) - Lenguaje de Especificación y Descripción (SDL) - Lenguaje de Especificación Orientado a Procesos, Aplicativo e Interpretable (PAISLey) • Orientadas a Objetos - Diagramas de Estado - Desarrollo de Sistemas Jackson (JSD)
5. Especificando requerimientos de no comportamiento	Todas las aplicaciones tienen requerimientos adicionales que definen cualidades o atributos que debe presentar el sistema de software. En esta sección se definen cada uno de los atributos de calidad del software propuestos por Barry Bohem como son: portabilidad, fiabilidad, eficiencia, ingeniería humana y facilidad de mantenimiento.
6. Prototipado de requerimientos	El uso de prototipos es común en la ingeniería, consiste en una implementación parcial de un sistema, que permita a los clientes, usuarios y desarrolladores comprender el problema y visualizar la solución. Se presentan dos enfoques: prototipos desechables y evolutivos.
7. Otras consideraciones finales	En este capítulo se describen algunas técnicas adicionales que coadyuvan en la especificación de requerimientos como son: procesadores de texto, bases de datos y herramientas para diagramar.

Managing Software Requirements: A Unified Approach
(Administrando Requerimientos de Software: Un Enfoque Unificado)
Dean Leffingwell, Don Widrig, Edward Yourdon.
Editorial *Addison-Wesley*. 1a. Edición. Noviembre, 1999.
Biblioteca de la Fac. de Ciencias, UNAM. Clasif. QA76.76D47 L44. ISBN 0201615932.

Introducción	
Capítulo	**Contenido**
1. El problema de los requerimientos	En el capítulo resaltan los siguientes puntos: • El objetivo es desarrollar software con calidad -dentro del tiempo y del presupuesto- que satisfaga las necesidades reales de los usuarios • El éxito de los proyectos depende de una buena administración de requerimientos • Los errores en requerimientos son los más comunes en el desarrollo de software y los más costosos de corregir Se presentan además una serie de estadísticas del STANDISH GROUP y del ESPITI (European Software Process Improvement Training Iniciative).
2. Introducción a la administración de requerimientos	En este capítulo se presentan los conceptos de: requerimiento, administración de requerimientos, característica y capacidad. Se mencionan tres tipos de aplicaciones de software: 1) Sistemas de información y otras aplicaciones desarrolladas para usarse dentro de una organización 2) Software comercial 3) Software inmerso
3. El equipo del software	La administración de requerimientos involucra a cada miembro del equipo de diferente manera. Las actividades recomendadas para lograr una efectiva administración de requerimientos son: 1) Analizar el problema 2) Comprender las necesidades del usuario 3) Definir el sistema 4) Definir el alcance 5) Refinar la definición del sistema 6) Construir el sistema correcto

Analizando el Problema	
Capítulo	**Contenido**
4. Los cinco pasos en el análisis del problema	El análisis del problema es el proceso de comprender los problemas del mundo real y las necesidades del usuario, y proponer soluciones que cubran estas necesidades. Además ganar una mejor comprensión del contexto e identificar los actores es clave antes de empezar el desarrollo. Los cinco pasos del análisis del problema son: 1) Ganar acuerdo sobre la definición del problema 2) Comprender las causas raíces: el problema detrás del problema 3) Identificar a los involucrados y a los usuarios 4) Definir los límites entre el sistema-solución y el mundo real 5) Identificar las restricciones de la solución.
5. Modelado del negocio	El modelado de negocio se usa en los sistemas de aplicación específica. Su propósito es: • Comprender la estructura y dinámica de la organización y • Asegurar que los clientes, usuarios finales y desarrolladores tengan una comprensión general de la organización.
6. Ingeniería de sistemas de software (sistemas inmersos)	Los conceptos de este capítulo aplican en el desarrollo de sistemas inmersos. La ingeniería de sistemas es un enfoque de análisis de problemas para sistemas inmersos. El software, no el hardware, determinará la última funcionalidad del sistema y el éxito del sistema y consume la mayoría de los costos de investigación y del desarrollo del sistema. Ocho principios de ingeniería de sistemas propuestos por el INCOSE (Internacional Council on System Engineering) son: 1) Conocer el problema, el cliente y el consumidor 2) Usar criterios efectivos basados en necesidades para tomar decisiones 3) Establecer y manejar requerimientos 4) Identificar y evaluar alternativas

	5) Verificar y validar requerimientos y el desempeño de la solución 6) Mantener la integridad del sistema 7) Usar un proceso articulado y documentado 8) Administrar el plan.
Comprendiendo las Necesidades de los Usuarios	
Capítulo	**Contenido**
7. El Reto de la obtención de requerimientos	En este capítulo se discuten algunos "síndromes" comunes en la obtención de requerimientos: *El síndrome "si, pero..." y el síndrome del "usuario y el desarrollador". Ambos vienen de diferentes ámbitos, hablan lenguajes diferentes y tienen diferentes objetivos.* Asimismo se enumeran las técnicas para la obtención de requerimientos: • Entrevistas y cuestionarios • Talleres de requerimientos • Lluvia de ideas y reducción de ideas • Storyboards • Casos de Uso • Juego de roles • Prototipado
8. Las características de un producto o sistema	Una necesidad de los interesados es una reflexión del negocio, o personal, un problema operacional u oportunidad que justifica la compra o uso de un nuevo sistema. Una característica es un servicio que el sistema proporciona para satisfacer una o más necesidades de los involucrados. Las características son fácilmente expresables en lenguaje natural y consisten en una frase corta. En el capítulo se mencionan algunos atributos de las características.
9. Entrevistas	En este capítulo se detalla el objetivo, características y técnicas para realizar entrevistas efectivas. Se presenta además una guía para realizar la entrevista. El cuestionario no es un sustituto de la entrevista. Pueden ser utilizados para validar suposiciones y obtener datos

	estadísticos.
10. Talleres de requerimientos	En este capítulo se detalla la técnica de las mesas de trabajo o talleres para la obtención de requerimientos. Los talleres de requerimientos permiten obtener consenso sobre los requerimientos de la aplicación y ganar acuerdos rápidos. Los involucrados clave del proyecto son reunidos por períodos cortos pero intensivos. Permiten resolver aspectos políticos que interfieren en el éxito del proyecto. La salida es una definición preliminar el sistema. La lluvia de ideas es la parte más importante de los talleres.
11. Lluvia de Ideas y reducción de ideas	En este capítulo se detalla la técnica de "lluvia de ideas". Sus fases son: generación de ideas y reducción de ideas (analizar las ideas generadas).
12. Escenarios	En este capítulo se presenta la técnica de los *storyboards* o escenarios que consisten en una representación un tanto informal de las características de la solución.
13. Aplicando casos de uso	Los casos de uso como el *storyboard* identifican el quién, qué y cómo del comportamiento del sistema. Un caso de uso describe la secuencia de acciones que un sistema realiza para proporcionar un resultado de valor a un actor (usuario o sistema). Los casos de uso proporcionan una notación formal para capturar la funcionalidad deseada.
14. Juego de roles	El juego de roles permite que el equipo de desarrollo experimente en el mundo de usuario directamente. En este capítulo se describe esta técnica aplicada a la identificación y obtención de requerimientos. Otras técnicas complementarias son: el diagrama de pescado, los ensayos y las tarjetas CRC.
15. Prototipos	Los prototipos de requerimientos de software permiten realizar una implementación parcial del sistema. En este capítulo se presentan sus ventajas así como los distintos tipos de prototipos más comunes.
	Definiendo del Sistema
Capítulo	**Contenido**
16. Organización de la información de requerimientos	Los requerimientos deben ser documentados. Los documentos que definen el producto a construir, típicamente son llamados especificaciones. Pero los

	requerimientos, rara vez son definidos en un documento monolítico. Se necesitan muchos documentos con distintas perspectivas. En este capítulo se examinan diversos tipos de documentos de requerimientos.
17. El documento de visión	El documento de visión describe la aplicación en términos generales. En el capítulo se detallan los componentes de este documento.
18. El campeón	En esté capítulo se describe la importancia y el rol del líder de proyecto. Dependiendo del tipo y dimensión del proyecto, también es conocido como administrador de producto, administrador del proyecto o administrador de tecnologías de información.
Controlando el Alcance	
Capítulo	**Contenido**
19. El problema del alcance del proyecto	En el capítulo se consideran tres factores del alcance del proyecto: la funcionalidad, los recursos y el tiempo disponible. Diversos problemas en los proyectos de desarrollo de software son consecuencia de no definir un alcance viable y claro.
20. Definiendo el alcance del proyecto	El propósito de definir el alcance es establecer una línea base de requerimientos de alto nivel para el proyecto. En este capítulo se describe cómo obtener una línea base adecuada.
21. Dirigiendo a los clientes	En esta sección, los autores resaltan la importancia de involucrar al cliente en la administración de sus requerimientos y en la definición del alcance de su proyecto. Ofrecen algunos lineamientos de negociación con el cliente.
22. Administración del alcance y modelos de desarrollo de software	Se describen los principales modelos de desarrollo de software como son: el modelo de cascada, el modelo en espiral y el enfoque iterativo.
Refinando la Definición del Sistema	
Capítulo	**Contenido**
23. Requerimientos de software	En este capítulo se presenta una clasificación de los requerimientos de software en: funcionales, no funcionales y restricciones de diseño. Además se mencionan algunas recomendaciones para especificar correctamente los requerimientos de software.

24. Refinando los Casos de Uso	Los Casos de Uso son una técnica apropiada para modelar los requerimientos funcionales del sistema. En el capítulo se describen sus beneficios y cómo construirlos.
25. Una moderna especificación de requerimientos de software	Históricamente la técnica popular para documentar requerimientos fue el lenguaje natural. Hoy en día la especificación está integrada por una estructura lógica más que por un documento físico. En el capítulo se describe la especificación de requerimientos de software moderna.
26. Ambigüedad y especificación	En las especificaciones se debe encontrar un punto intermedio entre ambigüedad y comprensión. El detalle y nivel de especificación depende del contexto de la aplicación. En el capítulo se ofrecen algunas técnicas y recomendaciones para reducir la ambigüedad.
27. Métricas de calidad de requerimientos de software	En este capítulo se describen nueve métricas de calidad de las especificaciones de requerimientos, por ejemplo que sean: correctas, completas, consistentes, verificables y comprensibles. Además se incluye una serie de preguntas para verificar la calidad de los casos de uso.
28. Métodos técnicos para la especificación de requerimientos	Los métodos técnicos para la especificación de requerimientos son útiles cuando la descripción es muy compleja o debe ser muy exacta. En este capítulo de describen brevemente algunos métodos como son: pseudocódigo, máquinas de estado finito, árboles de decisión, diagramas de flujo, modelos entidad-relación, análisis orientado a objetos y análisis estructurado.
Construyendo el Sistema Correcto	
Capítulo	**Contenido**
29. Construyendo correctamente el sistema correcto	La construcción del producto correcto depende en gran medida de la continua revisión de las actividades y los productos durante el desarrollo. En este capítulo se distingue entre lo que es verificación y validación, y se detallan los aspectos relacionados con la verificación.
30. De los requerimientos a la implementación	Cuando hay una pequeña correlación entre los requerimientos, el diseño y la implementación, se habla del problema ortogonal. En este capítulo se presentan algunas técnicas y recomendaciones para pasar de los requerimientos a la implementación adecuadamente, por ejemplo: la orientación a objetos, la realización de casos de

	uso y el modelado arquitectónico bajo la arquitectura 4+1.
31. El rastreo de requerimientos como apoyo a la verificación	El rastreo (traceability) o trazabilidad es una técnica que apoya la actividad de verificación. En este capítulo se describen diversos aspectos del rastreo de requerimientos.
32. Validando el sistema	La validación es el proceso de confirmar que el sistema implementado opere conforme e los requerimientos establecidos. En este capítulo se describe el proceso de validación.
33. Usando el retorno sobre la inversión para determinar el esfuerzo de validación y verificación	El nivel de detalle (profundidad) para verificar un elemento del sistema puede variar. Se describen cinco niveles: examen, inspección, revisiones independientes, pruebas de caja negra y pruebas de caja blanca. La cobertura puede ser total o parcial, esta última está en función del riesgo y del retorno sobre la inversión.
34. Administrando el cambio	Los cambios existen y son inevitables. Se dan por factores internos y externos de los cuales algunas veces no se tiene control. En esta sección se explican cinco recomendaciones para un proceso adecuado de administración del cambio.
35. Empezando	En este último capítulo se presenta un resumen de toda la obra, y se ofrece una guía clara para empezar a aplicar todo lo relacionado con la administración de requerimientos, tratado en este libro.

System and Software Requirements Engineering
(Ingeniería de Requerimientos de Sistemas y Software)
Recopiladores: Thayer, Dorfman
IEEE, 1990.
Biblioteca del IIMAS, UNAM. Clasificación: QA76.6. S97

Capítulo I. Introducción, aspectos generales y terminología

Artículo	Contenido
Ingeniería de requerimientos de sistemas y software *Merlin Dorfman*	La Ingeniería de Requerimientos es una parte de la Ingeniería de Sistemas. Está estrechamente relacionada con el modelo de ciclo de vida o proceso utilizado; en el artículo se tratan los principales modelos. Además se enfatiza la diferencia entre los requerimientos de alto nivel (de sistema) y los requerimientos de bajo

	nivel (de software). Se presentan los fundamentos de la Ingeniería de Requerimientos como son: descomposición y abstracción, asignación, facilidad de rastreo, validación y verificación. Se presentan brevemente distintos enfoques, herramientas y métodos para el desarrollo de requerimientos.
Precisando requerimientos *Laura Scharer*	En al artículo se reconoce que una de las razones más comunes en el fracaso de los sistemas es la mala definición de requerimientos. Se discute la definición de requerimientos desde la perspectiva del analista y del usuario, así como algunas responsabilidades y problemas comunes de ambas partes. Además, se dan ciertos lineamientos para definir un sistema, controlar el sistema, capacitar a los usuarios y tener un enfoque preciso de definición de requerimientos.
Una taxonomía de los aspectos actuales en la Ingeniería de Requerimientos *Gruia-Catalin Roman*	El propósito del artículo es crear conciencia en varios aspectos de la especificación de requerimientos como son: • El papel que representan los requerimientos en todo el ciclo de vida de desarrollo • La diversidad de formas que los requerimientos pueden tomar • Los problemas relacionados. Se presenta la clasificación de requerimientos en: funcionales y no funcionales; una serie de características deseables en la especificación de requerimientos como son: facilidad de adecuación, facilidad de construcción, precisión, que sean completas y consistentes y que se puedan probar; y algunos criterios que pueden ser usados para clasificar las técnicas existentes para la especificación de requerimientos.

Capítulo II. Ingeniería de Sistemas e Ingeniería de Sistemas de Software	
Artículo	**Contenido**
Ingeniería de Sistemas: una introducción *J. Douglas Sailor*	Se presenta terminología básica de la Ingeniería de Sistemas (sistema, ingeniería de sistemas, proceso de ingeniería de sistemas, jerarquía, requerimientos, asignación y verificación); el ciclo de vida de sistemas; el proceso de ingeniería de sistemas; la organización de la ingeniería de sistemas (grupos involucrados); organización de los sistemas; herramientas para los requerimientos de sistema; los pasos para el desarrollo de requerimientos en la ingeniería de sistemas, así como algunos parámetros de medición.
Un enfoque estructurado para la especificación del concepto operacional *Bob Lano*	Presenta la estructura y características del Documento de Concepto Operacional (OCD) cuyo propósito es describir todas las características deseadas del sistema desde un punto de vista operacional; es la base para las fases subsecuentes en el desarrollo. El OCD está dirigido a varios tipos de lectores: usuarios, clientes y desarrolladores. Indica algunas recomendaciones para el desarrollo de las especificaciones. Se explican brevemente algunas herramientas que permiten definir el qué, dónde, por qué, quién, cuándo y cómo.
Ingeniería de Sistemas y asignación de requerimientos *Dale Nelsen*	Se describe cómo se debe realizar el análisis y la asignación de requerimientos, en contraste a la manera como típicamente se hace. Cubre los siguientes aspectos: la definición del problema; la solución; las fuentes de documentación; el proceso de asignación; los métodos de validación de requerimientos; el rastreo de requerimientos; y los aspectos adicionales de la Ingeniería de Sistemas. Presenta además diversos ejemplos con un enfoque estructural.
Ingeniería de Sistemas de Software *Dick Thayer y Win Royce*	En el artículo se reconoce la diferencia entre la Ingeniería de Software y la Ingeniería de Sistemas de Software. La Ingeniería de Sistemas de Software parte de la premisa de que la calidad del producto de software depende de la calidad del proceso usado para crearlo. Se describen todas las fases y actividades del ciclo de vida de

	desarrollo de sistemas de software.
Capítulo III. Análisis y Especificación de Requerimientos de Sistema y de Software	
Artículo	**Contenido**
El análisis y especificación de requerimientos de software y de sistema *Alan M. Davis*	En el documento: • Se señala la diferencia entre requerimientos y diseño • Se resuelve el dilema del "qué" y el "cómo" • Se define y muestra cómo poner en práctica los conceptos de ingeniería: partición, abstracción y proyección • Describe el análisis de las herramientas y métodos estructurados para la definición de requerimientos • Describe cómo escribir una buena especificación de requerimientos.
Especificando requerimientos de calidad de software con métricas *Steven Keller, Laurence Kahn y Roger Panara*	Se definen los diversos términos usados en el área de las métricas de calidad de software, que generalmente son tergiversados y aparentemente contradictorios. Además se describen las métricas directas e indirectas de la calidad del software. Contiene un apéndice con un ejemplo de la metodología para especificar requerimientos de calidad del software, desarrollado por la Fuerza Aérea de los Estados Unidos.
Usando el manual de usuario como la especificación de requerimientos *Norm Howes*	El autor recomienda preparar un manual de usuario como un sustituto de la especificación de requerimientos de software. Presenta un caso de estudio de la NASA. Este enfoque permite una mejor comunicación con el usuario y una mayor comprensión de los requerimientos.
Guía para la especificación de requerimientos de software de la IEEE. *IEEE*	La guía es una copia del estándar 830-1984 de la ANSI/IEEE que describe los aspectos que una especificación de requerimientos debe contener. Describe varias alternativas para dicha especificación y diversas características deseables como son: no ambiguas, completas, verificables, consistentes, modificables y con posibilidad de rastreo.

Capítulo IV. Metodologías y Métodos de Representación de Requerimientos de Software	
Artículo	**Contenido**
Una comparación de los principales enfoques para las especificaciones del software *Pamela Zave*	Señala las diferencias entre las especificaciones en lenguaje natural, las especificaciones operacionales y las especificaciones matemáticas. También presenta una breve descripción de varias técnicas de especificación y define las categorías en las cuales ellas encajan.
Una comparación de técnicas para la especificación del comportamiento externo del sistema *Alan M. Davis*	El autor opina que el lenguaje natural por sí solo es inadecuado para la representación de especificaciones en sistemas grandes. Para minimizar la ambigüedad, la inconsistencia y la insuficiencia, se deben usar otras técnicas complementarias. Se presenta la aplicación de algunos métodos de distinto nivel de formalidad a algunos problemas elementales.
Análisis estructurado: un tutorial *Cyril Svoboda*	Una guía del análisis estructurado (AE) que incluye aspectos para sistemas en tiempo real. Ilustra mediante dibujos el análisis estructurado convencional, señala algunas de las variaciones en la notación (SADT, Gane and Sarson, Hatley y Ward-Mellor), menciona algunas herramientas automatizadas para implementar el AE y discute sobre el futuro de este enfoque.
Enfoque entidad-relación para el modelado de datos *Peter Chen*	Los modelos entidad-relación son ampliamente usados en el desarrollo de sistemas en los cuales las interacciones de los datos representan una característica principal. Algunas veces estos modelos son considerados parte del análisis estructurado. El autor del artículo proporciona un método para modelar los requerimientos desde el punto de vista de los datos, mediante el enfoque entidad-relación.
La matriz N2 *Robert Lano*	La matriz N2 es una herramienta para la tabulación, definición, análisis y descripción de interacciones funcionales y sus interfaces. Esta herramienta es utilizada en el Departamento de Defensa de los Estados Unidos.

	En el artículo se describe el formato de la matriz y se presentan ejemplos.
Introducción al análisis orientado a objetos *Peter Coad y Edward Yourdon*	Los autores describen lo que en ese momento era un nuevo enfoque, orientado a objetos, para el análisis de requerimientos. Reconocen que algunos aspectos del análisis estructurado no son adecuados en todos los sistemas. Proponen un enfoque que combina las ventajas de la descomposición funcional y el modelado de flujo de datos con los objetos y clases.
Una comparación de los métodos de desarrollo estructurados y orientados a objetos *Patrick Loy*	El autor resume la historia y características de los métodos orientados a objetos, considerando y rechazando la idea de que la orientación a objetos es un enfoque más natural con respecto al enfoque funcional. Presenta además un plan para evaluar la transición hacia el desarrollo orientado a objetos.
CORE. Un método para especificaciones de requerimientos controladas *G.P. Mullery*	La Expresión Controlada de Requerimientos es un método estructurado para la especificación de requerimientos que es aplicable para la descripción de los aspectos dinámicos de muchos sistemas, no sólo informáticos. El método CORE es apoyado por una notación y diagramas cuyas características se derivan de varias notaciones ampliamente usadas como son el análisis estructurado y la notación Jackson.
SSADM. Metodología estructurada de análisis y diseño *J. Williams*	La Metodología Estructurada de Análisis y Diseño es una técnica muy utilizada en el Reino Unido y que fue originalmente desarrollada por el gobierno de ese país. Es muy parecida al Análisis Estructurado pero tiene algunas extensiones.
El enfoque operacional vs el enfoque convencional para el desarrollo de software *Pamela Zave*	PAISLey (Lenguaje de Especificación Orientado a Procesos, Aplicativo e Interpretable) es uno de los llamados lenguajes de especificación ejecutables. La especificación ejecutable es una herramienta de prototipado. El desarrollador especifica un sistema en el lenguaje PAISLey de acuerdo con su comprensión de las necesidades del usuario, y ejecuta la especificación para observar el comportamiento del sistema. El usuario y el

	desarrollador van corrigiendo la especificación hasta que puede ser usada como base para el desarrollo; no es el sistema final. En el artículo se discute esta metodología, la cual no es muy utilizada en la industria.
El lenguaje de especificación descartes *Joe Urban*	El autor describe el lenguaje que desarrolló. Se trata de otro lenguaje que puede ser directamente ejecutado y que permite mejorar el análisis de requerimientos y las fases de diseño preliminares del ciclo de vida.
Z *M. Norris*	Z es un lenguaje de especificación formal desarrollado por la Universidad de Oxford. El documento es una serie de reportes publicados en los que se destaca que Z es una técnica rigurosa para la especificación de sistemas basado en especificaciones abstractas.
Métodos formales *Darrel Ince*	El autor revisa brevemente la historia de los métodos de especificación, del lenguaje natural y los métodos estructurados hasta llegar a los métodos formales.
Métodos formales *Douglas Barrie*	El autor sostiene que el estándar 00-55 (método formal para la especificación) alentará al gobierno de Reino Unido a invertir tiempo y dinero para adoptar una metodología formal, incorporar herramientas dar y capacitación al personal de sistemas.

IEEE Guide to Software Requirements Specifications **(Guía para las especificaciones de requerimientos de software de la IEEE). Estándar ANSI/IEEE 830-1984** Instituto de ingenieros en eléctrica y electrónica (IEEE). 1984.
Contenido
Este documento de la IEEE proporciona una guía estándar para la especificación de requerimientos de software, define diversos conceptos relacionados, así como las características de calidad que deben cumplir las especificaciones.

2.1.2 Lectura Ligera

Ingeniería de Software
Roger S. Pressman
Editorial Mc Graw Hill. 4ª Edición, 1998.

Capítulo	Contenido
1. El producto	El software se ha convertido en el elemento clave de la evolución de los sistemas y productos informáticos. No obstante, la cultura del desarrollo de software ha creado un conjunto de problemas que persisten en la actualidad. El objetivo de la ingeniería del software es proporcionar un marco de trabajo para construir el software con mayor calidad.
2. El proceso	La ingeniería de software es una disciplina que integra procesos, métodos y herramientas para el desarrollo de software. En el capítulo se describen diversos modelos para el desarrollo de software como son: Modelo Secuencial, Construcción de Prototipos, Desarrollo Rápido de Aplicaciones, Modelo Incremental y Modelo en Espiral.
6. Gestión del riesgo	Se proponen siete subcategorías genéricas de riesgos, las cuales son: tamaño del producto, impacto en el negocio, características del cliente, definición del proceso, entorno de desarrollo, tecnología a construir, tamaño y experiencia del equipo de trabajo. El autor, propone una serie de preguntas interesantes para identificar riesgos, algunas de las cuales son: • ¿Tiene el cliente una idea formal de lo que se requiere? ¿Se ha molestado en escribirlo? • ¿Aceptará el cliente "gastar" su tiempo en reuniones formales de requisitos para identificar el ámbito del proyecto? • ¿Está dispuesto el cliente a establecer una comunicación fluida con el analista/desarrollador? • ¿Está dispuesto el cliente a participar en las validaciones/verificaciones? • ¿Se preveen cambios a los requisitos del producto?

	¿Antes o después de la entrega? • ¿Se llevan a cabo revisiones técnicas formales de las especificaciones de requisitos? • ¿Se emplea algún mecanismo de control para mantener la consistencia entre los requisitos del sistema/software, diseño, código y casos de prueba? • ¿Hay algún mecanismo de control de cambios de los requisitos del cliente que afecten en el software? • ¿Se emplean técnicas de especificación de aplicaciones para ayudar en la comunicación entre el cliente y el desarrollador? • ¿Se emplean métodos específicos para el análisis del software?
10. Ingeniería de Sistemas	Un sistema consta de los siguientes elementos: software, hardware, personas, bases de datos, documentación y procedimientos. En el capítulo se presenta un panorama de la ingeniería de sistemas de computadora, la ingeniería de información y la ingeniería de productos. Un apartado del capítulo trata sobre la especificación del sistema, la cual es un documento que sirve como fundamento para la ingeniería, ya que describe la función y el rendimiento de un sistema basado en computadora y las restricciones que gobernarán su desarrollo. Se presenta además un esquema para la especificación de sistemas.
11. Conceptos y Principios del Análisis	El análisis de requisitos es la primera fase del proceso de ingeniería de software. En este punto se refina la declaración general del ámbito del software en una especificación concreta que sirve como base para todas las actividades siguientes del proceso. Puede dividirse en cinco áreas de esfuerzo: 1) Reconocimiento del problema, 2) Evaluación y síntesis, 3) Modelado, 4) Especificación y 5) Revisión. En esta sección se describen algunas técnicas para favorecer la comunicación entre el cliente y el equipo de trabajo en el desarrollo de las especificaciones y se ofrecen recomendaciones.

Specification and Transformation of Programs. A Formal Approach to Software Development
(Especificación y Transformación de Programas. Un Enfoque Formal para el Desarrollo de Software)
Helmut A. Partsch
Editorial *Springer-Verlag Berlin Heidelberg New York*. 1ª. Edición. 1990.
Biblioteca del IIMAS, UNAM. Clasif. QA76.76D47 P37. ISBN 3-540-52356.

Capítulo	Contenido
1. Introducción	El autor define la Ingeniería de Software como "el desarrollo y mantenimiento sistemático de software certificado que resuelve correctamente una tarea específica". Incluye además la definición de otros atributos de calidad del software como son: confiabilidad, robustez, eficiencia, portabilidad, adaptabilidad, mantenimiento, modularidad y reusabilidad. Se describen los modelos de ciclo de vida clásicos de Bohem y Blazer. Posteriormente se presentan la adaptación de ambos modelos enfatizando la importancia y ventajas de la especificación formal del problema y los requerimientos. Se trata también el concepto de programación transformante.
2. Ingeniería de Requerimientos	En este capítulo se reconoce la importancia de la ingeniería de requerimientos, ya que influye en los altos costos del software. Además se presentan algunas causas de la problemática relacionada con la ingeniería de requerimientos, entre ellas el uso del lenguaje natural, ya que puede resultar ambiguo. La ingeniería de requerimientos es considerada como el período de tiempo en el ciclo de vida del software durante la cual los requerimientos para un producto de software (como las características funcionales y de desempeño) son definidos y documentadas. De acuerdo con *Horowitz*, esta fase comprende dos actividades: especificación de requerimientos y análisis de requerimientos, que consisten en la traducción de una

	necesidad del usuario en una sentencia que describa las funciones a realizar por un sistema, seguida del análisis del costo y tamaño del sistema. Zave menciona tres actividades dentro de la ingeniería de requerimientos: identificación del problema, comprensión del problema y especificación del problema. Kühnel propone un ciclo con las siguientes tareas en la ingeniería de requerimientos: investigación de requerimientos, formulación de requerimientos, análisis de requerimientos, hasta llegar al acuerdo final de los requerimientos. Clasifica los requerimientos en: • Requerimientos funcionales • Atributos de calidad de las funciones deseadas • Requerimientos para la implementación de sistemas • Requerimientos para pruebas, instalación y mantenimiento • Requerimientos para el proceso de desarrollo Se presentan una serie de criterios esenciales para la definición de buenos requerimientos, así como algunas técnicas usadas en la especificación de requerimientos como son: diagramas de flujo, tablas de decisión, lenguajes formales, mecanismos de estados finitos, redes de Petri, SA/SADT, PSL/PSA, RSL/REVS, EPOS, Gist.
3. Especificación formal del problema	La propuesta del autor consiste en obtener una especificación formal del problema como resultado de la fase de ingeniería de requerimientos. En este capítulo propone los formalismos para una descripción precisa del problema a resolver. Considera a todas las técnicas de especificación tratadas en el capítulo dos, a excepción de Gist, como informales.

El Proceso Unificado de Desarrollo de Software
Ivar Jacobson, Grady Booch y James Rumbaugh
Editorial Addison Wesley 1ª. Edición en español. 2000.

Capítulo	Contenido
1. El Proceso Unificado	Un proceso de desarrollo de software es el conjunto de actividades necesarias para transformar los requisitos de

	un usuario en un sistema de software. El Proceso Unificado es un proceso de desarrollo de software, el cual: • Está dirigido por casos de uso • Está centrado en la arquitectura • Es iterativo e incremental En el capítulo se da una perspectiva general de las características del Proceso Unificado.
3. Un proceso dirigido por Casos de Uso	Los requisitos funcionales del sistema se representan con Casos de Uso. Los Casos de Uso son la base para el diseño, la implementación y las pruebas del sistema a lo largo del proceso de desarrollo. En el capítulo se describen los elementos de los casos de uso y cómo pasar de los casos de uso a los modelos de análisis y diseño. Además se resalta la importancia de los casos de uso en todo el proceso, sus beneficios y se presentan algunos conceptos relacionados para comprender y realizar los casos de uso.
6. Captura de requisitos: de la visión a los requisitos	En el capítulo se menciona por qué la captura de requisitos es un proceso complicado, y se da una visión general para realizarlo, considerando las siguientes actividades: • enumerar los requisitos candidatos (lista de características) • comprender el contexto del sistema (modelo del dominio o del negocio) • capturar requisitos funcionales (modelo de casos de uso) • capturar requisitos no funcionales (requisitos adicionales). Se señala el papel de los requisitos en el ciclo de vida del software y se dan los lineamientos para construir un modelo de dominio y un modelo del negocio que faciliten la comprensión del contexto.
7. Captura de requisitos como Casos de Uso	En este capítulo se detalla la construcción de los Casos de Uso, y se describe el flujo de trabajo de requisitos desde tres perspectivas:

	• Los artefactos creados • Los trabajadores participantes • El detalle de las actividades del flujo de trabajo.
8. Análisis	En la captura de requisitos se utiliza un lenguaje comprensible para el cliente y se tiene una vista externa del sistema, por lo que es probable que aún queden aspectos sin resolver relativos a los requisitos del sistema. El análisis se basa en la captura de requisitos y permite analizarlos con mayor profundidad, utilizando el lenguaje de los desarrolladores y reflexionando sobre aspectos internos del sistema, es decir, permite refinar los requisitos. En el capítulo se presenta una explicación más detallada del proceso de análisis, el refinamiento y la estructuración de requisitos.

Software Requirements: Analysis and specification
(Requerimientos de Software: Análisis y especificación)
Michael Davis
Editorial Prentice Hall
ISBN 0-13-824673-4

Capítulo	Contenido
2. Un punto de vista industrial *P. Worthington*	El autor define requerimiento como la petición del usuario de lo que quiere, y especificación como la descripción de cómo cumplir el requerimiento en el ambiente del software. Relata brevemente, de acuerdo a su experiencia, algunos aspectos relacionados con el desarrollo de requerimientos y las especificaciones de software en los sistemas militares.
4. Agregando formalidad al pragmatismo *K. Jackson*	Es conocido el hecho de que aún usando mejores herramientas y técnicas para desarrollar software no se ha logrado producir mejor software. Algunas veces se entiende por mejor técnica que ésta sea formal. En el capítulo se describe un método de diseño llamado MASCOT (Enfoque Modular para la Construcción, Operación y Pruebas de Software) y se señala cómo los métodos formales pueden ser introducidos en el desarrollo.

5. Definición de requerimientos de software *G. Rzevski*	En el capítulo se señala la importancia de encontrar los requerimientos precisos del software a desarrollar, antes de que el diseño del mismo sea emprendido, y se reconoce que desafortunadamente, la definición de requerimientos no siempre se hace adecuadamente; como consecuencia muchos productos de software no cubren las necesidades de los usuarios. El autor describe una serie de problemas asociados con esta actividad y además presenta una serie de atributos o requerimientos restrictivos del software.

Software That Works
(Software que Funciona)
Michael Ward
Editorial Academic, 1990.
Biblioteca del IIMAS, UNAM. Clasif. QA76.76D47 W37 ISBN 0-12-735040-3

Capítulo	Contenido
5. Prototipos	En este capítulo se señalan las ventajas que pueden tener los prototipos en algunos casos como parte de la especificación de los requerimientos de software; se comparan la técnica de diseño ascendente contra la de diseño descendente, siendo ambas válidas de acuerdo al problema, ya que por ejemplo, hay ciertos tipos de programas pequeños en lo que el diseño detallado no está justificado.
6. Análisis	El análisis significa dividir un todo en sus partes para encontrar su naturaleza. Para realizar el análisis se toman las piezas y se convierten en algo significativo, clasificándolas en categorías, para posteriormente organizarlas en un todo nuevamente, pasando así al diseño. El autor en este capítulo da un breve panorama de lo que se debe considerar al analizar un problema para pasar así al diseño.
7. Requerimientos	Uno de los elementos más importantes para comenzar un diseño, es tener una lista con los requerimientos que el proyecto debe cumplir. Cuando los diseñadores de software se niegan a considerar esta lista, están negando un elemento clave del diseño. El autor recomienda el uso de un Manejador de Bases de

	Datos para la gestión de los requerimientos, presenta un bosquejo del modelo relacional para ello; la lista de requerimientos no debe ser un documento muerto, si no en constante cambio hasta llegar a un documento de partida para una versión del diseño. Además señala que si no se tiene una lista de requerimientos no es posible realizar un análisis costo-beneficio decente.

A User's Guide for Defining Software Requirements
(Guía del Usuario para Definir Requerimientos de Software)
Carolyn Shamlin
Biblioteca del IIMAS, UNAM. Clasif. QA76 .76D47 S43 1989.

Capítulo	Contenido
1. El rol del usuario en el desarrollo de sistemas	En el capítulo de destacan los beneficios de involucrar a los usuarios en el desarrollo de especificaciones debido a la experiencia y dominio que tienen en su área de negocio. Inclusive la posibilidad de que sean los usuarios quienes desarrollen dichas especificaciones directamente. Dentro de todo el ciclo de desarrollo dividido en cuatro fases, se señalan las actividades en las que el usuario debe tener una responsabilidad y participación activa.
2. El Punto de partida	En esté capítulo se proporcionan algunos lineamientos para que un usuario especifique sus necesidades. Los pasos a seguir son: • Evaluar las necesidades externas del sistema • Con base en estas necesidades, establecer metas para la solución del problema • Para cada meta, establecer números que permitan lograr y verificar dicha meta. Se señala también la posibilidad de evaluar y adquirir paquetes comerciales en lugar de pensar desarrollar software a la medida, si es que éstos cubren las necesidades.

2.1.3 Lectura Rápida

Software Requirements & Specifications: A Lexicon of Practice, Principles and Prejudices
(Requerimientos y Especificaciones de Software: Un Léxico de Práctica, Principios y Prejuicios)
Michael Jackson.
Editorial Addison-Wesley. 1a. Edición. Julio, 1995.
Biblioteca de la Fac. de Ciencias, UNAM. Clasif. QA76 .76D47 J33. ISBN 0201877120.

Capítulo	Contenido
1. Introducción	El desarrollo de software es como construir una máquina: el sistema. Los elementos del mundo que serán afectados por la máquina constituyen el dominio de la aplicación. El autor hace una clara diferencia entre **qué** hace el sistema –lo cual está en el dominio de la aplicación– y el **cómo** lo hace –lo cual es el dominio de la solución. Los programas son descripciones de máquinas; los requerimientos son descripciones del dominio de la aplicación. Los ingenieros de software tendemos a enfocarnos a la solución, y los requerimientos se encuentran en el dominio de la aplicación.
2. El Léxico	El autor describe el impacto de la ambigüedad en las sentencias que utilizamos diariamente y cómo puede afectar esto al desarrollo del software. Describe una serie de técnicas para la especificación de requerimientos, la mayoría con un enfoque matemático y formal.

Modelo de Referencia de Procesos y Capacidades, Modelo de Evaluación de Procesos y Guía de Indicadores Según ISO 15504 (SPICE)
M. en C. María del Pilar Angeles
Asociación Mexicana para la Calidad en Ingeniería de Software (AMCIS)
Noviembre, 2000.

Tema

En este trabajo se describen las características, estructura y contenido del modelo ISO 15504 (SPICE).

Se agrupan los procesos en tres ciclos de vida: 1) Primario, 2) Organizacional y 3) Soporte. Éstos contienen cinco categorías de proceso, las cuales son: A) CUS. Categoría de Procesos Cliente-Proveedor, B) ENG. Categoría de Procesos de Ingeniería, C) SUP. Categoría de Procesos de Soporte, D) MAN. Categoría de Procesos de Administración y E) ORG. Categoría de Procesos de Organización.

Cada categoría de procesos, contiene una serie de prácticas base, muchas de las cuales están relacionadas con la ingeniería de requerimientos, principalmente de las categorías CUS, ENG y SUP.

2.2 Tesis

Cuestionario para la Evaluación del Proceso de la Administración de la Calidad basado en ISO 15504 (SPICE)
María Guadalupe Hernández White
Maestría en Ciencias de la Computación. IIMAS. UNAM. Año 2001. Clasif. 218.

En el trabajo se pretende brindar a través de un conjunto de cuestionarios, una herramienta que permita mejorar la calidad en el desarrollo de software.

Los cuestionarios propuestos están dirigidos a tres roles involucrados en el proceso: cliente, líder de proyecto y participante. Entre las preguntas de dichos cuestionarios se considera la documentación de los requerimientos, así como el conocimiento y disponibilidad de estos documentos como aspecto importante para mejorar la calidad del software.

Modelado de Áreas Clave de Procesos para CMM Nivel 2
Claudia Alquicira Esquivel
Maestría en Ciencias de la Computación. IIMAS. UNAM. Año 2000. Clasif. 215.

El objetivo de la tesis es construir un modelo gráfico usando diagramas de actividad de UML de las áreas clave de proceso del CMM nivel 2 para facilitar la comprensión del mismo.

Propone además una serie de cuestionarios de apoyo para evaluar y mejorar las áreas clave de proceso, entre ellas la Administración de Requerimientos.

Guía para el Proceso de Mejora en el Desarrollo de Software
José Gabriel Ramírez López
Maestría en Ciencias de la Computación. IIMAS. UNAM. Año 2001. Clasif. 213.

El objetivo del trabajo es apoyar a las organizaciones mexicanas y latinoamericanas en la aplicación del proceso de mejora en el desarrollo de software, proporcionándoles una guía de manera resumida y complementada con una parte gráfica que facilite su interpretación y aplicación.

La guía propuesta está basada en el modelo ISO 15504, el cual considera diversas prácticas relacionadas con la Ingeniería de Requerimientos.

Guía de Autoevaluación de la Categoría de Procesos de Ingeniería de Software bajo el Estándar ISO-15505 (SPICE)
Víctor González Castro
Maestría en Ciencias de la Computación. IIMAS. UNAM. Año 2000. Clasif. 208.

El trabajo es una guía de autoevaluación que ayuda a las empresas de desarrollo de software a generar planes de mejora de procesos, y a entender y aplicar el estándar ISO-15504. Este estándar es un modelo de referencia que describe los procesos que una organización puede realizar ya sea para adquirir, proveer, desarrollar, operar, evolucionar y soportar algún software, así como los atributos que caracterizan la capacidad de cada uno de los procesos.

Dentro de la categoría de los Procesos de Ingeniería del estándar ISO-15504 se encuentran los siguiente procesos relacionados con la Ingeniería de Requerimientos:

- El proceso de análisis y diseño de los requerimientos del sistema.
- El proceso de análisis de los requerimientos de software.

La tesis menciona otros modelos como son CMM, ISO 9000 y BOOTSTRAP.

Guía para la Administración de Proyectos Pequeños de Software, cumpliendo CMM-2 y guiados por el PMBOK (Project Management Body of Knowledge)
Blanca Lucía Gil Castellanos
Maestría en Ciencias de la Computación. IIMAS. UNAM. Año 2000. Clasif. 205.

El objetivo de la tesis es presentar una metodología para construir software de forma que su desarrollo sea planteado como un proyecto y poder aplicar las mejores prácticas en la Administración de Proyectos que el *Project Management Body of Knowledge* propone, incluyendo los requisitos para cumplir el nivel 2 del CMM.

En el marco teórico del trabajo se presentan diversos conceptos relacionados con la Administración de Proyectos y ubica esta actividad como elemento clave del nivel 2 del CMM. Describe además los niveles de este modelo y las características de las áreas clave del proceso, entre ellas la Administración de Requerimientos.

2.3 Revistas

Software Development Magazine
http://www.sdmagazine.com

Artículo	Contenido
Give them what they want (Démosles lo que quieren) *Doug Rosenberg and Kendall Scott* Junio, 2001	La revisión de requerimientos es una parte esencial del proceso de modelado. En el artículo se presentan algunos lineamientos para asegurar que los Casos de Uso y el modelo de dominio trabajen juntos y cumplir así adecuadamente los requerimientos funcionales de los usuarios finales. Se mencionan diez de los errores más comunes cuando se hace una revisión de los requerimientos.
Top ten use case mistakes *(Los diez errores más comunes en los Casos de Uso)* *Doug Rosenberg and Kendall Scott* Febrero, 2001	Los Casos de Uso son la parte del UML que permite representar los requerimientos de un sistema. La construcción de Casos de Uso es una de las primeras etapas en los procesos de desarrollo. En el artículo se mencionan diez errores muy comunes en la construcción de Casos de Uso, por ejemplo: no escribir requisitos funcionales, dependencia de la interfaz de usuario, no escribirlos desde una perspectiva de usuario y omitir el texto de flujos alternos. Se presentan ejemplos de

	estos errores.
Creating the project plan *William H. Roetzheim* Enero, 2001	Una de las tareas más difíciles para un administrador de proyectos es crear una línea base para el desarrollo. EL artículo describe como se puede complementar el costo y tiempo del proyecto para crear un plan completo. Una de las tareas del proyecto es precisamente la definición de requerimientos.
Writing good requirements ***(Escribiendo buenos requerimientos)*** *Karl Wiegers* Septiembre, 2000	Si la calidad de un producto final depende de la calidad de la materia prima, una especificación pobre de requerimientos no contribuirá a obtener software de calidad. Muchas especificaciones de requerimientos de software (SRS) se llenan de requerimientos mal escritos. En el artículo se describen varias características que deben tener las sentencias de requerimientos de software de alta calidad. Entre ellas se encuentran: ser correcto, ser ponderado, ser entendible, ser necesario para el proyecto, no ser ambiguo, verificable, completo y modificable.
First things first: prioritizong requirements ***(Primero lo primero: Priorizar requerimientos)*** *Karl Wiegers* Septiembre, 2000	Cualquier proyecto con restricciones o no, debe ponderar las características solicitadas. Establecer prioridades permite al administrador del proyecto resolver conflictos, planear entregas parciales y tomar decisiones. En al artículo se presentan dos escalas típicas en la ponderación de requerimientos: • Alta, Media y Baja • Esencial, Condicional y Opcional. Además e describen una serie de pasos de un modelo de ponderación de requerimientos.
How to avoid use-case pitfalls ***(Cómo evitar el peligro en los casos de uso)*** *Susan Lilly* Enero, 2000	Los Casos de Uso son una técnica popular para documentar los requerimientos de sistema y de software. En el artículo se presentan los diez peligros más comunes de los casos de uso y se dan algunas recomendaciones para prevenirlos.
Capturing business rules ***(Capturando las reglas del negocio)*** *Ellen Gottesdiener*	Las reglas del negocio proporcionan el conocimiento detrás de cada proceso o estructura de negocio. Son por lo tanto el foco de los requerimientos funcionales. En el proceso de requerimientos son importantes las

Diciembre, 1999	técnicas que serán usadas para captar los requerimientos (por ejemplo: entrevistas, mesas de trabajo y prototipado), las herramientas para representar y manejar la información, así como el hecho de que los clientes y usuarios estén activamente involucrados en el proceso. En al artículo se señala la importancia de las reglas del negocio y algunas consideraciones para identificarlas y representarlas adecuadamente.
Customer rights and responsibilities *(Derechos y Responsabilidades de los Clientes)* *Karl Wiegers* Diciembre, 1999	El éxito del software depende en gran medida de la colaboración entre los desarrolladores de software y sus clientes y usuarios. Los problemas surgen porque no se tiene una comprensión clara de cuáles son los requerimientos y quiénes son los usuarios. El autor propone una interesante lista de los derechos y responsabilidades de los clientes y usuarios en los proyectos de desarrollo de sistemas.
Decoding business needs (Descifrando necesidades de negocio) *Ellen Gottesdiener* Diciembre, 1999	Facilitando los talleres y las mesas de trabajo, se reduce el riesgo de que el alcance crezca, se acelera la entrega en fases tempranas del ciclo de vida y se ahorra tiempo y esfuerzo. El *Standish Group* sugiere que los cuatro factores de éxito para un proyecto de Tecnologías de Información son: • Involucrar al cliente/usuario • Apoyo de la alta dirección • Definición clara de los requerimientos • Desarrollar una adecuada planeación El autor señala que el éxito del software depende de cinco importantes factores: • El personal del negocio debe tener roles formales en los proyectos de IT • El personal del negocio debe participar desde el inicio • Los desarrolladores no deben empezar un proyecto sin patrocinio del negocio • Si involucrar al negocio es un factor de riesgo, debe ser considerado en la estrategia de administración de riesgo

	• El proceso y las técnicas deben maximizar la colaboración del cliente/usuario. Para ello, hay tres maneras de involucrar a los clientes del negocio en el proceso de desarrollo de software, las cuales se describen en el artículo: • Establecer una plan del proyecto • Facilitar los talleres y mesas de trabajo y • Direccionar el cambio organizacional.
A software metrics primer ***(Una Primer métrica de software)*** *Karl Wiegers* Julio, 1999	En el artículo de denota la importancia de las métricas de software, ya que permiten cuantificar el tiempo, esfuerzo y tamaño de un proyecto de desarrollo, así como su estado y sus características de calidad. Representan un aspecto importante en los programas de mejora de procesos. Goal-Question-Metric (GMQ) es una excelente técnica para seleccionar métricas apropiadas a necesidades particulares. Las métricas se pueden clasificar en tres niveles de grupos participantes en el desarrollo: desarrolladores, equipos del proyecto y equipos de desarrolladores. Una de las métricas que se deben establecer inicialmente se refiere a la distribución del esfuerzo en cada actividad, entre ellas la especificación de requerimientos. Otras métricas se refieren al tamaño del producto, la duración estimada contra la real, y los defectos encontrados.
Building better business systems with scenarios ***(Construyendo mejores sistemas de negocio con escenarios)*** *Carlos Jerome* Junio, 2000	La comprensión del dominio del problema es primordial para el diseño de la solución. Los Casos de Uso son el método más utilizado por la comunidad de la orientación a objetos para especificar requerimientos y por lo tanto para manejar todo el proceso de desarrollo. Un escenario tiene la misma estructura de un Caso de Uso. Los escenarios permiten ilustrar circunstancias reales que pueden presentarse y que se deben prever para una mayor comprensión del problema. Se consideran clave para el éxito en la construcción de aplicaciones. En el artículo se presentan algunos puntos a considerar en la construcción efectiva de escenarios y se resalta su importancia y utilidad dentro del proceso.

Requirements Engineering patterns ***(Patrones de Ingeniería de Requerimientos)*** *Scott Ambler* Mayo, 2000	La Ingeniería de Requerimientos –obtención, documentación y validación de requerimientos– es un aspecto fundamental en el desarrollo de software. Sin embargo no se le da la importancia adecuada, porque lo que se desea obtener es código lo más pronto posible. En el artículo se resalta la importancia de la Ingeniería de Requerimientos en el proceso de desarrollo y se presenta el modelo de Proceso Unificado.
Requirements by pattern ***(Requerimientos por patrón)*** *Christopher Creel* Diciembre, 1999	Debido a la obsesión de la industria del software de tener los productos finales, han surgido lenguajes y herramientas de desarrollo de todas las formas, tamaños y colores. Pero todo esto no resuelve el problema si nosotros primero no conocemos el espacio del problema. El propósito de crear patrones es capturar aspectos que puedan ser reutilizables. Se presentan tres patrones que son: especificación, presentación y ponderación de requerimientos.

STQE Magazine Http://www.stqemagazine.com	
Artículo	**Contenido**
Requirements when the field isn't green ***(Requerimientos cuando el campo no es verde)*** *Karl E. Wiegers* Mayo/Junio 2001	Muchos desarrolladores trabajan en proyectos de mantenimiento de sistemas legados, añadiendo nuevas características, modificando funcionalidad o comunicándolos con otros sistemas para cumplir con las reglas del negocio. No es raro que dichos sistemas no se encuentren documentados y que dichas modificaciones tampoco se documenten en una especificación de requerimientos. Sin embargo, es posible aplicar diversos métodos de la ingeniería de requerimientos, aún en estas situaciones. En el artículo se describen siete principios útiles para estos casos que pueden ayudar para mejorar la calidad y facilitar el futuro mantenimiento: 1) Dedicar tiempo a documentar las características descubiertas en los sistemas al realizar la ingeniería inversa, que sirva de referencia para futuras modificaciones

2) Practicar en la primera oportunidad las técnicas de ingeniería de requerimientos
3) Reconstruir los requerimientos selectivamente
4) Probar los requerimientos al desarrollarlos
5) Seguir un proceso de control de cambios
6) Inspeccionar la cadena de rastreo
7) Empezar cuanto antes.

2.4 Cursos

✓ **Rational Unified Process Fundamentals**
(Fundamentos del Proceso Unificado de Rational)
Sede: ITERA. Jose Luis Lagrange 103 8° piso. Los Morales Polanco, México, D.F.
Fechas: 6 y 7 de Diciembre del 2001
Impartido por: Mtra. Mónica Alegría Vázquez

Objetivos del curso: 1) Comprender las seis mejores prácticas para el desarrollo de software; 2) Describir el Proceso Unificado en términos de sus fases y disciplinas; 3) Comprender los beneficios de los proceso dirigido por casos de uso y una arquitectura centralizada; 4) Describir el enfoque iterativo para el desarrollo de software y 5) Comprender la adaptabilidad del Proceso Unificado de acuerdo a la organización y al proyecto.

El curso se dividió en 6 módulos:

1. Mejores prácticas para el desarrollo de software
2. Estructura del proceso
3. Disciplinas
4. Implementación del proceso
5. Soluciones de comercio electrónico.

Building better software with Use Cases
(Construyendo mejor software con Casos de Uso)
Sede: Construx Software Builders, Inc.
Fechas: 5 y 6 de Febrero de 2002
Impartido por: Meilir Page-Jones

Los Casos de Uso son una técnica importante para obtener, organizar y verificar los requerimientos del sistema o negocio del usuario. En el curso se explica qué

✓ Curso tomado.

son y como usar los Casos de Uso para comprender y validar los requerimientos del usuario de manera rápida y precisa.
Temario:

1. Fundamentos de casos de uso
2. Consejos prácticos para desarrollar casos de uso
3. Aplicando las fortalezas de los eventos del negocio a los proyectos de software
4. Extendiendo los beneficios de los casos de uso a todo el proyecto.

Real world requirements
(Requerimientos del mundo real)
Sede: Construx Software Builders, Inc.
Fechas: 28 y 29 de Enero de 2002
Impartido por: Pamela Perrot

El curso presenta técnicas prácticas para explorar las necesidades del usuario, capturar requerimientos, controlar cambios y construir software altamente satisfactorio.

Temario:

1. El proceso de requerimientos
2. Definición de requerimientos de alta calidad
3. Ambigüedad
4. Descubriendo requerimientos
5. Técnicas de especificación
6. Procesos de requerimientos
7. Administración de requerimientos.

In search of excellent requirements
(En búsqueda de requerimientos excelentes)
Software Productivity Center
Impartido por: Douglas Muir

El curso proporciona un conjunto de prácticas basadas en sesiones prácticas en grupo que pueden ser utilizadas para mejorar la calidad del desarrollo y la administración de requerimientos en cualquier organización.

Temario:

1. Introducción a la ingeniería de requerimientos
2. Desarrollo de requerimientos de software

3. CMM
4. Administración de requerimientos de software
5. Prácticas clave en la ingeniería de requerimientos.

2.5 Conferencias

✓ **Procesos de construcción de software. CMM, SPICE: Mayor calidad en el software**
Vanguardia Tecnológica .NET.
Sede: Auditorio "Maestro Carlos Pérez del Toro" de la Facultad de Contaduría y Administración de la UNAM.
Fecha: 17 de Enero del 2002.

En la plática de "Procesos de Construcción de Software" ofrecida por un ingeniero de software de la empresa CERTUM (www.certum.com) se resaltó la importancia de los estándares ISO 15504 (SPICE) e ISO 14143 para mejorar la calidad en los productos y los servicios de software y lograr así una mayor competencia al nivel de las empresas internacionales. Además se señalaron los 3 aspectos que integran el trinomio de la calidad: costo, tiempo y fiabilidad (esta última relacionada con el número de defectos).

2.6 Seminarios

✓ **MEGA TRENDS in Software Development by Ivar Jacobson**
(Tendencias en el desarrollo de software)
Sede: Hotel Marriot. Reforma. México, D.F.
Fecha: 7 de Junio, 2001.

En el seminario participó Ivar Jacobson, uno de los tres veteranos en el enfoque de orientación a objetos, y quien propuso el Proceso Unificado de Desarrollo y los Casos de Uso. Dentro de las tendencias se resaltó la importancia de dedicar más tiempo a las primeras fases (análisis-diseño), facilitando la implementación y promoviendo la reutilización. Además se recomendó utilizar herramientas que apoyen este proceso bajo el enfoque de objetos.

Otras empresas como Elektra en México, platicaron sus experiencias con la adopción las herramientas Rational.

✓ Presenciado.

✓ **Seminario Rational**
Sede: Hotel Marriot. Reforma. México, D.F.
Fecha: Septiembre, 2001.

La empresa Rational presentó un panorama de todas sus herramientas y las mejores prácticas en el desarrollo de software. Se ofreció una introducción de lo que es el modelado de negocios con UML, lo cual consiste en la representación de una organización tanto interna como externamente, identificando su ambiente, sus actores, sus procesos y sus flujos de trabajo. Para ello se utilizan también UML y los estereotipos.

Success starts with Requirements Management
(El éxito empieza con Administración de Requerimientos)
Sede: Marriot City Center Hotel. Pittsburg.
Fecha: 31 de Enero de 2002.

El tener el control de los requerimientos del proyecto es un paso importante para entregar una aplicación o sistema que satisfaga las necesidades del usuario y de su negocio. El seminario ofrece las técnicas para:

- Comprender y comunicar las necesidades reales del negocio detrás del sistema o aplicación
- Administrar los requerimientos más efectivamente
- Usar y manejar los casos de uso para describir la funcionalidad del sistema
- Organizar y ponderar el aumento y los cambios en los requerimientos.

✓ **Primer Seminario Internacional en Calidad en Desarrollo de Software**
Sede: ITAM, Campus Santa Teresa.
Fecha: Del 27 de Mayo al 7 de Junio del 2002.

El seminario tuvo como objetivo promover y apoyar los esfuerzos por mejorar los procesos de desarrollo de software en México, así como difundir algunas de las alternativas más exitosas para desarrollar software de alta calidad, contribuyendo así al Programa para el Desarrollo de la Industria del Software de la Secretaría de Economía.

Participaron el Dr. Pankaje Jalote -de la India-, Daniel Roy, la AMCIS y algunas de las principales empresas de software en México. Se trataron diversos tópicos entre los que resaltan: mejoramiento de procesos de software, CMM, ISO 9000,

✓ Presenciado.

administración de proyectos, TSP, PSP, métricas de software y pruebas de software.

2.7 Seminarios por Web

- **Adopting RM in existing projects**
  Adoptando la Administración de Requerimientos en proyectos existentes
  http://www.rational.com/events/webminars
  Fecha: 22 de enero de 2002

  En el seminario se presentan diez síntomas comunes de los proyectos mediocres, entre ellos extensión de las horas de trabajo, usuarios insatisfechos, costos sobrepasados, entre otros. La mayoría de estos síntomas tienen como causas raíces la pobre definición de requerimientos y la mala estimación del alcance del proyecto. Se presentan cinco pasos para un mejoramiento incremental en la administración de requerimientos.

- **Uncover the benefits of using a Requirements Management tool**
  (Descubriendo los beneficios de usar una herramienta para la Administración de Requerimientos)
  http://www.rational.com/events/webminars
  Fecha: 29 de enero de 2002

  En la presentación se consideraron los siguientes puntos: 1) La importancia de administrar requerimientos; 2) Antecedentes; 3) Problemas típicos en el desarrollo de software; 4) Una solución para la administración de requerimientos; 5) Demostración de Requisite Pro.

  Se mencionan que una de las causas por las que los proyectos fracasan es precisamente por una mala administración de requerimientos; se clasifican los requerimientos en funcionales, no funcionales y de restricción; y se consideran 3 actividades importantes para superar estos problemas: organizar, comunicar y manejar los cambios.

- **Five steps in problem analysis – How to ensure you solve the right problem**
  (Cinco pasos para el análisis del problema. Cómo asegurarse de resolver el problema correcto)
  http://www.rational.com/events/webminars
  Fecha: 5 de febrero de 2002

  Se describen cinco tareas que se deben considerar en el análisis del problema: 1) Llegar a un acuerdo sobre la definición del problema; 2) Comprender las causas raíces: el problema tras del problema; 3) Identificar a todos lo involucrados con el sistema; 4) Definir el límite de la solución del sistema; y 5) Identificar las restricciones que limitan la solución.

- **Writing good Use Cases with Rational Requisite Pro and Rational Rose**
  (Escribiendo buenos Casos de Uso con Rational RequisitePro y Rational Rose)
  http://www.rational.com/events/webminars
  Fecha: 14 de febrero de 2002

  Se consideró la siguiente agenda: 1) Casos de uso y requerimientos; 2) Qué es un caso de uso; 3) Diagramas de casos de uso con UML; 4) Especificaciones de casos de uso; 5) Administración de casos de uso; 6) Sugerencias y tips sobre casos de uso.

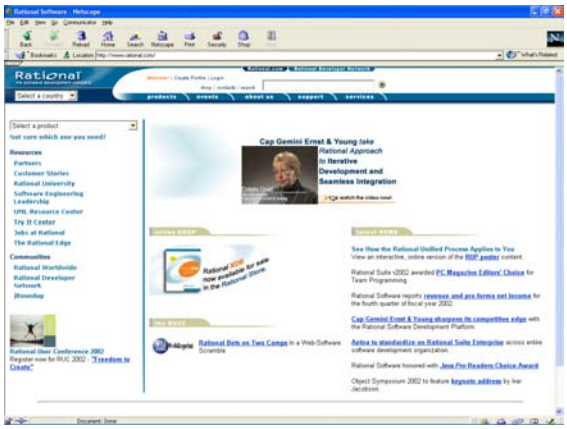
- **The principles behind writing good requirements**
  (Los principios detrás de escribir buenos requerimientos)
  http://www.rational.com/events/webminars
  Fecha: 26 de febrero de 2002

  En el seminario se presentaron los distintos tipos de requerimientos de software (funcionales y no-funcionales), algunos principios para escribir buenos requerimientos, los signos de alerta de algunos problemas, los niveles de detalle requerido y cómo ayudan los casos de uso a especificar los requerimientos de software.

2.8 Internet

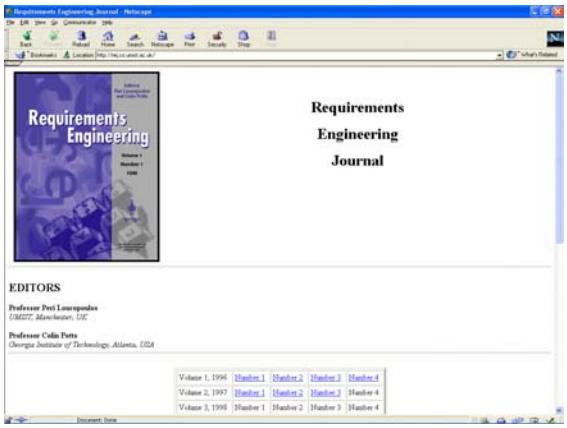
Los principales sitios de Internet consultados son los siguientes:

El Instituto de Ingeniería de Software (http://www.sei.cmu.edu) es un centro de investigación y desarrollo patrocinado por el Departamento de Defensa de los Estados Unidos, y residente en la Universidad de Carnegie Mellon, cuyo principal objetivo es ayudar a mejorar las capacidades en la ingeniería de software.

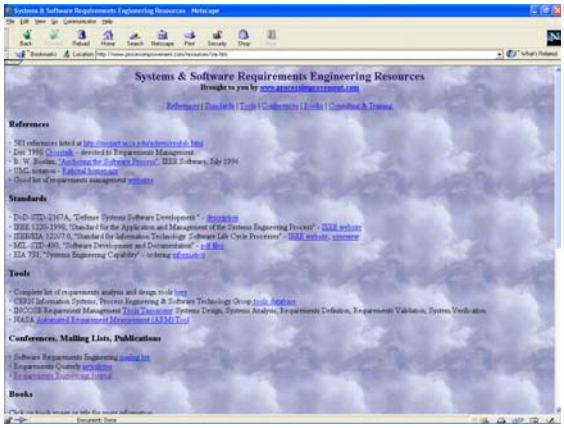


Rational Software (http://www.rational.com) es una de las principales empresas proveedoras de herramientas para apoyar para el desarrollo de software en todo el ciclo de vida. Una de las 6 prácticas a las que se enfoca, es la Administración de Requerimientos.

Rational Edge (http://www.therationaledge.com) es un sitio de información gratuita sobre tendencias, técnicas y herramientas de ingeniería de software, dedicado principalmente a los involucrados con la tecnología de Rational. Se publican mensualmente artículos de interés.

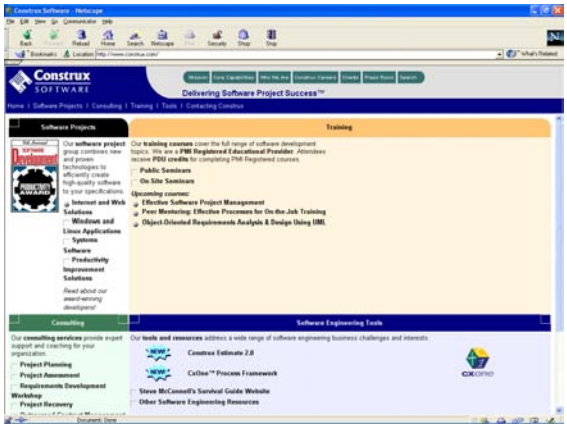


La publicación de Ingeniería de Requerimientos (http://rej.co.umist.ac.uk/) se enfoca a difundir información sobre la obtención, representación y validación de requerimientos. Además el sitio ofrece ligas de interés sobre el tema.

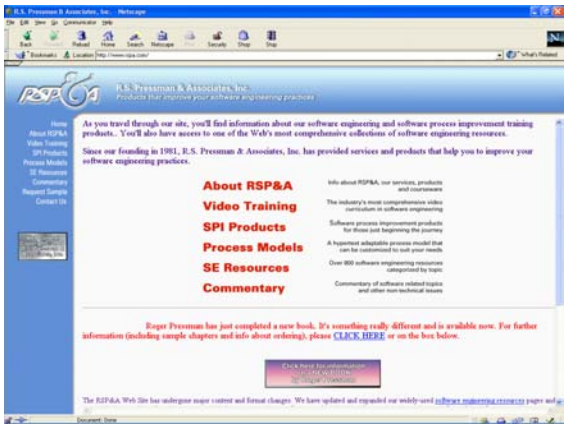


Process Improvement Associates es un sitio en el que se ofrece información sobre el mejoramiento de los procesos de software. Uno de las áreas es la ingeniería de requerimientos.Se ofrecen referencias, estándares, herramientas, eventos y libros relacionados. (http://www.processimprovement.com/resources/sre.htm)

Telelogic es un proveedor de herramientas para apoyar el desarrollo de software. Cuentan con herramientas para la administración de requerimientos. Además en el sitio se ofrecen artículos e información de interés sobre requerimientos. (http://www.telelogic.com)

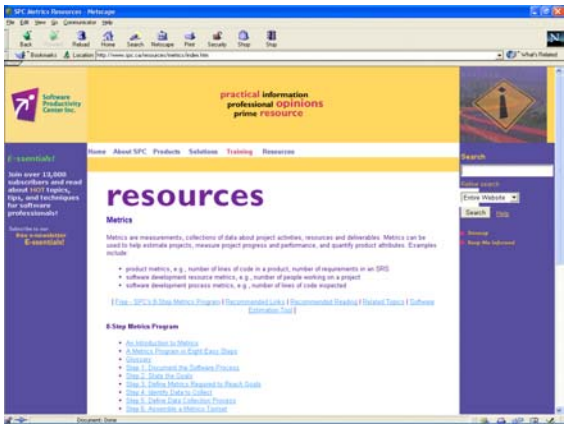


Construx Software es una empresa dedicada a la creación de software, capacitación y asesoría en los procesos de desarrollo (http://www.construx.com) En el sitio se ofrece información muy interesante sobre el proceso de desarrollo de software y ligas a revistas de interés como la "Software Development Magazine".

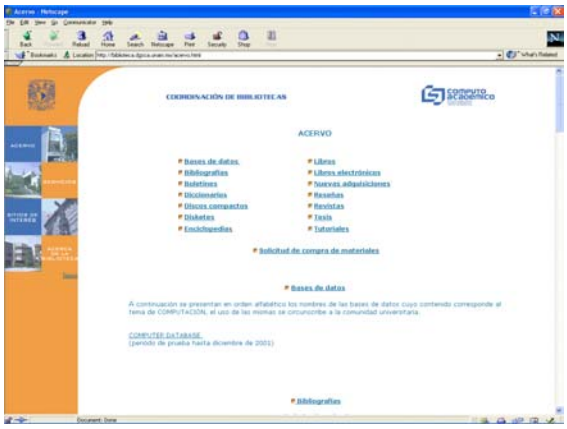


Roger Pressman es uno de los contemporáneos más reconocidos en la Ingeniería de Software. En el sitio de su empresa: (http://www.rspa.com) ofrece información de interés y referencias sobre todo el proceso de desarrollo de software.

El IEEE es una de las organizaciones más involucradas en el cómputo y la ingeniería de software. De hecho es quien formalizó el concepto de Ingeniería de Requerimientos. En su sitio especializado en cómputo (www.computer.org) ofrece artículos e información de interés.



En el sitio del Centro de Productividad de Software se ofrecen recursos sobre diversos temas y aspectos del desarrollo de software, así como sobre la mejora de procesos.
(http://www.spc.ca)



La biblioteca de la Dirección General de Servicios de Cómputo Académico (DGSCA) de la UNAM, ofrece acceso a una serie de recursos de información sobre cómputo y bases de datos especializadas. http://biblioteca.dgsca.unam.mx

Requirements Engineering
(Ingeniería de Requerimientos)
Merlin Dorfman
Software Engineering Institute (SEI). Carnegie Mellon University.
http://www.sei.cmu.edu
Artículo reimpresión del libro "Software Requirements Engineering", segunda edición. De Richard H. Thayer y Merlin Dorfman.

Sección	Contenido
Introducción	La deficiente especificación de requerimientos fue una de las causas más importantes de lo que en los años 60's se denominó "Crisis del Software". Algunos beneficios de la ingeniería de requerimientos son: lograr acuerdos entre los desarrolladores, clientes y usuarios; tener una base efectiva para la estimación de recursos; mejorar atributos de calidad como facilidad de uso y mantenimiento; y lograr objetivos con el mínimo de recursos.
Ingeniería de requerimientos y el ciclo de vida de desarrollo	Existen varios modelos del ciclo de vida del software, como por ejemplo: Modelo en Cascada, Desarrollo por Prototipos, Desarrollo incremental y Modelo en Espiral. Todos ellos incluyen una o más fases relacionadas con los requerimientos.
Requerimientos de sistemas y requerimientos de software	Los requerimientos de sistema y los requerimientos de software generalmente son tratados de la misma manera, ya que las herramientas y métodos para obtenerlos y las técnicas de representarlos son similares; no obstante existen importantes diferencias. Los requerimientos de sistema describen el comportamiento del sistema desde una perspectiva externa, por ejemplo desde la perspectiva del usuario, aunque muchas veces los documentos y especificaciones no pueden ser fácilmente comprendidos por los usuarios. Los requerimientos de software sirven para comunicar a los desarrolladores los elementos técnicos de software y hardware que describen las características del sistema.

Fundamentos de la ingeniería de requerimientos	Existen diversas taxonomías de la ingeniería de requerimientos. Por ejemplo: • Obtención, determinación de soluciones, especificación y mantenimiento. • Obtención, análisis, especificación, validación/verificación y administración. Cualquier sistema puede ser descompuesto en elementos jerárquicos más pequeños. Los requerimientos de sistema se ubican en la parte más general. La obtención de requerimientos de sistema consiste en trabajar con los usuarios para determinar sus necesidades. Cada requerimiento es asignado a niveles más específicos (subsistemas), en donde son detallados; los requerimientos de más bajo nivel son llamados derivados. El nivel de detalle se incrementa conforme se avanza en la jerarquía. Es por ello que uno de los aspectos clave en la ingeniería de requerimientos es la descomposición y la abstracción. El número de requerimientos prolifera rápidamente conforme se van detallando los requerimientos del sistema, el rastreo consiste en conocer la relación entre requerimientos de distintos niveles. Se presentan los atributos de una buena especificación de requerimientos de Bohem.

Software Requirements. SEI Curriculum Module SEI-Cm-19-1.2 (Requerimientos de Software)
John W. Brackett, Universidad de Boston
Software Engineering Institute (SEI). Carnegie Mellon University.
http://www.sei.cmu.edu

Contenido General

El documento es una guía para la impartición de un curso relacionado con la Ingeniería de Requerimientos, es decir, con el proceso de identificar, analizar, representar y comunicar requerimientos, así como desarrollar criterios de aceptación. Además presenta interesante bibliografía relacionada con el tema.

What is Requirement Management
(Qué es Administración de Requerimientos)
Richard Stevens & James Martín
Software Engineering Institute (SEI). Carnegie Mellon University.
http://www.sei.cmu.edu
Artículo reimpresión del libro "Software Requirements Engineering", segunda edición. De Richard H. Thayer y Merlin Dorfman.

Contenido General

La Administración de Requerimientos comienza con la definición de los requerimientos y continúa durante todo el proyecto, culminando con la aceptación del producto que cumple los requisitos acordados. Permite asegurarnos de:

- Conocer lo que quieren los clientes (calidad)
- Tener una solución eficiente para satisfacer esos requerimientos (conformidad)

El documento trata además brevemente los siguientes puntos: actividades de la administración de requerimientos; requerimientos para comunicar lo que se quiere; requerimientos para guiar el diseño y la implementación; requerimientos para manejar el cambio; requerimientos para probar el producto; requerimientos para administrar el proyecto; características de los requerimientos; y generalidades de las herramientas para la administración de requerimientos.

Software Requirements Engineering knowledge units
(Unidades de conocimiento de la Ingeniería de Requerimientos de Software)
Software Engineering Institute (SEI). Carnegie Mellon University.
http://www.sei.cmu.edu

Contenido General

Presenta de manera general la descripción de las sub-áreas de la ingeniería de requerimientos de software que son:

- Identificación de requerimientos (Requirements elicitation)
- Análisis de requerimientos (Requirements analysis)
- Especificación de requerimientos (Requirements specification).

When telepathy won't do: Requirements Engineering key practices
(Cuando la telepatía no funciona: prácticas clave de Ing. de Requerimientos)
Karl E. Wiegers
Cutter IT Journal. Mayo, 2000.
http://www.cutter.org

Sección	Contenido
Un marco de la Ingeniería de	La Ingeniería de Requerimientos es una actividad principalmente de comunicación, no técnica.

Requerimientos	Un proyecto tiene tres niveles de requerimientos: requerimientos de negocio, requerimientos de usuario y requerimientos funcionales.
Identificación de requerimientos	Se señalan como prácticas clave: • Definir los requerimientos de negocio • Involucrar al usuario • Enfocarse en las tareas del usuario • Definir atributos de calidad.
Especificación de requerimientos	Se señalan como práctica clave almacenar los requerimientos en una herramienta automatizada. Se mencionan algunas como: Caliber-Rm, DOORS y RequisitePro.
Verificación de requerimientos	Se señala como práctica clave inspeccionar las especificaciones de requerimientos.

Towards the Engineering Requirements
(Hacia la Ingeniería de Requerimientos)
Tom Gilb
Requirements Engineering Journal
http://www.rej.com

Contenido General

El documento da respuesta a algunas preguntas como:

- ¿Qué es ingeniería?
- ¿Qué son los falsos requerimientos?
- ¿Qué son los requerimientos?
- ¿Cómo se relacionan los requerimientos con el diseño?
- ¿Cómo se relacionan los requerimientos con la ingeniería?
- ¿Cómo mejorar las prácticas de Ingeniería de Requerimientos?
- ¿Cuáles son los principios fundamentales de los Requerimientos?

Software Engineering checklist
(Listas de verificación de Ingeniería de Software)
R.S. Pressman & Associates, Inc.
http://www.rspa.com/checklists

Contenido

Las listas de verificación (checklists) son parte del Modelo de Proceso Adaptable (APM-Adaptable Process Model) propuesto por R.S. Pressman & Associates, y son utilizadas para evaluar la calidad de los productos de la ingeniería de software y del proceso del software.

Cada lista de control contiene una serie de preguntas de acuerdo al producto o proceso que se esté evaluando. Las disponibles son las siguientes:

- Revisando el Modelo de Análisis
- Análisis de Factibilidad
- Evaluando el Plan del Proyecto
- Estimaciones de Costo y Tiempo
- Calidad con ISO 9001
- Revisando el Modelo de Diseño de Software
- Asegurando la Calidad de las Aplicaciones en Web.

Las dos primeras están muy relacionadas con la actividad de requerimientos, no obstante en las otras existe información complementaria a este tema.

Some tips for requirements elicitation
(Algunos tips para la obtención de requerimientos)
Karl E. Wiegers
http://spc.ca/essentials

Contenido

La obtención es el proceso de identificar y comprender las necesidades, problemas y restricciones de los distintos tipos de usuarios para un producto de software propuesto. El autor brinda algunos consejos para mejorar aspectos de la obtención de requerimientos.

Rational Requisite Pro. Evaluators guide
(Guía del evaluador de Rational Requisite Pro)
Rational Software Corp.
http://www.rational.com

Contenido

El documento es una guía para evaluar Requiste Pro, una herramienta de software para la Administración de Requerimientos, desarrollada y comercializada por Rational Software Corp.

En la guía se mencionan algunos aspectos técnicos de la herramienta además de presentar diversos conceptos e información relacionada con la Administración de Requerimientos desde la perspectiva de Rational Software Corporation.

The top risk of Requirements Engineering
(Los principales riesgos de la Ingeniería de Requerimientos)
Brian Lawrence, Karl Wiegers y Christol Ebert
IEEE Software. Noviembre-Diciembre 2001.
http://www.computer.org

Contenido

El punto central de la ingeniería de requerimientos es guiar el desarrollo para producir el software correcto. El no obtener los requerimientos correctamente, afectará el éxito del proyecto. Los autores identifican algunos riesgos al respecto:

1. Pasar por alto un requerimiento crucial
2. Representación inadecuada y esporádica del cliente
3. Modelar únicamente requerimientos funcionales
4. No inspeccionar los requerimientos
5. Esperar a tener requerimientos perfectos antes de comenzar la codificación
6. Representar los requerimientos en forma de diseño de soluciones.

Requirements Engineering as a success factor in software projects
(La Ing. de Requerimientos como factor de éxito en los proyectos de software)
Hubert F. Hofmann, Franz Lehner
IEEE Software. Julio-Agosto 2001.
http://www.computer.org

Contenido

Los requerimientos deficientes representan la causa más grande del fracaso de proyectos. Los autores realizan un completo estudio en el que identifican las prácticas y factores de la Ingeniería de Requerimientos (IR) que contribuyen claramente al éxito de los proyectos de software. Se enfocan a tres factores: conocimiento, recursos y proceso. Como resultado, obtienen las siguientes diez mejores prácticas:

1. Involucrar a los clientes y usuarios en la IR
2. Identificar y consultar todas las fuentes probables de requerimientos
3. Asignar a miembros del equipo capacitados en las actividades de la IR
4. Invertir del 15% al 30% de los esfuerzos totales a las actividades de la IR
5. Proporcionar formatos y guías de especificación con ejemplos
6. Mantener buenas relaciones entre los involucrados
7. Ponderar requerimientos
8. Desarrollar modelos complementarios y prototipos
9. Mantener una matriz de rastreo de requerimientos
10. Usar revisiones pares y ensayos para validar y verificar requerimientos.

De cada práctica se indica el costo de introducción, el costo de aplicación y los beneficios clave.

Software management seven deadly sins
(Los siete pecados capitales de la gestión de software)
Donald J. Reifer
IEEE Software. Marzo-Abril 2001.
http://www.computer.org

Contenido

El autor, con base en su experiencia y relación con directivos de diversas empresas, reflexiona sobre siete errores clave en la administración de software, los cuales son:

1. Requerimientos volátiles
2. Planeación insuficiente
3. Estimaciones de tiempo y costo no realistas
4. Controles inadecuados
5. Falta de capitalización
6. Considerar a la industria del software como diferente de las demás
7. Perder de vista la calidad.

Capturing the benefits of Requirements Engineering
(Capturando los beneficios de la Ingeniería de Requerimientos)
Pete Sawyer, Ian Sommerville y Stephen Viller
IEEE Software. Marzo-Abril 1999.
http://www.computer.org

Contenido

Los autores desarrollaron un método que permite el mejoramiento sistemático e incremental de la ingeniería de requerimientos: REGPG (Requirements Engineering Good Practice Guide).

El REGPG, como CMM, está basado en niveles de madurez (inicial, repetible y definido). Describe 66 prácticas que han sido abstraídas de estándares, reportes de prácticas de requerimientos y de la experiencia; las prácticas son clasificadas en básicas, intermedias y avanzadas de acuerdo a la especialización requerida. Como en el modelo SPICE, cada práctica es evaluada por criterios (estandarizado, normal, discrecional y nulo); además cada práctica considera el costo de introducción, el costo de aplicación y el beneficio clave. Los aspectos que considera el REGPG son: el documento de requerimientos; la obtención, el análisis y la negociación, la descripción, la validación y la administración de requerimientos; y el modelado del sistema.

El propósito del REGPG no es la certificación, sólo pretende consolidar las prácticas de la ingeniería de requerimientos más que dictar un estándar, y contribuir así con los programas de mejoramiento.

User involvement key to success
(Involucramiento del usuario, clave del éxito)
Carl Clavadetscher
IEEE Software. Marzo-Abril 1998.
http://www.computer.org

Contenido

En este breve artículo, el autor manifiesta la importancia de los usuarios en la Ingeniería de Requerimientos. Los usuarios tienen problemas para comprender y comunicar adecuadamente los requerimientos; el equipo de desarrollo y los analistas deben ayudarlos y colaborar con ellos ya que únicamente los usuarios conocen qué es lo que el sistema realmente debe hacer.

Blueprint for the ideal requirements engineer
(Prototipo del ingeniero de requerimientos ideal)
Mark Christensen y Carl Chang
IEEE Software. Marzo 1996.
http://www.computer.org

Contenido

Los autores señalan las principales características y actividades que debe considerar un ingeniero de requerimientos, como son:

- Capacidad para organizar los requerimientos
- Madurez para conocer cuándo ser general y cuando ser específico
- Mantener fluida la evolución de los requerimientos
- Ser un buen oyente para considerar opiniones tanto técnicas como no técnicas
- Como intermediario, debe ser un buen organizador de equipos y facilitador que llene los huecos de comunicación entre las partes
- La formación formal del ingeniero de requerimientos debe ser multidisciplinaria.

Fifteen principles of Software Engineering
15 principios de la Ingeniería de Software
Alan M. Davis
IEEE Software. Noviembre 1994.

http://www.computer.org

Contenido

1. Hacer de la calidad lo primero
2. El software de alta calidad es posible
3. Hacer entregas tempranas a los clientes
4. Determinar el problema antes de escribir los requerimientos
5. Evaluar alternativas de diseño
6. Usar un modelo de proceso apropiado
7. Usar diferentes lenguajes para diferentes fases
8. Minimizar la distancia intelectual: entre el problema del mundo real y la solución computarizada
9. Aplicar técnicas antes que herramientas
10. Hacer lo correcto antes de hacerlo rápido
11. Inspeccionar el código
12. La buena administración es más importante que la buena tecnología
13. La gente es clave en el éxito
14. De la moda en tecnología, técnicas y herramientas, lo que acomoda
15. Asumir la responsabilidad de lo que falla.

A field guide to effective requirements management under SEI's Capability Maturity Model
(Una guía para la administración efectiva de requerimientos bajo el Modelo de Madurez de Capacidades del Instituto de Ingeniería de Software)
Dean A. Leffingwell
Rational Software Corporation. 1996.
http://www.rational.com

Contenido

El Modelo de Madurez CMM, surgió como un estándar de la industria de software para tener un proceso de calidad del software, que fuera más comprensible y profundo que los estándares de ISO 9000. Proporciona una vista de las actividades del software que están siendo ampliamente aceptadas, siendo la Administración de Requerimientos una de las principales.

La Administración de Requerimientos es una actividad clave del proceso diseñada para asegurar que los productos de software satisfagan las necesidades de los usuarios tanto en calidad como en funcionalidad. El artículo resume esta actividad bajo del CMM y proporciona una breve guía para ello.

Habit's of effective analysts (Hábitos de los analistas efectivos)
Karl E. Wiegers
http://www.processimpact.com

Contenido

Los líderes de proyectos de software asumen que las capacidades de los programadores son adecuadas y suficientes para entrevistar a los usuarios y escribir especificaciones de requerimientos sin ninguna capacitación o asesoría. Esta no es una creencia razonable, ya que la ingeniería de requerimientos tiene su propio cuerpo de conocimiento y conjunto de habilidades. El rol de los analistas de requerimientos es crítico en el proyecto de software. Cada organización debería desarrollar un equipo de analistas capacitados.

En el artículo se describen varias características y prácticas de los analistas de requerimientos exitosos, tales como:

- Ser un intermediario entre el usuario y los desarrolladores
- Definir el alcance y las tareas del usuario necesarias para cumplir con el sistema
- Hacer preguntas reveladoras
- Priorizar a tiempo y frecuentemente
- Crear un ambiente de colaboración
- Pulir sus habilidades en el uso y aplicación de técnicas de comunicación, escritura, modelado, etc.

Requirements checklist (Listas de verificación de requerimientos)

http://www.construx.com

Contenido

El documento contiene una serie de preguntas para verificar los requerimientos, desde tres aspectos:

- Contenido
- Calidad
- Integridad.

What is a requirement (Qué es un requerimiento)

Richard Hardwell

http://www.incose.org

Contenido

El documento examina la naturaleza de un requerimiento desde el punto de vista del administrador, del analista y del usuario. Menciona también la importancia de que los requerimientos representen una necesidad (y no una solución directa), y a considerar aspectos de calidad.

2.9 Herramientas de Software

Rational Requisite Pro v. 2002

Versión de evaluación obtenida de: http://www.rational.com

Rational Unified Process v. 2002

Versión de evaluación obtenida de: http://www.rational.com

El software muestra todos los elementos del Proceso Unificado de Desarrollo de Rational, como un libro electrónico en un entorno web. Se describen las fases y las disciplinas, las actividades, los participantes y los artefactos utilizados y producidos en cada flujo de trabajo. La disciplina de requerimientos, se describe detalladamente, además de conceptos importantes de la Administración de Requerimientos. Proporciona un glosario y algunos proyectos de demostración del proceso.

Caliber RM

Demostración obtenida de: http://www.spc.ca

Herramienta para la administración de requerimientos.

2.10 Organizaciones

Asociación Mexicana para la Calidad en Ingeniería de Software (AMCIS)

La misión de la AMCIS es ser un foro que contribuya al aumento de la productividad y la calidad en los procesos de software en las organizaciones o instancias responsables de desarrollo de software en México.

Itera

Representante de *Rational Software Corporation* en México, cuyas herramientas permiten controlar todas las actividades del ciclo de vida de desarrollo de software, entre ellas, la Administración de Requerimientos. Además imparten cursos especializados y organizan eventos para la difusión de las mejores prácticas en el desarrollo de software.

Fundación Arturo Rosenblueth

Imparte maestrías, diplomados, seminarios, talleres y cursos especializados en las áreas de cómputo, desarrollo de software, sistemas de información y telecomunicaciones.

3 MARCO CONCEPTUAL

3.1 Fundamentos de la Ingeniería de Requerimientos

3.1.1 Problemática relacionada con los requerimientos

Son innumerables, las opiniones de diversos autores sobre los problemas relacionados con los requerimientos en el desarrollo de software:

> *"Una de las razones más comunes por las que un sistema falla es una mala definición de los requerimientos."* [Laura Scharer]

> *"El escribir especificaciones de sistema y de software buenas, correctas, completas y medibles es uno de los problemas más comunes en el desarrollo de software."* [IEEE]

> *"Para el éxito de un desarrollo de software es esencial una comprensión total de los requisitos del software. No importa lo bien diseñado o codificado que esté un programa si no se ha analizado correctamente, pues defraudará al usuario y frustrará al desarrollador....es verdad que los requisitos del software cambian, pero el impacto del cambio varía según el momento en que se introduzca. Si se pone cuidado al dar la definición inicial, los cambios solicitados al principio pueden acomodarse fácilmente. Cuando se solicitan al final de un proyecto, los cambios pueden producir un orden de magnitud más caro que el mismo cambio pedido al principio."* [Pressman]

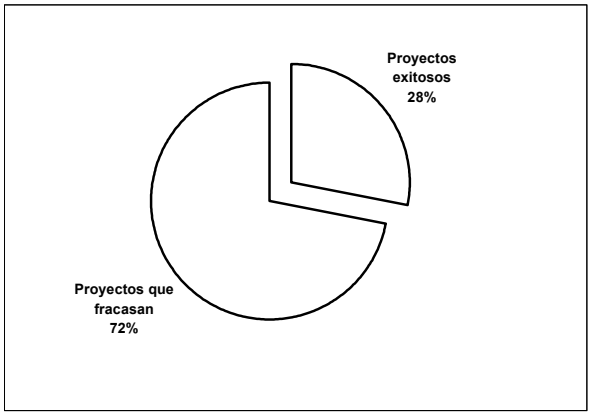
> *"Cuando surgió la Crisis del Software en los años 60's, hubo muchos esfuerzos por encontrar las causas del -ahora familiar- síndrome del problema. Las investigaciones determinaron que las deficiencias en los requerimientos son una de las causas más importantes del problema… en la mayoría de los proyectos de software que no se cumplen los objetivos de realización y costo, los requerimientos inadecuados juegan un papel principal y costoso en el fracaso del proyecto … el desarrollo de la especificación de requerimientos en muchos casos parece trivial pero probablemente es la parte del proceso que conlleva a mayores fracasos que cualquier otra."* [IEEE]

"Un inadecuado o insuficiente análisis de requerimientos da inicio a un proyecto con el pie izquierdo." [Software Development Magazine]

"El problema de la ingeniería de requerimientos se incrementa exponencialmente con el tamaño del sistema." [Ian Sommerville]

"El costo de agregar o corregir requerimientos una vez entregado un sistema es aproximadamente 100 veces más que hacerlo durante la definición inicial del sistema." [Gruia-Catalin]

Existen además diversos estudios que evidencian la problemática con requerimientos. Por ejemplo, las estadísticas de la industria del software según el Standish Group, CHAOS, 2001 revelan lo siguiente:



Se entiende por proyecto exitoso aquel que se termina con los recursos razonablemente estimados y que satisface las expectativas de los clientes y usuarios. Los proyectos que fracasan son por ejemplo, aquellos que se retrasan, se cancelan, sobrepasan por mucho los recursos inicialmente estimados para su desarrollo, nunca se usan o no satisfacen las expectativas de los clientes y usuarios.

Además, otros números interesantes a considerar son los siguientes:

- El costo real de un proyecto casi se triplica con respecto a lo estimado, es decir, el promedio del costo sobrepasado es de 189%.

- El tiempo de retraso promedio es de 222%, es decir, los proyectos se retrasan por más del doble de lo estimado.
- El promedio de fallas según las expectativas iniciales es de 61%. Más de la mitad de las fallas tienen que ver con las perspectivas iniciales del proyecto.

Asimismo, el estudio demuestra que los factores que conllevan a errores relacionados con los requerimientos son principalmente:

- Escasa participación del usuario en el proyecto.
- Objetivos no claros desde un principio.
- Requerimientos y especificaciones incompletas.
- Cambios en los requerimientos y especificaciones.
- Falta de planeación y gestión del riesgo.

Por otro lado cabe mencionar que la identificación de requerimientos no es fácil, debido a que el sistema por desarrollar debe satisfacer las necesidades reales de los usuarios, las cuales muchas veces no son claras, provienen de diversas fuentes y son de diversos tipos, por lo que se requieren habilidades y herramientas tanto técnicas como administrativas por parte de los analistas. Esta es la parte medular para que el sistema brinde la funcionalidad correcta.

La principal relación de los requerimientos con el retraso de los proyectos radica en que algunos aspectos están fuera del alcance original, el cual nunca fue documentado, por lo que no hay un mecanismo formal para justificarlo. Además esto da lugar a malos entendidos entre los clientes y el equipo de desarrollo.

Con respecto al número de requerimientos, entre más grande sea el proyecto, puede ser difícil controlarlos, sobre todo si no se tienen herramientas y mecanismos establecidos, considerando además que se relacionan unos con otros y se relacionan con otros artefactos (planes, código, pruebas) del proceso de ingeniería de software.

Es un hecho además que es muy caro hacer cambios a requerimientos después de que han sido acordados, ya que conllevan a costos relacionados con la re-especificación, el re-diseño, la re-codificación, el re-probar, la re-implantación, las acciones correctivas, la responsabilidad civil, los costos de desplazamiento, la documentación y el mantenimiento constante.

3.1.2 Conceptos generales de la Ingeniería de Requerimientos

3.1.2.1 Requerimiento

Un **requerimiento** es una característica identificable expresada en términos de funcionalidad o desempeño que un sistema debe poseer para lograr su objetivo. A continuación se presentan diversas definiciones y enfoques de lo que es un requerimiento:

> *Si algo dice que algo debe ser cumplido, transformado, producido o proveído, es un requerimiento.* [Richard Hardwell]

> *Un requerimiento es una condición o capacidad que debe satisfacer un sistema.* [Rational Software]

De acuerdo con la IEEE, entendemos por **sistema** a "un conjunto de hardware, software, datos, personas, facilidades y procedimientos organizados para lograr algún objetivo en común", y un **requerimiento de sistema** como:

> *Una característica del sistema necesitada por un usuario para resolver un problema o lograr un objetivo.*

> *Una característica que debe contener un sistema o un componente del sistema para satisfacer un contrato, estándar, especificación u otro instrumento impuesto formalmente.*

Por lo tanto un requerimiento de software es:

> *Una capacidad del software necesitada por un usuario para resolver un problema o lograr un objetivo.*

> *Una capacidad del software que debe ser cumplida o contenida por un sistema o un componente del sistema para satisfacer un contrato, estándar, especificación u otro instrumento formalmente impuesto.*

3.1.2.2 Ingeniería de Requerimientos

El Instituto de Ingenieros en Electricidad y Electrónica (IEEE), definen a la Ingeniería de Requerimientos como:

> *la ciencia y disciplina relacionada con el análisis y documentación de requerimientos, incluyendo el análisis de necesidades, el análisis de requerimientos y la especificación de requerimientos. También proporciona los mecanismos adecuados para facilitar las actividades de análisis, documentación y verificación de estos.*

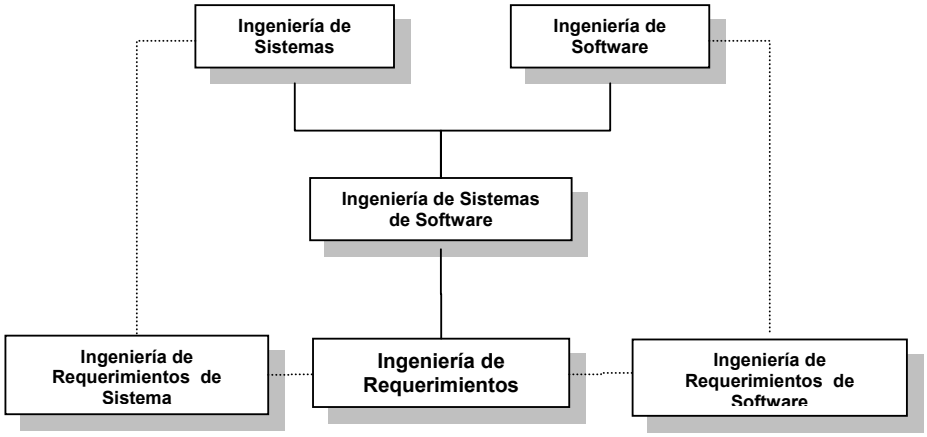
Para Ian Sommerville y Pete Sawyer Ingeniería de Requerimientos es:

> *el término que engloba todas las actividades relacionadas con descubrir, documentar y mantener un conjunto de requerimientos para un sistema informático. El término "ingeniería" implica el uso sistemático y repetible de técnicas que son usadas para asegurar que los requerimientos del sistema sean completos, consistentes y relevantes.*

Recientemente a la Ingeniería de Requerimientos se le denomina también Administración de Requerimientos y se interpretan de forma similar; no obstante de manera estricta la Administración de Requerimientos es complemento de la Ingeniería de Requerimientos. La ingeniería es la construcción de éstos; la administración es la organización y gestión.

3.1.3 Relación de la Ingeniería de Requerimientos con otras disciplinas

La Ingeniería de Requerimientos puede enfocarse a sistemas o a software. No obstante, en los sistemas informáticos la relación sistema-software es muy estrecha, es por ello que este trabajo cubre de cierta manera ambas perspectivas. En este punto se ubicará a la Ingeniería de Requerimientos -desde nuestra perspectiva- con respecto a otras disciplinas reconocidas formalmente por la IEEE.



Ingeniería de Sistemas *es la aplicación de esfuerzos científicos y de ingeniería para 1) transformar una necesidad operacional en una descripción de los parámetros de desempeño del sistema y una configuración del sistema mediante el uso de un proceso iterativo de definición, síntesis, análisis, diseño, prueba y evaluación; 2) integrar los parámetros técnicos relacionados y asegurar compatibilidad de todas las interfaces funcionales y del programa de una manera que optimice la definición y el diseño del sistema; 3) integrar fiabilidad, facilidad de mantenimiento, seguridad, facilidad de perdurar, aspectos humanos y otros factores dentro del esfuerzo total de ingeniería para lograr los objetivos de costo, tiempo y de realización técnica.*

La ingeniería de sistemas es un proceso iterativo que usa un enfoque sistemático para asegurar que el sistema, cuando sea desarrollado, satisfaga objetivos específicos. Debe considerar los elementos de desempeño, costo, riesgo, tiempo y complejidad operacional para asegurar que se logró un diseño eficaz.

La ***Ingeniería de Software*** es la aplicación de un enfoque sistemático, disciplinado y cuantificable hacia el desarrollo, operación y mantenimiento del software; es decir la aplicación de la ingeniería al software.

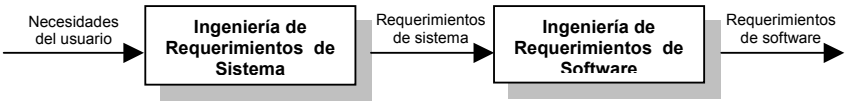
Ingeniería de Sistemas de Software *es tanto un proceso técnico como administrativo. El proceso técnico es el esfuerzo analítico necesario para transformar una necesidad operacional en un diseño de software del tamaño y configuración apropiados, y su documentación en especificaciones de diseño y requerimientos. El proceso administrativo implica evaluar el riesgo y el costo, integrar a los ingenieros*

especialistas y a los grupos de diseño, mantener el control de la configuración y auditar continuamente el esfuerzo para asegurar que los objetivos de costo, de tiempo y de realización técnica son satisfechos para cubrir la necesidad operacional original.

*La **Ingeniería de Requerimientos de Sistema** es la ciencia y disciplina relacionada con el análisis y documentación de los requerimientos del sistema. Comprende la transformación de una necesidad operacional en una descripción del sistema, los parámetros de desempeño del sistema y la configuración del sistema mediante el uso de un proceso iterativo de análisis, diseño, y prototipos.*

*La **Ingeniería de Requerimientos de Software** es la ciencia y disciplina relacionada con el análisis y documentación de los requerimientos de software. Implica la división de los requerimientos de sistema en varios subsistemas y tareas y la asignación de estos subsistemas y tareas al software, así como su transformación en una descripción de requerimientos de software y parámetros de desempeño mediante el uso de un proceso iterativo de análisis, diseño y prototipos.*

La principal diferencia entre Ingeniería de Requerimientos de Sistema y la Ingeniería de Requerimientos de Software es que el origen de los requerimientos de sistema radica en las necesidades del usuario mientras que el origen de los requerimientos de software en los requerimientos y/o especificaciones del sistema.



Por lo tanto los ingenieros de requerimientos –o analistas- del sistema trabajan con los clientes y usuarios, identificando sus necesidades, planes y recursos disponibles. Deben producir documentos que sean comprensibles por todos (analistas, gerentes, arquitectos de sistema, usuarios y clientes).

Los ingenieros de requerimientos de software trabajan con los documentos de requerimientos de sistema y los traducen en requerimientos de software que deben ser comprendidos tanto por los diseñadores de software como por los ingenieros de requerimientos.

3.1.4 Actividades de la Ingeniería de Requerimientos

Dentro de la Ingeniería de Requerimientos se identifican una serie de actividades distintas pero complementarias entre sí, las cuales van desde la identificación misma de los requerimientos hasta su gestión una vez que se encuentran debidamente especificados. Estas actividades no se dan de una forma secuencial, sino que se realizan continuamente a lo largo de la actividad de requerimientos en el ciclo de desarrollo. Se presentan en el siguiente cuadro:

La ***obtención*** es la identificación y adquisición de los requerimientos y restricciones de los involucrados en el sistema, del dominio de la aplicación y de los ambientes operacionales del sistema y organizacional.

El ***análisis y negociación*** incluye la clasificación, ponderación y análisis de los requerimientos identificados, la definición del alcance de la solución, así como la resolución de asuntos que surgen de la obtención de requerimientos.

El ***modelado y especificación*** consisten en la representación y documentación de los requerimientos utilizando diversas técnicas, herramientas y metodologías.

La ***validación y verificación*** consiste en establecer un proceso para comprobar que los requerimientos sean correctos, completos, consistentes y claros, así como asegurar que cumplen con los estándares de calidad.

La ***administración*** se enfoca al control y organización de la información, documentos y productos relacionados con los requerimientos a través de todo el ciclo de vida de desarrollo. La adecuada administración de requerimientos permite encontrar, documentar, organizar y rastrear los cambios en los requerimientos de un sistema.

3.1.5 Clasificación de los Requerimientos

3.1.5.1 Niveles de requerimientos

De acuerdo a la perspectiva, se identifican principalmente tres niveles de requerimientos que se expresan en la siguiente pirámide:

El dominio del problema representa el ambiente de los usuarios reales y otros involucrados. Los usuarios tienen problemas y necesidades en su ámbito de negocio, los cuales como informáticos, debemos comprender, además de entender su cultura y su lenguaje para construir el sistema que satisfaga sus necesidades. Dentro del dominio del problema, comprendemos el problema a resolver. Por otro lado, en el espacio o dominio de la solución, nos enfocamos a definir y modelar una solución para el problema del usuario.

Los requerimientos ***de negocio*** representan las necesidades de los usuarios y otros involucrados que están siendo afectados por el problema y/o serán afectados por la solución. Un requerimiento de negocio (o necesidad) es una reflexión del negocio, personal, un problema operacional u oportunidad que debe ser destinada en orden para justificar el desarrollo, compra o uso de un nuevo sistema.

Es un hecho que el equipo de desarrollo construirá un mejor sistema solamente si comprende las necesidades reales de los involucrados. Estas necesidades de usuario proveen una pieza crucial del rompecabezas.

Los requerimientos ***de sistema*** son descripciones simples, en el lenguaje del usuario que se utilizan para comunicar la solución de alto nivel a los involucrados. Se denominan características, es decir servicios que el sistema proporciona para cumplir una o más necesidades de los involucrados.

Una ***característica*** es un servicio que el sistema proporciona para satisfacer una o más necesidades de los involucrados. Las características son fácilmente expresables en lenguaje natural mediante frases cortas. Son expresiones de alto nivel sobre el comportamiento deseado del sistema; las características de un producto o sistema.

Las necesidades y las características están estrechamente relacionadas. Si el equipo no comprende o conoce lo que hay detrás de la característica, existe un gran riesgo; tal vez la característica no resuelve una necesidad real y entonces el sistema no cumplirá los objetivos y expectativas de los usuarios.

Por su parte los requerimientos ***de software*** describen de manera más específica la solución. Canalizan una o más características requeridas por los usuarios en soluciones de software específicas.

3.1.5.2 Tipos de Requerimientos

La mayoría autores coinciden en clasificar los requerimientos en funcionales y no funcionales.

- Los ***requerimientos funcionales*** describen lo que el sistema debe hacer, es decir especifican acciones que el sistema debe ser capaz de realizar, sin considerar restricciones físicas. Los requerimientos funcionales especifican el comportamiento del sistema.
- Los ***requerimientos no funcionales*** describen únicamente atributos del sistema o atributos del ambiente del sistema y pueden ser por ejemplo: requerimientos de interfaz, de diseño, de implementación, legales, físicos, de costo, de tiempo, de calidad, de seguridad, de construcción, de operación, entre otros.

Una subclasificación más detallada de los requerimientos funcionales y no funcionales que presenta el Proceso Unificado de Desarrollo (RUP), se muestra a continuación:

Tipo		Subtipo	Descripción
Funcionales	Funcionales	Funcionalidad	Los requerimientos funcionales incluyen características y capacidades del sistema o del software.
No Funcionales	De Calidad	Uso	Los requerimientos de facilidad de uso pueden incluir categorías relacionadas con: factores humanos, estéticos, consistencia en las interfaces de usuario, ayuda en línea, asistentes, documentación de usuario, materiales de capacitación.
		Confiabilidad	Algunos requerimientos de fiabilidad a considerar son: frecuencia y severidad de fallas, facilidad de recuperación, previsión, exactitud, tiempo entre fallas.
		Desempeño	Un requerimiento de desempeño impone condiciones en requerimientos funcionales. Por ejemplo, para una acción dada, se pueden especificar parámetros de desempeño de: velocidad, eficiencia, disponibilidad, exactitud, tiempo de respuesta, tiempo de recuperación, recursos utilizados.
		Mantenimiento y Soporte	Los requerimientos de facilidad de mantenimiento pueden incluir: facilidad de probar, posibilidad de crecimiento, adaptabilidad, facilidad de mantenimiento, compatibilidad, facilidad de configuración, facilidad de servicio, facilidad de instalación, internacionalización y facilidad de adaptación y personalización.
	De Restricción	Diseño	Un requerimiento de diseño (algunas veces llamado restricción de diseño) especifica o limita el diseño de un sistema. Incluye aspectos relacionados con: sistema operativo, ambiente, compatibilidad, aplicación de estándares, integración de sistemas.
		Implementación	Un requerimiento de implementación especifica o limita la codificación o construcción de un sistema. Por ejemplo: estándares requeridos, lenguajes de implementación, políticas para la integridad de la base de datos, limitación de recursos, ambientes de operación.
		De Interfaz	Un requerimiento de interfaz especifica: un elemento externo con el cual el sistema debe interactuar; restricciones en formatos, regulación de tiempos u otros factores usados en dicha interacción.
		Físicos	Un requerimiento físico especifica una característica física que el sistema posee. Por ejemplo: material, forma, tamaño, peso. Este tipo de requerimiento puede ser usado para representar requerimientos de hardware así como las configuraciones físicas de la red.

De acuerdo al ***nivel de cumplimiento***, los requerimientos pueden ser:

- ***Obligatorio.*** Requerimientos que debe ser implementado.
- ***Recomendable.*** Es deseable que sea implementado.
- ***Opcional.*** No es crítica su implementación.

Otra clasificación de los requerimientos, de acuerdo al valor que proporcionan al cliente o usuario, señalada por *Pressman* es la siguiente:

- ***Normales.*** Se declaran como objetivos y metas para un producto o sistemas durante las reuniones con el cliente. Si estos requisitos están presentes, el cliente quedará satisfecho.
- ***Esperados.*** Estos requisitos son implícitos al producto y pueden ser tan fundamentales que el cliente no los declara explícitamente. Su ausencia sería motivo de una insatisfacción significativa.
- ***Innovadores.*** Estas características van más allá de las expectativas del cliente y suelen ser muy satisfactorias.

De acuerdo al origen y aplicación contractual de un requerimiento, *Hardwell* los clasifica en:

- ***Primarios.*** Estos requerimientos están comprometidos en un contrato o acuerdo y provienen de una fuente externa al equipo de desarrollo.
- ***Derivados.*** Los requerimientos que son generados a partir de los requerimientos primarios y no están explícitamente especificados.

3.2 Actividades de Ingeniería de Requerimientos

La Ingeniería de Requerimientos como se mencionó anteriormente, consta de 5 actividades: obtención; análisis y negociación; modelado y especificación; validación y verificación; y administración. A continuación se detalla cada una de éstas, con las técnicas y herramientas relacionadas, así como sus participantes.

3.2.1 Participantes

Los involucrados *(stakeholders)* son los individuos y organizaciones que están relacionados activamente en un proyecto de software, tienen influencia directa o indirecta sobre los requerimientos, o sus intereses se ven afectados por el proyecto. Pueden incluir clientes, usuarios finales, directivos, administradores de proyecto, analistas, programadores, y personal de aseguramiento de la calidad.

No existe una terminología única de los distintos roles de las personas involucradas en el desarrollo de software [♦]. A continuación se enlistan los más generales con sus términos similares, aunque cabe resaltar que existen leves diferencias entre ellos.

- Líder de Proyecto/Administrador de Proyecto/Gerente de Proyecto
- Analista/Ingeniero de Requerimientos
- Ingeniero de Sistemas/Arquitecto
- Programador/Desarrollador/Ingeniero de Software
- Probador/Asegurador de la Calidad
- Administrador de Bases de Datos

Los principales roles involucrados en el proceso de ingeniería de requerimientos, así como las actividades en las que tienen mayor participación se presentan en el siguiente cuadro:

♦ RUP propone un conjunto de roles muy completo y detallado para todo el ciclo de desarrollo de software.

Rol	Descripción	Obtención	Análisis y negociación	Modelado y especificación	Validación y verificación	Administración
Cliente	Representa a la persona u organización que solicita la creación de un sistema a un área de desarrollo y quien lo paga. Es con quien se negocia el tiempo, costo y alcance del proyecto. Pueden o no ser usuarios del sistema.	◆	◆		◆	
Usuario	Son las personas que interactuarán con el sistema. Proporcionan información fundamental para el éxito del proyecto, ya que conocen y conviven con los procesos diarios.	◆	◆		◆	
Líder de proyecto	Por parte del equipo de desarrollo, es el representante ante el cliente. Es la persona responsable de completar el proyecto exitosamente con los recursos dados.	◆	◆		◆	◆
Analista	Su labor se enfoca a la ingeniería de requerimientos, los identifica, analiza, modela y documenta. Establece contacto directo con los usuarios y utiliza diversas técnicas de comunicación y de recopilación de información para lograr su objetivo.	◆	◆	◆	◆	◆
Programador	Con base en los requerimientos recibidos de los ingenieros de requerimientos, el programador realiza la codificación para producir el sistema deseado.				◆	◆
Asegurador de la Calidad	Garantiza el cumplimiento del proceso y de los estándares del producto. Enfocado a los requerimientos los verifica y valida para imprimir la calidad desde las primeras etapas del desarrollo. Paralelamente prepara planes de prueba para esos requerimientos del sistema.				◆	◆

3.2.1.1 Rol de los clientes y usuarios

El *usuario* es el experto en sus necesidades y sus procesos de negocio. Es quien debe comunicar estas necesidades claramente a los desarrolladores de software. Está involucrado directamente con los procesos de la organización y es quien finalmente operará el sistema.

Los clientes son quienes financian el proyecto y los que recibirán directa o indirectamente los beneficios de éste. Ellos juegan un papel muy importante en la toma de decisiones y en la negociación del proyecto.

Para promover una adecuada participación de los usuarios, es necesario tener en cuenta lo siguiente:

- Los usuarios están capacitados en su área de negocio pero no necesariamente en sistemas, cómputo e informática.
- El usuario pretende facilitar sus tareas, pero también proteger sus intereses.
- Los usuarios requieren atención periódica. Después de que se obtuvieron todos los requerimientos de su parte, es importante informarles del avance, presentarles entregables y mantenerlos en contacto.
- A los usuarios básicamente les interesa conocer las generalidades del sistema en términos de los beneficios que van a obtener, sin tecnicismos ni complicaciones.
- La flexibilidad del sistema y la facilidad de uso son consideraciones clave para los usuarios.
- Algunas veces lo usuarios solicitan 10 características y realmente necesitan 1. El total de los requerimientos recibidos del usuario debe ser reducido a los que sean necesarios, suficientes, controlables y viables de realizar.
- A veces no es posible tener contacto con toda la comunidad de usuarios, por lo que se debe tener un representante que presente opiniones consensuadas al equipo de desarrollo y que tenga autoridad suficiente para tomar decisiones relacionadas con el proyecto.

Karl Wiegers, en su artículo "Derechos y responsabilidades de los Usuarios" presenta una entretenida lista de aspectos a considerar con los usuarios:

Derechos de los usuarios:

- Que los analistas hablen su mismo lenguaje.
- Que los analistas aprendan sobre los objetivos del negocio y del sistema.
- Que los analistas traduzcan las necesidades presentadas en especificaciones de software.
- Contar con desarrolladores que expongan los productos de trabajo de requerimientos.
- Tratar con desarrolladores que mantengan el respeto y una actitud profesional.
- Contar con analistas que presenten ideas y alternativas tanto en los requerimientos como en la implementación.
- Describir características que harán el producto fácil de usar.
- Ser informado de alternativas y oportunidades para adaptar los requerimientos y permitir la reutilización de modelos ya existentes.
- Recibir adecuadas estimaciones de costo y acuerdos cuando se requieran cambios en los requerimientos.
- Recibir un sistema que contenga la funcionalidad y calidad esperadas.

Responsabilidades de los usuarios:

- Instruir a los analistas sobre el negocio y definir el vocabulario de su área.
- Invertir tiempo en proporcionar requerimientos y aclarar dudas.
- Ser específico y preciso sobre las necesidades del negocio y los requerimientos del sistema.
- Tomar decisiones a tiempo sobre los requerimientos.
- Respetar las estimaciones de costo y viabilidad del equipo de desarrollo.
- Fijar prioridades para requerimientos individuales, características del sistema o casos de uso.
- Revisar los documentos de requerimientos y los prototipos.
- Comunicar prontamente los cambios a los requerimientos del producto.
- Seguir el proceso de cambio de los requerimientos.
- Respetar el proceso de ingeniería de requerimientos que usan los desarrolladores.

3.2.1.2 Rol de los analistas

El equipo de desarrollo, y en específico los analistas (ingenieros de requerimientos) deben comprender los problemas del usuario, su cultura y su lenguaje, así como presentar propuestas de solución comprensibles a los usuarios que cubran sus necesidades y expectativas.

Karl E. Wiegers en su artículo "Hábitos de los analistas efectivos, sugiere las siguientes tareas del analista:

- Mejorar la comunicación entre los clientes y el equipo de desarrollo. El analista es un hombre intermediario en la comunicación. Debe primero comprender las necesidades actuales del usuario para luego definir un conjunto de requerimientos funcionales y objetivos de calidad que permitirán a los diseñadores, programadores y probadores, construir y verificar el sistema.
- Aprender y usar el vocabulario del dominio del problema, más que forzar a los clientes a comprender la jerga informática. Incluir los términos del negocio en un glosario, el cual deber ser parte de la documentación de requerimientos.
- Escuchar y comprender a los usuarios para orientar adecuadamente sus puntos de vista en el producto. Tomar en cuenta las expectativas implícitas y características como: desempeño, facilidad de uso, eficiencia y fiabilidad.
- Escribir requerimientos de alta calidad en documentos bien organizados. Que puedan ser revisados por otros analistas, representantes de los usuarios, desarrolladores y probadores.
- Hacer preguntas significativas, que permitan obtener información clave para el sistema.
- Priorizar a tiempo y frecuentemente los requerimientos. Tratar de construir primeramente las características más críticas. Aunque para los usuarios todo tiene la máxima prioridad, el analista debe negociarlo e identificar las características realmente necesarias y las características urgentes.
- Crear un ambiente colaborativo, ya que finalmente los usuarios van a tener lo que necesitan; la organización que desarrolla va a ofrecer un producto adecuado y los desarrolladores van a construir un buen sistema. La participación del usuario es un factor clave.

- Aplicar eficazmente herramientas y buenas prácticas de ingeniería de requerimientos.
- Capacitarse y pulir sus habilidades en el uso y aplicación de técnicas de comunicación, escritura, modelado, entre otras. Un analista competente debe combinar habilidades de comunicación y relaciones interpersonales. Por ejemplo, técnicas para liderear mesas de trabajo, para entrevistar y platicar con individuos y grupos; habilidades de escuchar y escribir; habilidades interpersonales para negociar prioridades y resolver conflictos entre los involucrados en el proyecto; tener credibilidad y conversar efectivamente, así como habilidades de modelado.

3.2.1.3 Rol del líder de proyecto

El Líder de Proyecto es el representante ante el cliente, coordina al equipo de trabajo y tiene la responsabilidad de terminar el proyecto satisfactoriamente con los recursos asignados.

Algunas de las principales tareas del Líder de Proyecto son:

- Coordinar al equipo de trabajo.
- Estimar el tiempo, el costo y los recursos estimados para realizar el proyecto. Planear y dar seguimiento a dichos planes.
- Conducir el proceso de obtención de requerimientos.
- Manejar y resolver los conflictos de los involucrados.
- Lograr la definición del conjunto de características que proporcionarán el más alto valor a la mayor cantidad de involucrados.
- Definir la visión del producto.
- Negociar con los directivos, usuarios y desarrolladores.
- Mantener una estrecha relación entre lo que el cliente desea y lo que el equipo de desarrollo puede entregar en un tiempo determinado.
- Ser el canal oficial entre el cliente/usuario y el equipo de desarrollo.
- Comunicar las características de las versiones a los involucrados.
- Revisar las especificaciones de software para asegurarse que cumplan con la visión del producto.
- Administrar los cambios de prioridad en requerimientos, así como la adición y supresión de características.

3.2.2 Obtención

La obtención de requerimientos es la consecución de los requerimientos y restricciones de los involucrados en el sistema, del dominio de la aplicación, del ambiente operacional del sistema y de la organización.

En el proceso de obtención se identifican y comprenden las necesidades, problemas y restricciones de los distintos tipos de usuarios, mediante la comunicación con los clientes, usuarios del sistema y otros involucrados en su desarrollo. Para ello es importante tener conocimiento del problema, del dominio de la aplicación y del negocio.

3.2.2.1 Fuentes de requerimientos

Los requerimientos del sistema son obtenidos de diversas orígenes, por ejemplo de la consulta con los involucrados, de documentos relacionados, del conocimiento del dominio y otras como son:

- Estudios de mercado sobre las necesidades de informatización de la organizaciones similares.
- Comentarios de los usuarios sobre los sistemas actuales, los procesos de negocio o la problemática de la organización.
- Especificaciones generales preparadas por los usuarios con las necesidades básicas del área.
- Documentos de organización y procedimientos.
- Observaciones, salidas y medidas de los sistemas actuales.
- Entrevistas con los clientes y usuarios.
- Documentación de los sistemas actuales.
- Modelos y prototipos.
- Información sobre otros productos de software en el mercado que cubran necesidades similares.

3.2.2.2 Problemática en la obtención de requerimientos

La obtención de requerimientos, no es un fácil. Ian Sommerville, por ejemplo menciona los siguientes problemas:

- La mayoría de las veces, los involucrados no saben lo que realmente quieren del sistema, les es difícil articularlo o hacen peticiones no viables por el costo que ello implica.
- Los clientes expresan los requerimientos en sus propios términos, que nos son comprendidos por el equipo de desarrollo.
- Diferentes involucrados tienen diferentes requerimientos y son expresados de diferente manera.
- Aspectos organizacionales y políticos influyen en los requerimientos del sistema. Estos factores muchas veces no son obvios.
- El ambiente económico y de negocio del sistema es muy dinámico. Hay cambios durante la obtención de requerimientos.

Además agregaríamos otros aspectos como:

- Algunas veces los clientes y usuarios no están consciente de la importancia de esta actividad y no le dedican el tiempo necesario o no proporcionan información completa y veraz.
- Las organizaciones no tienen documentados sus procesos, no están actualizados estos documentos o en la realidad no se llevan a cabo como se especifica.
- Los usuarios desean ver resultados rápidos en términos de código.
- Internamente en su organización, los usuarios no se ponen de acuerdo en sus necesidades, políticas y estrategias para la incorporación de un sistema.
- Los usuarios desean la mayor cantidad de características del sistema en tiempo récord y con bajo costo.

No obstante existen técnicas y herramientas que nos ayudan a lograr una identificación y obtención de requerimientos exitosas; algunas de éstas se mencionan más adelante.

3.2.2.3 Análisis de problema

Para la adecuada identificación del los requerimientos es necesario una realizar una actividad clave: analizar el problema.

El análisis del problema es el proceso de comprender los problemas del mundo real y las necesidades del usuario para posteriormente proponer soluciones que cubran estas necesidades. El objetivo es ganar una mejor comprensión, antes de empezar el desarrollo del problema a resolver.

Esta actividad implica comprender el "dominio del problema" para pasar de éste al "dominio de la solución". Dean Leffingwell y Don Widrig sugieren los siguientes pasos del análisis del problema en el desarrollo de software:

1) Definir del problema. Por escrito, describir el problema, a quién afecta, en qué impacta y ver si todos los involucrados están de acuerdo en que ese es el verdadero problema.
2) Descubrir las causas raíces, "el problema detrás del problema". Muchas veces se tienen síntomas pero no son los problemas reales.
3) Identificar a los involucrados, es decir cualquier persona que puede ser afectada materialmente por la implementación de un nuevo sistema o aplicación, especialmente a los usuarios.
4) Definir los límites entre la solución y el mundo real. Esto nos ayudará a delimitar el alcance y a precisar las fronteras del sistema.
5) Identificar las restricciones de la solución. Estas restricciones son muy importantes ya que orientan o dictan la solución, y puede ser de tipo: económico, político, técnico, ambiental, de recursos, entre otros. Además, posteriormente se convertirán en requerimientos no funcionales del sistema.

Es importante mencionar que no todos los sistemas y aplicaciones son desarrollados para resolver un problema, algunas veces se construyen para tomar ventaja de oportunidades que se presentan en el mercado, para mantenerse actualizados tecnológicamente o para lograr una mayor competitividad.

O bien, también hay que considerar que muchas veces la solución del problema no es un sistema nuevo, sino más bien una revisión de los procesos de negocio, capacitación de las personas o cambios en la cultura organizacional.

3.2.2.4 Técnicas para la obtención de requerimientos

Las principales **técnicas** utilizadas en la obtención de requerimientos son:

- Entrevistas y cuestionarios.
- Talleres y mesas de trabajo.
- Lluvia de ideas.
- Presentaciones.
- Juego de roles.
- Prototipos.

A continuación se describen cada una de ellas. Los prototipos se usan tanto para obtener nuevos requerimientos como para especificarlos; éstos se describen más detalladamente en la sección de Modelado y Especificación (3.2.4).

3.2.2.4.1 Entrevistas

La entrevista es una técnica de recopilación de información que consiste en una conversación entre dos personas generalmente con un objetivo común, en la que el entrevistador formula una serie de preguntas al entrevistado. En la Ingeniería de Requerimientos, el entrevistador es el analista, el líder de proyecto o algún miembro del equipo del desarrollo; los entrevistados son los clientes, usuarios u otros involucrados con el sistema.

Para coadyuvar a que una entrevista sea eficaz, se pueden considerar las siguientes recomendaciones:

- Indagar el contexto de los involucrados y la organización. Recopilar información general sobre la organización, su misión, sus objetivos y su estructura.
- Definir el objetivo de la entrevista, preparar y revisar las preguntas antes de su realización.
- Ir de lo general a lo particular.
- Tratar de que la entrevista no sea tediosa, tardada o monótona para evitar aburrir al entrevistado.
- Una vez que se han obtenido las respuestas del entrevistados, verificarlas.

- Anotar las respuestas para tener referencia posterior.
- Mostrarse sensibles y empáticos con los entrevistados.
- En una entrevista de obtención de requerimientos, evitar en la medida de los posible brindar soluciones anticipadas.

3.2.2.4.2 Cuestionario

El cuestionario no es un sustituto de la entrevista, ya que el contacto directo con los clientes y usuarios y la observación del ambiente en donde operará el sistema son elementos muy importantes para comprender el problema y brindar una solución adecuada.

Sin embargo, son una técnica útil, que consiste en plantear por escrito una serie de preguntas, son distribuidos entre los diversos involucrados, y se utilizan para validar suposiciones, obtener datos estadísticos y obtener información cuando es difícil coincidir con los encuestados.

Puede ser aplicado previo a la entrevista, para que los usuarios ubiquen el tipo de información requerida y reflexionen sus respuestas. También puede ser aplicado después de entrevistas iniciales y de la actividad del análisis para corroborar o detallar información.

3.2.2.4.3 Talleres y mesas de trabajo

Los talleres de requerimientos son diseñados para lograr consenso sobre los requerimientos de la aplicación y para ganar un rápido acuerdo sobre el curso de una acción, en un corto tiempo. Con esta técnica, los involucrados clave del proyecto son reunidos por un período intensivo, corto y por no más de 2 días. El taller de preferencia es coordinado por un externo con experiencia en el proceso de administración de requerimientos; cuando esto no es posible, puede ser un miembro del equipo que tenga capacitación en el proceso.

La lluvia de ideas es la parte más importante de los talleres, ya que crea una atmósfera creativa y positiva en la que participan todos los involucrados. Algunos beneficios de los talleres son:

- Se tienen reunidos a los principales involucrados en la definición de requerimientos y objetivos del sistema, tanto de la organización (cliente

y usuarios) como del equipo de desarrollo (analistas/ingenieros de requerimientos).

- Contribuye a la consolidación de un equipo efectivo y comprometido a un solo propósito: el éxito del proyecto.
- Todos lo involucrados tienen voz.
- Permite lograr un acuerdo entre los involucrados y el equipo de desarrollo sobre lo que la aplicación debe y no debe hacer.
- Se exponen y resuelven aspectos políticos que interfieren con el éxito del proyecto.
- Existe retroalimentación constate y distintos puntos de vista de las necesidades, de los problemas y de las expectativas.
- La salida inmediata, es una definición preliminar de las características de alto nivel del sistema.

No obstante el reunir y coordinar a un grupo de personas con puntos de vista distintos para lograr un consenso no es tarea fácil. Las siguientes recomendaciones pueden ayudar a que esta técnica resulte efectiva:

- Vender la idea de que no es una reunión más, sino es la oportunidad de que los involucrados participen para obtener lo que quieren, resolver un problema y obtener beneficios comunes.
- Asegurar la participación del los involucrados correctos.
- Contar con la logística adecuada (traslados, lugar, materiales, entre otros.)
- Proporcionar materiales previos (De 2 días a una semana antes).
- Proporcionar información específica sobre el proyecto, por ejemplo copias de entrevistas anteriores, cartas de los clientes, diagnósitcos y análisis anteriores.
- Proporcionar además materiales fuera del contexto, por ejemplo: textos sobre creatividad, trabajo en equipo, administración de requerimientos, entre otros.
- Contar un moderador que de preferencia no sea miembro del equipo para lograr imparcialidad en las decisiones.
- Establecer una agenda con horario, asunto y descripción. Respetarla.

El moderador, entre otras responsabilidades, debe mantener un tono profesional y objetivo para el encuentro, comenzar y terminar el encuentro a tiempo, establecer y hacer cumplir las reglas para el encuentro, presentar los objetivos y la agenda para el encuentro, facilitar el proceso de toma de decisiones, proporcionar las facilidades

para asegurar que se cumpla la agenda, tratar que todos los involucrados participen y escuchen y controlar el comportamiento de los participantes.

3.2.2.4.4 Lluvia de Ideas

Las ideas más creativas e innovadoras son resultado de la combinación de múltiples ideas, es por esto que la lluvia de ideas es una de las técnicas más efectivas en la identificación de requerimientos. Consta de dos fases: generación de ideas y reducción de ideas (analizar las ideas generadas).

Entre las ventajas que proporciona esta técnica están:

- Permite la participación espontánea de todos los participantes.
- Se genera gran número de ideas.
- Permite asociar otras ideas.
- Típicamente el resultado es un conjunto de soluciones.
- Permite pensar sin los límites normales.
- Se requieren recursos mínimos para realizarla.

Para una adecuada sesión de lluvia de ideas, se recomienda no permitir criticismo o debate, permitir surgir la imaginación, generar el máximo de ideas posible y mantener una duración máxima de 2 a 3 horas.

En la organización de una sesión de lluvia de ideas hay que reunir a los involucrados y explicarles las reglas y el objetivo del proceso. Las fases del proceso son:

1. Generar las ideas y escribirlas conforme van surgiendo; en esta fase no se cuestionan, critican o analizan profundamente.
2. Reducir ideas, para lo cual hay que definirlas con más precisión, podarlas, organizarlas, clasificarlas, agruparlas, refinarlas y ponderarlas. Para la ponderación se pueden considerar los siguientes parámetros: crítica (indispensable), importante y útil.

3.2.2.4.5 Presentaciones

El propósito de los *storyboards* o presentaciones es obtener ofrecer una exposición de la problemática, de los requerimientos identificados, de un prototipo del sistema o de cualquier otro aspecto que sirva como base para la especificación exacta de los requerimientos y para tener una retroalimentación temprana de los usuarios. Pueden

ser pasivas, activas o interactivas. Algunas de las ventajas de las estas presentaciones son las siguientes:

- Son amigables, informales y pueden ser interactivas.
- Pueden tomar diversos enfoques, de acuerdo a lo que se quiera analizar, discutir o validar.
- En su caso, proporcionan una revisión temprana de las interfaces de usuario del sistema, permiten comprender la visualización de datos y así acelerar el desarrollo conceptual de diferentes facetas de la aplicación.
- Son fáciles de crear y modificar ya que existen diversas herramientas para su creación.
- Permiten definir reglas del negocio que serán implementadas en una aplicación.
- Son útiles por ejemplo para definir algoritmos y demostrar salidas de la aplicación.
- También son útiles en la validación de requerimientos por parte del usuario, una vez que se encuentran especificados.

En el desarrollo de una presentación es importante considerar: quién es el involucrado (usuarios del sistema), qué se está representando (comportamiento de los usuarios cuando interactúan con el sistema, o viceversa) y cómo lo va a hacer.

Las herramientas para desarrollar un storyboard van desde papel hasta herramientas de software para crear presentaciones y animaciones como: Power Point, Harvard Graphics, HyperCard, SuperCard, Macromedia's Director, Cinemation, entre otras.

3.2.2.4.6 Juego de Roles

El juego de roles permite que el equipo de desarrollo experimente en el mundo de usuario directamente. Esta técnica es útil sobre todos porque algunos usuarios no pueden describir adecuadamente el proceso que ellos siguen, no reconocen que no siguen procedimientos preestablecidos, tienen patrones de trabajo muy arraigados o es necesario interactuar con los procesos para comprenderlos.

La técnica consiste en que el analista y otros miembros del equipo de desarrollo toman el lugar del usuario y ejecutan las actividades de su trabajo en el sitio real. Cuando por alguna razón esto no es posible, se utilizan técnicas alternativas como:

- Los ensayos, en los cuales cada participante sigue una guía que define un rol específico.
- Las técnica de las tarjetas CRC *(Class-Responsability-Collaboration)*, que se utilizan en análisis orientado a objetos para identificar las responsabilidades de las clases.

3.2.2.5 Guía de preguntas útiles para la obtención de requerimientos

Las preguntas específicas a realizar en la obtención de requerimientos dependen de las características del proyecto y de la naturaleza de la organización; no obstante a continuación e presentan una serie de cuestiones clave que pueden considerarse como guía general.

GUÍA DE PREGUNTAS ÚTILES PARA LA OBTENCIÓN DE REQUERIMIENTOS

I. CONTEXTO

- ¿Cuál es el perfil del usuario y de la organización? (Nombre, organización, puesto, responsabilidades clave, salidas que produce).
- ¿Quiénes están detrás de la iniciativa de trabajo? ¿De dónde surgió la iniciativa, qué áreas intervienen y con qué apoyo se cuenta?
- ¿Qué beneficios intangibles y económicos se esperan de la solución?
- ¿Cómo se evaluaría el éxito de la solución?

II. PROBLEMÁTICA

- ¿Qué problemas existen actualmente? ¿Cuál es su causa raíz? ¿Por qué existe el problema? ¿Cómo se resuelve actualmente?
- ¿Cuál es el entorno en que se da la problemática?

III. PROCESOS

- ¿Cuáles procesos están relacionados?
- Descripción general de los procesos.
- ¿Qué procesos son críticos en la organización?
- ¿En qué ambiente se utilizará la solución?

IV. VOLUMEN Y TIPO DE INFORMACIÓN

- ¿Qué tipo de información se va a procesar?
- ¿Quién provee dicha información?
- ¿Cómo se encuentra almacenada, representada o distribuida la

información?
- Volúmenes de información ¿Cuántas transacciones? ¿Cuántos MB? ¿Cuántos archivos?

V. PERFIL Y PARTICIPACIÓN DE LOS USUARIOS
- ¿Quiénes utilizarán la solución? ¿Quiénes son los usuarios? ¿A qué área pertenecen?
- ¿Cuántos usuarios utilizarán la solución? ¿Con qué rol?
- ¿Qué perfil tienen? ¿Cuáles es su formación profesional? ¿Cuáles es su formación en cómputo?
- ¿Tienen experiencia en este tipo de aplicaciones?
- ¿Qué tipo de capacitación será necesaria?

VI. CARACTERÍSTICAS DESEADAS
- ¿Cuáles son características principales que se esperan del sistema?
- ¿Qué funcionalidad debe de proporcionar?
- ¿Cuáles son las expectativas de fiabilidad y desempeño?
- ¿Qué características de calidad se esperan?
- ¿Quién dará soporte al producto? ¿Tienen necesidades especiales para el soporte?
- ¿Cuáles son sus requerimientos de seguridad?
- ¿Cómo será distribuido el software?

VII. RESTRICCIONES
- ¿Existen restricciones a considerar?
- ¿Con qué presupuesto se cuenta? ¿Que limitaciones financieras o de presupuesto son aplicables?
- ¿Existen aspectos políticos internos o externos que afecten las soluciones potenciales?
- ¿Existe alguna restricciones de la tecnología a utilizar? ¿Estamos prohibidos a nuevas tecnologías?
- ¿La solución a construir interactuará con otros sistemas? ¿Debe mantenerse compatibilidad con soluciones existentes?
- ¿Existe algún requerimientos de seguridad especial?
- ¿Existen políticas, estándares o lineamientos que debamos seguir?
- ¿Hay un tiempo definido para su desarrollo?
- ¿Existe algún requerimiento legal, regulatorio o de ambiente, u otros estándares que deban ser considerados?

VIII. INFRAESTRUCTURA
- ¿Qué plataformas usan? ¿Existen planes para plataformas futuras?
- ¿Con qué recursos de hardware y software cuentan actualmente?

- ¿Existen otros sistemas o aplicaciones que sean relevantes para este proyecto?
- ¿Se cuenta con licencias de software institucionales?

3.2.2.6 Recomendaciones

A continuación se mencionan algunas recomendaciones en la obtención de requerimientos:

- No confundir el qué con el cómo. La obtención debe enfocarse al qué.
- Cuidar que el alcance del proyecto sea definido, y que no sea ni muy corto ni muy largo.
- Obtener información de todos los involucrados clave. Las reuniones con muchos participantes pueden resultar lentas y tediosas; por el contrario, recolectar información de pocos representantes, o escuchar solamente determinados puntos de vista pueden no ayudar a obtener los requisitos completos.
- La actividad de obtención disminuye cuando el usuario no piensa en más "Casos de Uso" –los cuales serán tratados más adelante en este trabajo– sino más bien son flujos alternos o excepcionales de éstos; cuando los usuarios discuten aspectos que se han tratado en discusiones pasadas; cuando los nuevos casos de uso están fuera del alcance o son de baja prioridad con respectos a los otros.
- Evaluar la viabilidad y necesidad del sistema, preguntarse ¿Realmente es necesario el sistema? ¿Qué consecuencias tendrá el no desarrollar el sistema? ¿De qué manera directa o indirecta el sistema contribuirá con los objetivos del negocio? ¿Cómo afectará el sistema a otros sistemas que están operando?.
- Ser sensitivos a las consideraciones organizacionales y políticas, a la cultura organizacional, a las diferencias entre las áreas, a la falta de responsabilidad, a la visión de la organización, entre otras.
- Registrar la fuente y justificación de los requerimiento, quién los solicitó o los propuso.
- Definir el ambiente operacional del sistema (Plataforma, información de las interfaces, interacción con otros sistemas).
- Tomar como base los objetivos del negocio.
- Identificar todas las restricciones del dominio ya que orientarán o incluso dictarán el tipo de solución.

- Reutilizar requerimientos o modelos ya existentes.

3.2.3 Análisis y Negociación

Después de que se ha identificado un conjunto de requerimientos, es necesario analizarlos para descubrir conflictos, traslapes, omisiones e inconsistencias. El objetivo del análisis y negociación de requerimientos tiene como finalidad obtener un conjunto acordado de requerimientos completos, consistentes y no ambiguos, los cuales se usarán como base para el desarrollo del sistema.

Dentro de esta actividad los requerimientos son analizados en detalle y deben pasar por un proceso formal de negociación y resolución de conflictos en el que participen los involucrados para llegar a un acuerdo. Se descubren requerimientos faltantes, conflictos entre los requerimientos y requerimientos no viables, además de que estos deben ser clasificados y priorizados.

El análisis y negociación de requerimientos aplica generalmente a los requerimientos de alto nivel, es decir a los requerimientos de sistema o de usuario. Para realizar el análisis de requerimientos se requiere habilidad y experiencia para revisar cuidadosamente los documentos y pensar en las características, riesgos e implicaciones de cada uno. El proceso de negociación por su parte tiene como intención establecer compromisos que satisfagan a todos los involucrados. Esta última no es una actividad simple, ya que influyen factores organizacionales, políticos y personales de los involucrados.

El análisis y la obtención de requerimientos son procesos estrechamente relacionados. Muchas veces conforme se van obteniendo requerimientos, paralelamente se van analizando y en ese momento se resuelven posibles conflictos, o en su caso se debe volver a la actividad de obtención.

Para el análisis y la negociación de requerimientos son fundamentales las siguientes actividades: establecer el alcance del proyecto, así como clasificar y ponderar los requerimientos.

El objetivo del análisis es descubrir problemas en los requerimientos; no es propiamente una actividad de administración de la calidad.

3.2.3.1 Definición del alcance del proyecto

Definir los límites del sistema es una actividad clave en el análisis y negociación. Se debe establecer un conjunto acordados de requerimientos del sistema a desarrollar. Determinar cuáles son requerimientos de sistema, cuáles son del proceso de negocio y cuáles salen del alcance del sistema.

Esto no es siempre fácil, dado que la mayoría de los clientes y usuarios desean que el sistema a desarrollar les ofrezca la mayor funcionalidad como sea posible, comprometiendo con ello la calidad, la viabilidad y el éxito del proyecto.

Según Dean Leffingwell y Don Widrig, el alcance del proyecto está en función de tres factores:

- La funcionalidad que se debe proporcionar a los usuarios.
- Los recursos disponibles para el proyecto (equipo de trabajo, tecnología y presupuesto).
- El tiempo disponible.

Una estrategia para establecer claramente el alcance es establecer líneas base. Una línea base es *"Un conjunto detallado de características, o requerimientos que se pretenden entregar en una versión específica de la aplicación"*.

Una línea base debe ser aceptable por el cliente y tener una probabilidad razonable de éxito, desde el punto de vista del equipo de desarrollo. Para establecerla es importante considerar lo siguiente:

- Tener una lista de las características identificadas.
- Clasificar y ponderar cada característica, tanto por los clientes, usuarios, equipo de desarrollo y otros involucrados clave, y mínimo desde las siguientes perspectivas: importancia, esfuerzo requerido y riesgo. (Ver punto 3.2.3.2)

Con base en lo anterior, es posible obtener la línea base de un conjunto de características para determinar el alcance del proyecto de acuerdo con los factores mencionados anteriormente: funcionalidad, recursos y tiempo.

Para un proyecto se establecen tantas líneas base como se quieran, dependiendo también de la estrategia y la metodología de desarrollo. Se recomienda establecerlas para lograr liberaciones diversas del producto, dividir el proyecto en fases, y minimizar así el riesgo asociado.

3.2.3.1.1 Línea base

Una línea base está integrada por un conjunto de elementos de software generados en una fase del ciclo de vida de un producto, revisados y aprobados que:

- Representan una base acordada para continuar el desarrollo,
- Pueden ser una especificación o un producto revisado que posteriormente servirán como base en el desarrollo y mantenimiento del producto de software,
- Pueden ser modificadas mediante procedimiento de control formales tales como la Administración de la Configuración.

Una línea base formal es un producto, usualmente documentos que han sido expresamente revisados y acordados por el cliente, el usuario y el equipo de desarrollo, y el cual sirve como fundamento para el desarrollo subsecuente. Un sistema de administración de configuración es precisamente el mecanismo para mantener las diversas líneas base.

3.2.3.2 Clasificación y ponderación de los requerimientos

En el proceso de análisis y negociación se establecen valores para un conjunto de atributos de los requerimientos. Los principales atributos son: estado, prioridad y riesgo.

El ***estado*** nos indica el progreso de los requerimientos durante el ciclo de vida de desarrollo. Desde el punto de vista de la aceptación de los requerimientos, los tres estados más comúnmente utilizados son:

- ***Propuesto***. Requerimientos que están bajo discusión o negociación pero que todavía no son revisados y aceptados por el canal oficial, es decir por el conjunto de personas (clientes y equipo de desarrollo) que dan el visto bueno a un requerimiento.

- ***Aprobado.*** Requerimientos útiles y factibles de realizar que han sido aprobados por el canal oficial para su implementación.
- ***Incorporado.*** Requerimientos o características incorporadas dentro de una línea base del producto en un tiempo específico.

El equipo de desarrollo en conjunción con los clientes y otros involucrados deben participar en la clasificación y ponderación de requerimientos. La prioridad debe reflejar la importancia para los involucrados y para el éxito del proyecto. Asignar prioridades permite a los involucrados decidir sobre los requerimientos medulares para el sistema, orientar la arquitectura del sistema, resolver conflictos y establecer líneas base. Las prioridades deben ser acordadas por ambas partes.

La ***prioridad***, comprende tres posibles valores:

- ***Crítico o Esencial.*** Representan requerimientos esenciales. Si se falla en su implementación significa que el sistema no satisfará las necesidades del usuario. Todas las características críticas deben ser implementadas.
- ***Importante.*** Representan requerimientos importantes para la efectividad y eficiencia del sistema. La funcionalidad no puede ser fácilmente proveída de otra manera. El no incluirlos puede afectar la satisfacción de los clientes y usuarios, pero una liberación de una versión no será retrasada por la falta de una característica importante.
- ***Deseable.*** Representan requerimientos de utilidad que serán usados con menor frecuencia. De no ser incluidos, no se tiene un impacto significativo en la satisfacción del usuario.

Otro atributo clave es el ***riesgo*** asociado a cada requerimiento. El riesgo es la probabilidad de que el requerimiento sea causa de eventos indeseables, como mayores costos, retrasos, o cancelación. Algunos aspectos de riesgo pueden ser: complejidad del requerimiento, tecnología para implementar el requerimiento, tamaño del requerimiento, entre otros.

Por cada conjunto de requerimientos es recomendable realizar un análisis de riesgos. El análisis de riesgos es un proceso analítico que implica conocimiento, experiencia y juicio de los involucrados para determinar dos atributos básicos del riesgo: el impacto en el proyecto y la probabilidad de ocurrencia.

El impacto del riesgo en los requerimientos puede ser:

- ***Bajo***. El requerimiento representa un riesgo leve para el proyecto.
- ***Medio***. El requerimiento representa un riesgo considerable para el proyecto.
- ***Alto***. El requerimiento representa un riesgo grave para el proyecto.

Con base en esta ponderación, los requerimientos que conviene implementar primeramente son los más importantes y de mayor riesgo.

Existen otros atributos que también deben ser valorados durante el análisis de requerimientos como son:

- ***Esfuerzo***. Estimación del número de personas, horas hombre, líneas de código, puntos de función, entre otras. Este atributo es importante para definir adecuadas líneas de base.
- ***Estabilidad***. Una medida de la probabilidad de que la característica cambiará o que la comprensión del equipo de desarrollo respecto a la característica cambiará. Permite establecer prioridades y determinar cuáles aspectos retomar en encuentros próximos con los involucrados.
- ***Versión destino***. Especifica la versión del producto estimada en la cual aparecerá la característica por primera vez.
- ***Asignado a.*** En muchos proyectos, las características son asignadas a "equipos de características" responsables de la obtención, especificación y quizá implementación de la misma.
- ***Razón***. Fuente de la característica requerida. Por ejemplo, la referencia a algún documento, objetivo de negocio, disposición legal o a una importante entrevista con el usuario.

3.2.3.3 Guía para el análisis de requerimientos

A continuación se presenta una breve serie de preguntas útiles para el análisis de requerimientos:

GUÍA PARA EL ANÁLISIS DE REQUERIMIENTOS

- ¿El requerimiento es acorde con los objetivos del negocio y el alcance del proyecto?
- ¿El requerimiento entra en conflicto con alguna restricción de dominio, política o regulación?
- ¿Es necesario el requerimiento?
- ¿El requerimiento se traslapa o entra en conflicto con otro?
- ¿El requerimiento es viable económica y técnicamente ?
- ¿El requerimiento exige cumplir con algún estándar de hardware o software?
- ¿El requerimiento es viable dada la tecnología y recursos disponibles para implementar el sistema?
- ¿La descripción de un requerimiento describe un solo requerimiento o debe ser dividido en varios requerimientos distintos?
- ¿A cada requerimiento se le ha asignado un estado, prioridad y riesgo?
- ¿Se ha considerado la portabilidad confiabilidad, facilidad de uso y facilidad de mantenimiento para el sistema?
- ¿Están claramente definidos los alcances del proyecto y del sistema?
- ¿Se han establecido líneas base basadas en requerimientos?

3.2.3.4 Recomendaciones

Algunas recomendaciones en el análisis y negociación de requerimientos se presentan a continuación:

- Negociar fechas, precio y alcance de forma que sea satisfactorio para el cliente y viable para el equipo de desarrollo.
- En la medida de lo posible, planear la posible ocurrencia y resolución de conflictos. Favorecer encuentros cara a cara en los que se resuelvan los conflictos.
- No pasar por alto la ponderación de requerimientos, es un mecanismo importante en la planeación y gestión del riesgo.
- Clasificar los requerimientos usando un enfoque multidimensional. Esto permite ubicar la relación entre los requerimientos y cuando ocurre un cambio, conocer los requerimientos afectados. Ejemplos de clasificaciones para agrupar requerimientos son: de sistema, de interfaz de usuario, de base de datos, de comunicaciones, de seguridad, entre otras.
- Usar matrices de interacción para encontrar conflictos y traslapes. Una matriz de interacción es una matriz donde las columnas y renglones son etiquetadas con identificadores de los requerimientos, el valor en la intersección entre dos requerimientos indican la dependencia o conflictos entre estos. (Ver tema de rastreabilidad en el punto 3.2.6)
- Identificar cada requerimiento de manera única.
- Registrar la fuente de cada requerimiento. (Quién lo propuso, de dónde surgió).
- Evaluar el riesgo de los requerimientos.

3.2.4 Modelado y Especificación

La especificación de requerimientos es la actividad en la cual se genera el documento que contiene la descripción completa de las características y funcionalidades del sistema, así como su alcance. Es por lo tanto la representación y documentación de los requerimientos funcionales y no funcionales del sistema o del software.

Comúnmente se le llama especificación de requerimientos de software (SRS por sus siglas en inglés) al documento que contiene la definición completa de los requerimientos de software. Este documento describe de manera clara y precisa cada uno de los requerimientos esenciales (funciones, desempeño, restricciones de diseño y atributos de calidad) del software y sus interfaces externas.

El documento de requerimientos de sistema es útil para diversas personas:

- Los **clientes y usuarios** del sistema quienes así validan que los requerimientos son adecuados a sus necesidades.
- Los **líderes de proyecto** para planear, costear y calendarizar el desarrollo del sistema.
- Los **ingenieros de software y arquitectos del sistema** responsables de diseñar e implementar el sistema.
- Los **probadores**, quienes usan el documento para generar pruebas y verificar que el sistema desarrollado satisfaga los requerimientos.
- El **personal** que mantiene y modifica el sistema, para comprender las características iniciales del sistema y las relaciones entre las diferentes partes del sistema.
- Los **documentadores** que realizan los manuales de usuario y técnicos, con base en la funcionalidad requerida y ofrecida por el sistema.

3.2.4.1 Características deseables en las especificaciones

De acuerdo con el estándar 830 de la IEEE, para que las especificaciones de requerimiento cumplan características de calidad, deben ser:

Atributo	Descripción
Correctas	Una especificación de requerimientos de software es correcta sí y solo sí cada requerimiento representa algo realmente necesario del sistema por construir. El usuario también valida que sean correctos.
Sin ambigüedad	Un requerimiento es no ambiguo si y sólo si puede ser sujeto a una sola interpretación por distintas personas.
Completas	Un requerimiento está completo si no necesita ampliar detalles para expresar la funcionalidad que expresa; es decir, si proporciona la información suficiente para su comprensión.
Consistentes	Un requerimiento es consistente si y sólo si esta redactado conforme a su objetivo específico y no entra en conflicto con otros requerimientos.
Verificables	Un requerimiento es verificable si y sólo si cada uno de los componentes contenidos en el requerimiento se pueden probar. Los requerimientos pueden ser considerados verificables si y sólo si existe un proceso finito con el cual una persona o máquina pueden determinar que el sistema de software desarrollado satisfaga los requerimientos.
Modificables	Un requerimiento es modificable si y sólo si su estructura y estilo son tales que cualquier cambio puede ser hecho de forma fácil, completa y consistente, conservando la uniformidad y estructura con los demás.
Con posibilidad de rastreo	Un requerimiento se puede rastrear si y sólo si el origen de cada uno de sus componentes es claro, y si existe un mecanismo que haga posible referirse al requerimiento a lo largo del ciclo de desarrollo. Esta característica es importante para reflejar cambios y evaluar el impacto de éstos.
Comprensibles	Un conjunto de requerimientos es comprensible si tanto la comunidad de usuarios como de desarrolladores son capaces de comprender plenamente los requerimientos por separado y la funcionalidad total del sistema.

Otras características deseables que sugieren otros autores son las siguientes:

Factibles	Debe ser evaluado para que se pueda realizar con los recursos (financieros, tecnológicos, técnicos, legales) disponibles.
Independientes de la implementación	Debe indicar el "qué" no el "cómo".

En relación con esta última característica, los requerimientos describen lo que el sistema debe hacer y el diseño describe cómo los requerimientos serán cumplidos. No obstante la relación entre los requerimientos y el diseño puede ser circular. Por ejemplo un requerimiento de sistema (característica) es el "qué" de un requerimiento de software, el cual en este caso puede interpretarse como un "cómo". Tomando como referencia el mismo requerimiento de software puede ser un "qué" con respecto al diseño detallado.

Para asegurar que una sentencia de requerimiento de sistema representa una necesidad y no una implementación, *Richard Hardwell* recomienda preguntar "por qué" el requerimiento es necesario. La respuesta debe ser una necesidad real.

3.2.4.2 Técnicas para el modelado y especificación

3.2.4.2.1 Casos de Uso

> *Un caso de uso es una descripción de un conjunto de acciones que el sistema realiza para ofrecer algún resultado de valor a un actor en particular.* [Booch]
>
> *"Los Casos de Uso han sido adoptados casi universalmente para la captura de requisitos de sistemas software en general, y de sistemas basados en componentes en particular..."*

Un ***caso de uso*** es un fragmento de funcionalidad del sistema que proporciona al usuario un resultado importante. Los casos de uso representan los requerimientos funcionales. Todos los casos de uso juntos constituyen el modelo de casos de uso, el cual describe la funcionalidad total del sistema.

El ***modelo de casos de uso*** está compuesto por todos lo actores y todos los casos de uso de un sistema. Un ***diagrama de casos de uso*** describe parte o todo el modelo de casos de uso y muestra un conjunto de casos de uso y actores asociados. Este modelo sirve como un medio de comunicación entre el equipo de desarrollo y puede servir como un contrato entre el cliente, los usuarios y el equipo de desarrollo sobre la funcionalidad total del sistema. Permite que los clientes y usuarios validen que el sistema hará lo que ellos esperan y el equipo de desarrollo construya lo esperado.

Normalmente, un sistema tiene muchos tipos de usuarios. Cada tipo de usuario se representa con un actor. Los actores utilizan el sistema interactuando con los casos de uso.

1) Usuarios de los Casos de Uso

Los Caso de Uso resultan de gran utilidad para:

- **Los clientes**, ya que así visualizan, validan y aprueban lo que el sistema debe hacer.
- **Los usuarios** porque ganan entendimiento del sistema. Son los destinatarios de las acciones del sistema y pueden complementar el modelo.
- **Los analistas**, ya que les proporciona las bases para el análisis y diseño conceptual.
- **Los desarrolladores**, que les sirve como documento del comportamiento del sistema para implementar el software con la funcionalidad de los casos de uso.
- **Los probadores** que los usan como base para las pruebas de los distintos escenarios del sistema.
- **El líder del proyecto** para establecer el alcance, determinar la planeación y los recursos del proyecto.
- **Al documentador** le sirve como base para el manual de usuario.
- **A los demás involucrados del negocio** les permite comprender las características del sistema y replicarlas.

2) Beneficios

Existen varios motivos por los cuales los casos de uso son buenos, se han hecho populares y se han adoptado universalmente:

- Proporcionan un medio sistemático e intuitivo de capturar requerimientos funcionales, centrándose en el valor agregado para el usuario.
- Dirigen todo el proceso de desarrollo debido a que las actividades subsecuentes toman como base los requerimientos funcionales.
- La perspectiva que proporcionan los casos de uso apoyan la creación de productos enfocados a los clientes.
- El modelo de casos de uso se utiliza para conseguir un acuerdo con los usuarios y clientes sobre qué debería hacer el sistema. Esta especificación puede utilizarse como parte del contrato con el cliente.
- Los casos de uso son intuitivos por lo que los clientes y usuarios no tienen que aprender una notación compleja.
- Los casos de uso ayudan a los jefes de proyecto a planear, asignar y controlar muchas de las tareas que los desarrolladores realizan. A partir de los casos uso se puede estimar el tamaño del proyecto y los recursos necesarios.
- Los casos de uso ayudan a llevar a cabo el desarrollo iterativo. Cada incremento del desarrollo es una realización funcional de un conjunto de casos de uso.
- Los casos de uso se utilizan como punto de partida para escribir el manual de usuario, ya que cada caso de uso describe una forma de utilizar el sistema.
- Los casos de uso son un método poderoso para comprender los requerimientos desde un punto de vista del usuario del negocio.
- Son relativamente fáciles de escribir, escritos en el lenguaje del usuario y comprendidos tanto por el usuario como por el desarrollador.
- Los escenarios de los casos de uso pueden servir como guía de prueba.
- Facilitan el re-uso.
- Se enfocan en el “qué”.
- Identifican a los usuarios y los límites del sistema.
- Identifican las interfaces del sistema.
- El UML se está convirtiendo en un estándar de la industria y es soportado por diversas herramientas de modelado.

3) Elementos

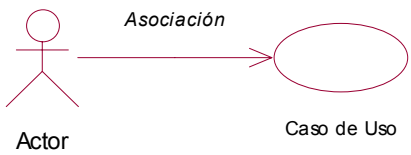
- **Actor**. Un Actor representa cualquier cosa que interactúe con el sistema. Los Actores no son parte del sistema; representan los roles que pueden jugar los usuarios del sistema. Un Actor puede intercambiar información activamente con el sistema o bien ser un receptor pasivo de información. Un Actor puede ser una persona, una máquina u otro sistema. Cada Actor participa en uno o más Casos de Uso. Se representa con el siguiente símbolo:

Actor

- **Caso de Uso.** Cada forma en que los actores usan el sistema se representa con casos de uso. Un caso de uso especifica una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto. Se representa de la siguiente forma:

Caso de Uso

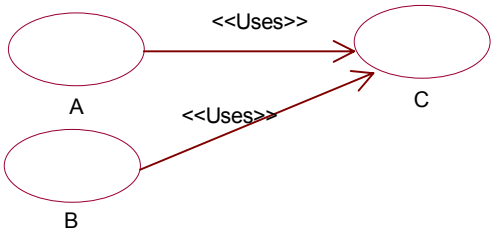
- **Asociación.** Una asociación representa la interacción entre el actor y el caso de uso. Indica qué actor inicia el caso de uso. Se representa con una flecha como sigue:

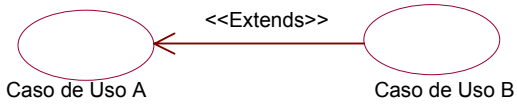


Existen además dos tipos de relaciones especiales: "uses" y "extend". Las relaciones "**uses"** ocurren cuando se tiene una porción de comportamiento que es similar en más de un caso y no se quiere duplicar la descripción de tal conducta. **"Uses"** permite incluir la misma funcionalidad en dos o más Caso de Uso separados sin necesidad de repetir los detalles.

Se utiliza la relación "extends" cuando se tiene un Caso de Uso que es similar a otro, pero que hace un poco más. "Extends" describe una variación de la conducta normal.

Se representan en UML por medio de un estereotipo de la relación, de la siguiente manera:





4) Documentación

En complemento con el modelo de Casos de Uso, se escribe un documento de especificación por cada Caso de Uso, el cual contiene lo siguiente:

- **Identificador** del Caso de Uso que puede ser un número o clave que lo identifique.
- **Nombre** del Caso de Uso que represente la funcionalidad, nombrándolo desde el punto de vista de vista del usuario.
- **Breve Descripción**. El propósito es explicar en pocas líneas (2 ó 3) el Caso de Uso.
- **Pre-condiciones**. Requisitos necesarios para que el Caso de Uso se lleve a cabo.
- **Flujo Principal.** Secuencia general de acciones que se dan normalmente.
- **Flujos Alternos.** Secuencias de acciones que detallan las posibles alternativas del Caso de Uso.
- **Flujos de Excepción**. Eventos extraordinarios o condiciones de error del Caso de Uso.
- **Post-condiciones**. Requisitos que deben cumplirse posteriormente.

5) Construcción de casos de uso

Elaborar los casos de uso es relativamente sencillo, si bien no existe una fórmula o una manera única de hacerlo, se pueden seguir los pasos:

1. Encontrar **actores**, preguntándose ¿quién interactuará con el sistema? ¿a quién se le ofrecerá determinada funcionalidad? ¿qué roles toman los diversos usuarios del sistema? ¿existe interacción con otros sistemas o aplicaciones? ¿existe algún departamento o entidad que tiene que ver con la funcionalidad del sistema?
2. Encontrar los casos de uso. ¿Qué funcionalidad requiere determinado actor? ¿Mediante qué funciones se va lograr el objetivo del sistema? ¿Qué debe realizar el sistema?
3. Describir brevemente cada caso de uso, es decir un primer acercamiento a las especificaciones (Punto 4-Documentación).
4. Detallar el modelo de casos de uso completo, con sus actores, casos de uso y relaciones, en su caso de tipo “uses” y “extends”.

5. Detallar casos de uso, escribiendo la especificación completa de éstos.

Un modelo de casos de uso está casi terminado cuando recoge todos los requerimientos funcionales de manera que pueden comprenderlos los clientes, usuarios y desarrolladores.

6) Consideraciones

Es necesario tener en cuenta lo siguiente en la construcción de Casos de Uso.

- Los casos de uso deben mantenerse tan independientes unos de otros como sea posible.
- Los casos de uso deben escribirse utilizando el lenguaje del cliente. Un caso de uso es más efectivo si esta escrito desde la perspectiva del usuario, como un conjunto de sentencias en presente y en voz activa.
- Cuestionar los casos de uso que no tienen flujos alternos. Se dice que más del 90% tienen al menos un flujo alterno. Además verificar que se hayan considerado todos los flujos alternos.
- Los casos de uso deben ser breves. El flujo básico debe ser de dos o tres párrafos a lo mucho.
- Comenzar con verbo el nombre del caso de uso.
- No confundir cada caso de uso con una interfaz del sistema
- Presentar el diagrama de casos de uso de tal manera que se sugieran los límites del sistema

Asimismo conviene prestar especial atención a síntomas como:

- Los casos de uso son muy pequeños, ya que quizá se estén considerando operaciones básicas en lugar de casos de uso.
- Existen muchos casos de uso porque están muy desglosados (granularidad) o porque salen del alcance del sistema.
- Existen casos de uso sin un resultado de valor para un actor, es decir, no son casos de uso.
- Sus nombres hacen referencia a operaciones bajo nivel.
- Los clientes y usuarios no comprenden los casos de uso.
- Los casos de uso no son escritos del punto de vista del actor. Es decir, son escritos (y titulados) del punto de vista del sistema.
- El límite del sistema es indefinido o inconstante.

- Los nombres de los actores son inconsistentes.
- Las relaciones parecen telarañas.
- Las especificaciones son muy largas o confusas.

Los casos de uso también tienen sus limitaciones, y no se recomiendan cuando el sistema tiene pocos usuarios e interfaces mínimas o cuando la mayoría de los requerimientos son no funcionales y restricciones de diseño.

7) Guía de preguntas para revisar y refinar los casos de uso

GUÍA PARA REVISAR Y REFINAR LOS CASOS DE USO

- ¿Cada Caso de Uso se relaciona con al menos un actor?
- ¿Cada Caso de Uso es independiente de otros?
- ¿Cada Caso de Uso tiene un nombre único, intuitivo y explicativo?
- ¿La breve descripción da un panorama del Caso de Uso?
- ¿Se han encontrado todos los actores?
- ¿Cada actor es relacionado con al menos un Caso de Uso?
- ¿Se pueden nombrar al menos una persona que sea capaz de realizar el rol de determinado actor?
- ¿Los actores juegan roles similares o el mismo rol en el sistema?
- ¿Un actor en particular interactua con el sistema en maneras completamente distintas?
- ¿Los actores tienen nombres intuitivos y descriptivos?
- ¿Están representado todos los requerimientos funcionales en los Casos de Uso?
- ¿El Modelo de Casos de Uso contiene algún comportamiento superfluo o redundante además de las funciones originalmente requeridas?
- ¿Hay Casos de Uso con comportamientos o flujos similares? ¿Cada parte de un flujo de eventos ha sido modelado como otro Caso de Uso?
- ¿El flujo de un Caso de Uso debería ser insertado dentro del flujo de eventos de otro? ¿El modelo hace uso de relaciones “uses” y “extends” para evitar la redundancia?
- Si se agrupan en paquetes, ¿La división en paquetes es apropiada? ¿Se hace el modelo más entendible, intuitivo y fácil de mantener?
- ¿Los clientes y usuarios comprenden los nombres y descripciones de los Casos de Uso?
- ¿Es claro cuánto comienza y termina el flujo de un Caso de Uso?
- ¿Existe la descripción del comportamiento en condiciones de excepción?

3.2.4.2.2 Prototipos

Un prototipo es un modelo (representación, demostración o simulación) fácilmente ampliable y modificable de un sistema, probablemente incluyendo su interfaz y su funcionalidad de entradas y salidas.

Un prototipo de requerimientos de software es una implementación parcial del sistema de software, que ayuda a los desarrolladores, usuarios y clientes a representar, comprender mejor y validar los requerimientos del sistema.

Permiten por un lado identificar que algo no es lo que el usuario quiere, así como nuevos requerimientos. Por ser desarrollados tempranamente pueden ayudar a eliminar el riesgo y son efectivos porque permiten al usuario interactuar con un prototipo en su ambiente, el cual da un acercamiento al mundo real.

Los prototipos pueden ser:

- ***Prospectos o experimentales***. Prototipos no reutilizables, utilizados para definir las metas del proyecto, identificar nuevos requerimientos y validarlos.
- ***Evolutivos u operacionales.*** Prototipos iterativos que son progresivamente refinados hasta que se convierten en el sistema final.

- ***Verticales.*** Modelan pocas características de un sistema pero con mucho detalle.
- ***Horizontales***. Prototipo que modela muchas características de un sistema pero con poco detalle.

- ***De interfaces de usuario***. Representan el diseño, contenido, organización y secuencia de la interfaces del sistema; no tienen ninguna funcionalidad.
- ***Algorítmicos***. Implementan todo o alguna una parte de un cálculo o proceso.

Algunas herramientas útiles para el desarrollo de prototipos son:

- Papel y lápiz.
- Software de diseño como Visio o Corel.
- Software para desarrollar presentaciones como Power Point.

- Software de animación y presentaciones como Flash, Director y Cinemation.
- Herramientas visuales para RAD, como Visual Basic y Borland Delphi .

3.2.4.2.3 Otras técnicas de especificación

Existen además diversas técnicas de especificación, algunas de las cuales complementan a los casos de uso y prototipos; otras se utilizaron hace algunos años con los lenguajes estructurados. A continuación se da una breve descripción de éstas.

- ***Lenguaje Natural***. Redactar en un documento los requerimientos en lenguaje natural y sin formalidad puede ser una alternativa de especificación, no obstante puede resultar incompleto y ambiguo. El lenguaje natural generalmente complementa y hace más explícitas otras técnicas de especificación.
- ***Pseudocódigo***. Combina la informalidad del lenguaje natural con la sintaxis y las estructuras de control de un lenguaje de programación.
- ***Máquinas de estado finito***. Una máquina hipotética que puede estar en uno y sólo un número determinado de estados en un tiempo específico. Se representan generalmente con los diagramas de estado.
- ***Tablas y árboles de decisión***. Cuando se requiere representar el comportamiento de que distintas combinaciones de entradas, ofrecen determinadas salidas.
- ***Modelos entidad-relación***. Si los requerimientos dentro de un conjunto involucran una descripción de la estructura y relaciones entre datos es conveniente utilizar un modelo entidad-relación.
- ***Análisis Estructurado (SA).*** Especifica los requerimientos mediante procesos, flujos de información y almacenamientos. Utiliza diagramas de contexto, de flujo, de flujo de datos (DFD), diagramas modulares y diagramas entidad-relación.
- ***Análisis y Diseño Orientado a Objetos***. Mediante la notación de UML, se modela el sistema con una perspectiva de objetos, en la que un objeto es una entidad que conjunta características (datos) y operaciones (procesos). Los casos de uso como parte de UML modelan los requerimientos.

3.2.4.3 Modelado del dominio y modelado del negocio

Un *modelo de dominio* captura los tipos más importantes de objetos en el contexto del sistema. Los objetos representan las "cosas" que existen o los eventos que suceden en el entorno en que trabaja el sistema.

El *modelado del negocio* es una técnica para comprender los procesos de negocio de la organización.

Un *modelo de casos de uso del negocio* describe los procesos de negocio de una empresa en términos de casos de uso del negocio y actores del negocio. Es una vista externa del negocio. En este caso el sistema es el negocio.

Un *modelo de objetos del negocio* es un modelo interno a un negocio. Describe cómo cada caso de uso de negocio es llevado a cabo por parte de un conjunto de trabajadores que utilizan un conjunto de entidades de negocio y unidades de trabajo.

El modelado de negocio se usa en los sistemas de aplicación específica. Su propósito es comprender la estructura y dinámica de la organización y asegurar que los clientes, usuarios finales y desarrolladores tengan una comprensión general de la misma.

3.2.4.4 Guía de contenido para la especificación de requerimientos

No existe una guía única para la especificación de requerimientos; muchas veces dependen el tipo de proyecto y del enfoque de la organización. Existen sin embargo diversas propuestas para especificar los requerimientos, por ejemplo: el estándar 830-1984 de la IEEE, las plantillas del Proceso Unificado de Rational, la guía de VOLERE, además de otras que los autores proponen o complementan en sus obras.

A continuación se presenta una guía general que integra los principales aspectos de todas estas propuestas, considera los Casos de Uso como parte de la especificación de los requerimientos funcionales y además incorpora otros elementos que en la práctica resultan útiles. Considera por ejemplo, algunos aspectos generales de la administración del proyecto, que si bien deben ser detallados en un documento especifico, algunas veces conviene incluir aspectos clave para ubicar mejor el contexto. También cabe resaltar que se integró una parte relacionada con el modelado de negocio para comprender la estructura y procesos de la organización.

La guía cubre diversos aspectos genéricos, pero puede adaptarse al tipo de organización, al tipo de proyecto, e incluso a quien va dirigido el documento.

GUÍA DE CONTENIDO PARA LA ESPECIFICACIÓN DE REQUERIMIENTOS

I. INTRODUCCIÓN

- **Presentación**. Presentación y objetivo del documento.
- **Convenciones**. Aspectos tipográficos para comprender mejor el documento, así como definiciones, acrónimos y abreviaciones.
- **Propósito y alcance**. Objetivo del producto. Breve descripción del producto, alcances y beneficios del mismo.
- **Problemática que resuelve**. Necesidades o deficiencias que satisface el producto. Contraste con los beneficios. Justificación.
- **Clientes, usuarios e involucrados**. En caso de tratarse de un producto a vender se refiere a los clientes y usuarios potenciales. Para un sistema específico, los roles, experiencia, niveles y prioridades de los involucrados en el proyecto.

II. CONTEXTO DE LOS USUARIOS Y LA ORGANIZACIÓN (AMBIENTE OPERACIONAL)

- **Objetivos de la organización.** Antecedentes, misión y objetivos de la organización destinataria del proyecto
- **Estructura de la organización.** Organigrama, áreas involucradas, y distribución regional de relevancia para el sistema. Área o usuarios finales del sistema.
- **Modelado del negocio**. Procesos actuales de la organización relevantes para el sistema.

III. REQUERIMIENTOS

- **Requerimientos Funcionales**
  - **Modelo de Casos de Uso.** Diagrama general de los casos de uso del sistema.
  - **Descripción de los Actores.** Breve descripción de los actores.
  - **Concentrado de los requerimientos funcionales.** Matriz concentrada de los casos de uso con sus atributos: tipo, estado, prioridad, riesgo, etc.
  - **Especificaciones de los Casos de Uso**
    - **Caso de Uso 1**
      - Identificador
      - Nombre
      - Pre-condiciones
      - Flujo principal

        - Flujos alternos
        - Flujos de excepción
        - Post-condiciones
    - **Caso de Uso 2**
    - **...**
    - **Caso de Uso n**

- **Requerimientos no Funcionales**
    - **Apariencia y estilo**. Descripción de las características de la Interfaces de usuario relacionadas con estándares de despliegue, resolución, distribución, menús, entre otros.
    - **Uso**. Facilidad de operación y de aprendizaje, dependiendo de las habilidades y experiencia de los usuarios, así como de la complejidad del producto.
    - **Confiabilidad**. Exactitud de los resultados esperados, tolerancia a fallas, facilidad de recuperación.
    - **Desempeño**. Permite especificar el tiempo esperado para completar determinadas tareas, así como los volúmenes de procesamiento, de almacenamiento, de usuarios simultáneos, entre otros.
    - **Soporte y mantenimiento**. Condiciones especiales de mantenimiento, de portabilidad, de configuración.
    - **Seguridad**. Especificaciones de acceso, control y confidencialidad para conservar la integridad del sistema.
    - **Documentación del usuario y ayuda en línea.** Lista de los productos de documentación del sistema requeridos. Aspectos de ayuda en línea.
    - **Otros**. Por ejemplo, requerimientos de migración de información o reemplazo de un sistema anterior.

- **Restricciones**
    - **Legales**. Violación de derechos de autor, contratos o licitaciones.
    - **Tiempo**. Tiempo máximo de desarrollo.
    - **Presupuesto**. Presupuesto disponible para el proyecto.
    - **De diseño**. Plataforma, estándar, arquitectura a cumplir.
    - **Estándares**. Estándares tanto de los clientes y usuarios como del grupo de desarrollo. Estándares de diseño, codificación y trabajo en grupo.
    - **Otras**.

- **Requerimientos de Interfaces.** Aspectos relacionados con la conectividad, comunicación y acceso hacia o desde componentes externos.
    - **Interfaces de hardware.** Comunicación con dispositivos de hardware.
    - **Interfaces de software**. Compatibilidad con componentes comerciales.
    - **Interfaces de comunicaciones.** Protocolos e interfaces de comunicación.

IV. ASPECTOS DEL PROYECTO

- **Estrategia de desarrollo.** Fases del desarrollo, metodología a utilizar. Planeación general, recursos y tiempo estimados, riesgos a considerar
- **Plataforma y requerimientos de desarrollo.** Sistema operativo, lenguajes, manejadores de bases de datos y otras herramientas a utilizar.
- **Entrega del producto**. Licencias, distribución, instalación, protección del código fuente.
- **Glosario de términos**

3.2.4.5 Recomendaciones

Además de las características de calidad de las especificaciones de requerimientos mencionadas anteriormente y las técnicas propuestas, se sugiere:

- Utilizar plantillas para documentar los requerimientos.
- Usar un lenguaje simple, consistente y conciso.
- Usar apropiadamente los diagramas. Complementar con lenguaje natural los requerimientos descritos de otras formas.
- Especificar cuantitativamente los requerimientos, ya que esto sirve como base para las pruebas de aceptación. Es necesario decidir la métrica adecuada para describir el requerimiento y establecerle el valor adecuado. Por ejemplo:
  - Desempeño. Número de transacciones por segundo, tiempo de respuesta para las entradas del usuario.
  - Almacenamiento. Tamaño máximo del sistema en KB.
  - Facilidad de uso. Tiempo promedio para aprender el 75% de las facilidades.
  - Robustez. Tiempo para reestablecerse después de una falla.
- Modelar el ambiente del sistema. Hacer un modelado del negocio relacionado con el sistema.
- Modelar la arquitectura del sistema.
- Usar un glosario y diccionario de datos. Definir términos especializados.
- Documentar las ligas entre los requerimientos del sistema y los requerimientos de los involucrados (rastreo).
- Explicar cómo llenar y usar el documento plantilla.

- Incluir un resumen o matriz de los requerimientos en el documento para facilitar la consulta y obtener una visión del tamaño y estado del proyecto.

3.2.5 Validación y Verificación

La validación y verificación permiten monitorear el proceso y los productos en el desarrollo de software. La validación y verificación (V&V) juegan un importante papel en el desarrollo de software con calidad.

> *La **validación** es el proceso de asegurar que lo que se está construyendo corresponde con los realmente requerido, que sea completo, consistente y de acuerdo con los requisitos.*
>
> *La **verificación** es el proceso de determinar si los productos de una determinada fase del ciclo de desarrollo de software, cumplen o no los requerimientos establecidos durante el inicio de la fase.*

La validación asegura que el sistema trabaje conforme a los requerimientos especificados. La verificación es un proceso analítico que trabaja a lo largo del proyecto para asegurar que se estén haciendo las cosas bien.

3.2.5.1 Validación

La diferencia entre el análisis de requerimientos (3.2.3) y la **validación**, es que el análisis se realiza sobre los requerimientos de sistema iniciales obtenidos de los involucrados, en cambio, la validación comprueba el documento final de requerimientos. Esta última revisión debe realizarse tanto por el equipo de desarrollo como por el cliente.

La IEEE define validación como:

> *"El proceso de evaluar un sistema o componente durante o al final del proceso de desarrollo para determinar si satisface los requerimientos especificados"*

Es decir, la validación permite confirmar que el sistema se haya implementado conforme a los requerimientos establecidos. Se puede realizar mediante 2 formas principalmente: pruebas de aceptación y pruebas de validación.

- **Pruebas de aceptación.** Las pruebas de aceptación involucran al usuario en la actividad de validación final para asegurar que el producto opere en la forma que el usuario realmente necesita.

- **Pruebas de validación.** Las actividades primarias de la validación son las actividades de prueba. Las pruebas de validación se realizan contra las especificaciones acordadas.

3.2.5.2 Verificación

La IEEE define verificación como:

> *El proceso de evaluar un sistema o componente para determinar si los productos de una determinada fase satisfacen las condiciones impuestas en el inicio de la fase.*

La verificación es un proceso continuo que nos ayuda a asegurar que cada paso del desarrollo sea correcto, y satisfaga las necesidades de la actividad siguiente. No es una actividad realizada sólo por el equipo de aseguramiento de la calidad si no que debe ser una filosofía de todos los grupos de trabajo, ya que es posible realizar verificación a todos los niveles: verificar los elementos de requerimientos, los elementos de diseño, los elementos de implementación, los elementos de prueba, etc.

Hay un punto de verificación muy importante al final del proceso de desarrollo que casi nunca se considera: que el producto haya demostrado trabajar dentro del ambiente de los usuarios y que los usuarios estén satisfechos con los resultados.

Los problemas más comunes en la **verificación** de requerimientos son referentes a la falta de cumplimiento de los estándares de calidad, la inadecuada especificación (mala redacción, ambigüedad), así como a conflictos entre los requerimientos que no fueron detectados durante el proceso de análisis.

Para la verificación de lo requerimientos es necesario considerar los atributos de calidad descritos por el estándar 830-1984 del IEEE y tratados en el punto 3.2.4.1 de este trabajo.

3.2.5.3 Guía de para la validación y especificación de requerimientos

Algunas preguntas útiles para la validación y verificación de requerimientos se presentan a continuación:

GUÍA PARA LA VALIDACIÓN Y VERIFICACIÓN DE REQUERIMIENTOS

- ¿Falta alguna información necesaria de los requerimientos? ¿Están completos?
- ¿Los requerimientos se encuentran dentro del alcance del proyecto?
- ¿Cada requerimiento se puede verificar de alguna manera? ¿Se puede probar?
- ¿Todos los requerimientos son escritos en un consistente y apropiado nivel de detalle?
- ¿Se ha verificado la gramática y ortografía de los documentos de requerimientos?
- ¿Son correctas todas las referencias a otros requerimientos? ¿Existen y se encuentran especificados?
- ¿Está indicada la prioridad de implementación de cada requerimiento?
- ¿Han sido definidos los algoritmos intrínsecos a los requerimientos funcionales?
- ¿Las especificaciones incluyen todas las necesidades del sistema y los usuarios?
- ¿Está documentado todo el comportamiento de las condiciones de error?
- ¿Algún requerimiento está duplicado o entra en conflicto con otro requerimiento?
- ¿Cada requerimiento está escrito en un lenguaje claro, conciso y sin ambigüedad?
- ¿Pueden ser implementados todos los requerimientos dentro de las restricciones conocidas?
- ¿Cada requerimiento es identificado de manera única?
- ¿Cada requerimiento funcional de software puede ser trazado a los requerimientos de alto nivel? ¿Existe posibilidad de rastreo? ¿Es posible saber de dónde derivó el requerimiento?
- ¿Los requerimientos representan aspectos de diseño o soluciones de implementación?

3.2.6 Administración

La administración de requerimientos consiste en organizar y mantener la información relacionada con los requerimientos a través de todo el ciclo de vida de desarrollo. Los principales aspectos de la administración de requerimientos son manejar: los cambios en los requerimientos, las relaciones entre requerimientos y sus dependencias.

Los problemas más comunes relacionados con la administración de requerimientos son: 1) los requerimientos no siempre son obvios y provienen de diversas fuentes, 2) no siempre son fáciles de expresar en palabras, 3) hay muchos tipos diferentes de requerimientos en diferentes niveles de detalle, 4) el número de requerimientos puede llegar a ser incontrolable si no se administra, 5) los requerimientos se relacionan unos con otros y se relacionan con otros artefactos del proceso de ingeniería de software, 6) los requerimientos cambian.

Existe una definición muy conocida de Administración de Requerimientos de Rational Software: *"Enfoque sistemático para obtener, organizar y documentar los requerimientos de un sistema, y establecer y mantener un acuerdo entre los clientes y el equipo de desarrollo del proyecto, sobre los cambios en los requerimientos del sistema".*

No obstante como se mencionó en el 3.1.2.2, existe una diferencia entre Ingeniería de Requerimientos y Administración de Requerimientos, bajo el enfoque de este trabajo, la administración es una actividad de la ingeniería de requerimientos, y su objetivo es:

> *Organizar y controlar la información y productos relacionados con los requerimientos, sus cambios y sus dependencias con otros a través de todo el ciclo de vida de desarrollo.*

3.2.6.1 Cambios en los requerimientos

Es un hecho que los requerimientos cambian. Esto no sólo significa más o menos tiempo de implementación sino que los requerimientos pueden tener impacto en otros.

Por ello es recomendable que el documento de especificación de requerimientos sea firmado por el cliente y por el equipo de desarrollo. La especificación se convierte en un contrato. Las peticiones de cambios en los requerimientos una vez que se ha finalizado la especificación no se negarán, pero el cliente debe saber que cada cambio posteriori significa un ampliación o modificación del alcance, y por lo tanto puede aumentar los costos y prolongar los tiempos. También es necesario usar ligas de rastreo para representar las dependencias entre los requerimientos y otros artefactos.

Existen varias razones, externas o internas, que están fuera del control de los desarrolladores y los usuarios. En los factores externos no se puede controlar ni prevenir el cambio pero sí controlar. Ejemplos de factores externos son: cambios en la economía, en las regulaciones gubernamentales, en las preferencias del mercado o en la tecnología. Ejemplos de factores internos: fallas en el proceso de obtención de requerimientos, por no realizar las preguntas correctas a las personas clave.

Las peticiones de cambio pueden ser oficiales, por ejemplo: requisiciones del usuario realizadas por los canales de comunicación apropiados; o extra-oficiales ("coladas"), como por ejemplo mejoras mencionadas por otros distribuidores, peticiones que los usuarios solicitan directamente a los programadores, características del hardware, errores, reacciones de los competidores o incluso funcionalidad añadida por los propios programadores.

Un aspecto técnico relacionado con el control del cambios es la administración de la configuración. Los beneficios de un proceso de administración de requerimientos basado en administración de configuración son: prevenir cambios no autorizados y riesgosos; preservar las revisiones de los documentos; facilitar la recuperación de versiones anteriores; y prevenir actualizaciones simultáneas de un mismo documento.

A continuación se presenta una lista de verificación para evaluar algunas implicaciones de un cambio, propuesta por Karl Wiegers:

GUÍA PARA EVALUAR E IMPACTO DE LOS CAMBIOS

- ¿Algún requerimiento de la línea base se ve afectado por el cambio?
- ¿Otros cambios pendientes entran en conflicto con el nuevo cambio?
- ¿Cuáles son las consecuencias técnicas o de negocio de no hacer el cambio?
- ¿Habrá efectos colaterales o riesgos de hacer el cambio?
- ¿El cambio afectará el desempeño o los atributos de calidad de otros requerimientos?
- ¿Es factible el cambio con los recursos disponibles?
- ¿Cómo afectará el cambio a la secuencia, dependencia, esfuerzo o duración de las tareas plasmadas en el plan?
- ¿Cuánto esfuerzo que ha sido invertido en el proyecto se perderá si el cambio se acepta?

Los principales elementos de software que se pueden ver afectados por un cambio propuesto son: interfaces de usuario, reportes, bases de datos o archivos, componentes de diseño, código fuente, casos de prueba, librerías, componentes de hardware, planes de administración del proyecto, planes de prueba, planes de administración de la configuración, entre otros.

Una vez que se han identificado los posibles componentes que se ven impactados por un cambio, es posible estimar el esfuerzo requerido para realizarlo. Una técnica es listar las actividades para realizar el cambio y asignarles las horas estimadas correspondientes. Ejemplos de dichas actividades son: actualización de las especificaciones de requerimientos, modificar componentes de diseño, desarrollar nuevo código fuente, modificar planes, desarrollar nuevos reportes, escribir nuevos casos de prueba, entre otras.

Los cambios son inevitables, para controlarlos se recomienda contar con un "comité" que evalúe su impacto y los apruebe o rechace, y así evitar peticiones "extra-oficiales".

3.2.6.2 Rastreabilidad

La IEEE define la rastreabilidad como:

> *El grado en el cual puede establecerse una relación entre dos o más productos del proceso de desarrollo, especialmente cuando los productos tienen una relación predecesor/ sucesor o principal/subordinado con otro.*
>
> *El grado en el que cada elemento en un producto del desarrollo de software establece su razón de existencia.*

El rastreo de requerimientos implica definir ligas lógicas entre requerimientos funcionales individuales y otros elementos del sistema, como los casos de uso, las reglas del negocio, el diseño, el código fuente y otros artefactos. Permite determinar cuáles componentes se tendrán que modificar si se implementa un cambio en un requerimiento específico.

La relación de rastreo puede ser:

- "Rastreado a"
- "Rastreado de"

El rastreo es una técnica efectiva que apoya la actividad de verificación. Las herramientas de rastreo permiten inspeccionar las relaciones de dependencia para asegurar que no existan omisiones; pero por sí solas no hacen todo el proceso. El punto clave es utilizar el rastreo para ayudar a comprender las relaciones entre los artefactos del proyecto y evaluar el impacto de los cambios.

Una técnica simple para manejar el rastreo (o trazabilidad) es mediante una matriz, en la que las columnas y renglones contienen el ID y/o el nombre del requerimiento. En la intersección de éstos, se indica la dependencia "de" o "hacia" los requerimientos.

Además es necesario asociar información a cada requerimiento sobre el identificador único (ID) del requerimiento, la fuente del requerimiento, un breve resumen del requerimiento, el archivo del texto completo del requerimiento, apuntadores a los requerimientos relacionados, así como la que se mencionó en el análisis de requerimientos (3.2.3.2).

3.2.6.2.1 Matriz de atributos

La matriz de atributos contiene la información asociada a cada requerimiento, por ejemplo: prioridad, estado y riesgo. Además puede mostrar los requerimientos derivados, por ejemplo, el requerimiento de sistema "2", deriva los requerimientos de software "2.1, 2.2 y 2.3". Esta derivación o dependencia de requerimientos se denomina también árbol de rastreo.

ID del Req.	Resumen	Prioridad	Estado	Riesgo	Fuente	Asignado a	Versión en la que se incorporará
Req. 1							
Req. 2							
▪ Req. 2.1							
▪ Req. 2.2							
▪ Req. 2.3							
Req. n							

3.2.6.2.2 Matriz de rastreo

La matriz de rastreo o de trazabilidad muestra la derivación de un requerimiento de alto nivel en otros. Por ejemplo un requerimiento de usuario da lugar a varios requerimientos de sistema, los cuales a su vez dan origen a requerimientos de software y de prueba. Por ejemplo:

<table>
<tr><td rowspan="6">Req. de Usuario 1</td><td rowspan="2">Req. de Sistema 1</td><td rowspan="2">Req. de SW 1</td><td>Req. de Prueba 1</td></tr>
<tr><td>Req. de Prueba 2</td></tr>
<tr><td>Req. de SW 2</td><td>Req. de Prueba 3</td></tr>
<tr><td rowspan="3">Req. de Sistema 2</td><td>Req. de SW 3</td><td rowspan="3">Req. de prueba 4</td></tr>
<tr><td>Req. de SW 4</td></tr>
<tr><td>Req. de SW 5</td></tr>
<tr><td>Req. de Sistema 3</td><td>Req. de SW 6</td><td></td></tr>
<tr><td rowspan="2">Req. de Usuario 2</td><td>Req. de Sistema 4</td><td></td><td></td></tr>
<tr><td>Req. de Sistema 5</td><td></td><td></td></tr>
<tr><td>Req. de Usuario n</td><td></td><td></td><td></td></tr>
</table>

Otro tipo de matriz de rastreo muestra la relación entre dos requerimientos, los cuales pueden ser del mismo o de distinto tipo, por ejemplo: requerimientos de software vs. requerimientos de usuario. En una columna de la matriz se enlistan los requerimientos de un determinado tipo y se confrontan con los requerimientos presentados como columnas.

	Req. De Usuario 1	Req. De Usuario 2	Req. De Usuario 3	Req. De Usuario 4	Req. De Usuario 5	Req. De Usuario 6	Req. De Usuario n
Req. de SW 1							
Req. de SW 2							
▪ Req. SW 2.1							
▪ Req. SW 2.2							
▪ Req. SW 2.3							
Req. De SW n							

Lo que se indica en esta matriz es que el "Requerimiento de Usuario 5" es trazado hacia el "Requerimiento de Software 2.1", por lo tanto existe una dependencia entre ambos requerimientos. Si cambia algo del requerimiento de usuario 5, habrá que revisar el impacto sobre el requerimiento de software 2.1.

Las herramientas de software para la Administración de Requerimientos, generan estas matrices automáticamente, a partir de la documentación asociada, lo que facilita la gestión y mantiene actualizada la información. No obstante, se pueden mantener matrices sencillas en hojas de cálculo.

3.2.6.3 Políticas para la administración de requerimientos

Aspectos clave para una efectiva administración de requerimientos incluyen: mantener un claro enunciado de los requerimientos, especificar sus atributos para cada tipo de requerimientos, así como su relación con otros requerimientos y otros artefactos del proyecto. Existe por lo tanto, una estrecha relación con todas las demás actividades de la Ingeniería de Requerimientos.

- **Organizar los requerimientos**
  - Categorizar los requerimientos y documentos por tipo.
  - Especificar los atributos (prioridad, nivel de dificultad, estatus, etc.).
  - Utilizar y adaptar las plantillas existentes.
- **Comunicar los requerimientos** (Para mantener a todos informados y reducir la ambigüedad)
  - Tener los requerimientos en una base de datos.
  - Generar reportes del proyecto.
  - Tener los requerimientos disponibles para que "todos" los consulten.
- **Manejar los cambios en los requerimientos** (Para asegurar la calidad y consistencia del proyecto)
  - Identificar y mantener las ligas (rastreo).
  - Medir el impacto de los cambios.
  - Seguir y documentar los cambios (quién, qué, cuándo, por qué).

3.2.6.4 Herramientas de software para la administración de requerimientos

El seleccionar una herramienta depende de muchos factores como:

- Las plataformas y el tipo de herramientas que se usan en la organización para lograr compatibilidad e interacción.
- El presupuesto disponible.
- El tamaño de los sistemas que se desarrollan.
- La estabilidad de las empresas proveedoras de la herramienta.

Una herramienta puede ir desde una hoja de cálculo, hasta aplicaciones más completas y especializadas como:

- Requisite Pro de Rational Software (www.rational.com)
- DOORS de Telelogic (www.telelogic.com)
- CaliberRM de StarBase (www.starbase.com)
- Analyst Pro de Analyst Tool (www.analysttool.com).

3.3 Los requerimientos como factor de calidad en el desarrollo de software

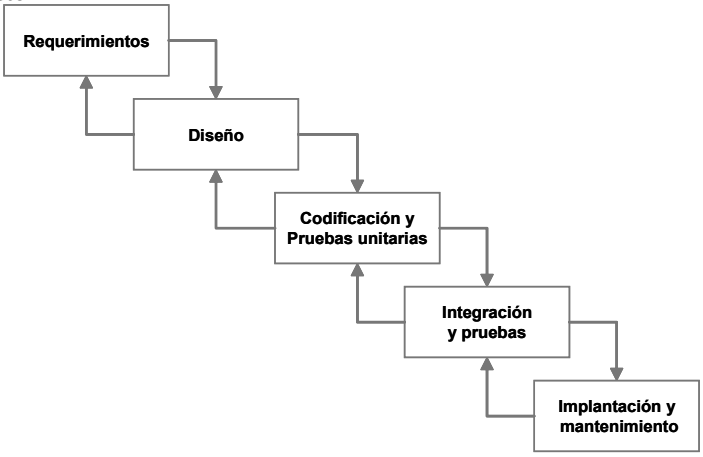
3.3.1 Los requerimientos en el ciclo de vida de desarrollo de software

Si no hay algún requerimiento, no hay nada por construir. Todos los procesos y ciclos de desarrollo de software consideran alguna actividad relacionada con la obtención y especificación de requerimientos. A continuación se describen los procesos de desarrollo más conocidos, resaltando la fase o actividad enfocada a los requerimientos.

3.3.1.1 Modelo en cascada

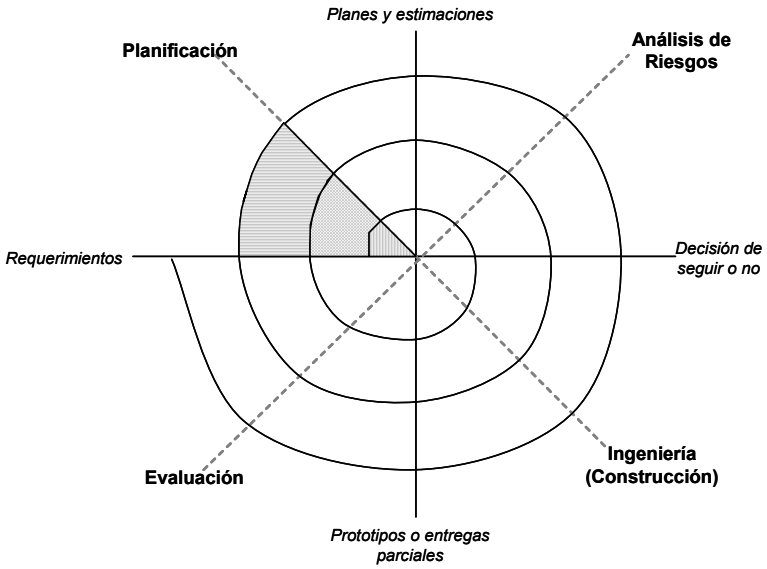
En el modelo de cascada, las actividades de software se dan de forma secuencial. Cada etapa trabaja sobre las actividades y los productos de la etapa anterior. Se establecen puntos de revisión al final de cada fase. Este modelo ha sido utilizado ampliamente y durante las pasadas dos décadas; no obstante actualmente no se utiliza completamente por el riesgo que implica, el continuo cambio en los requerimientos, el tamaño de los sistemas, la detección tardía de errores u omisiones y la dinámica de las organizaciones.

La fase de requerimientos es de extrema importancia y es estrictamente necesaria para pasar al diseño. Durante ésta se escribe el documento de especificación de requerimientos cuyo propósito es: contener una descripción completa de QUÉ hará el software sin describir CÓMO lo hará; es decir, contener una descripción completa del comportamiento externo del producto pero sin información de la estructura interna del producto.



3.3.1.2 Modelo en espiral

El modelo en espiral considera la gestión del riesgo y el desarrollo incremental. Inicialmente empieza con planeación de requerimientos y validación de conceptos, seguido de uno o más prototipos que ayudan a confirmar tempranamente la comprensión de los requerimientos del sistema. Su principal ventaja es la posibilidad de una continua retroalimentación con los clientes y usuarios.

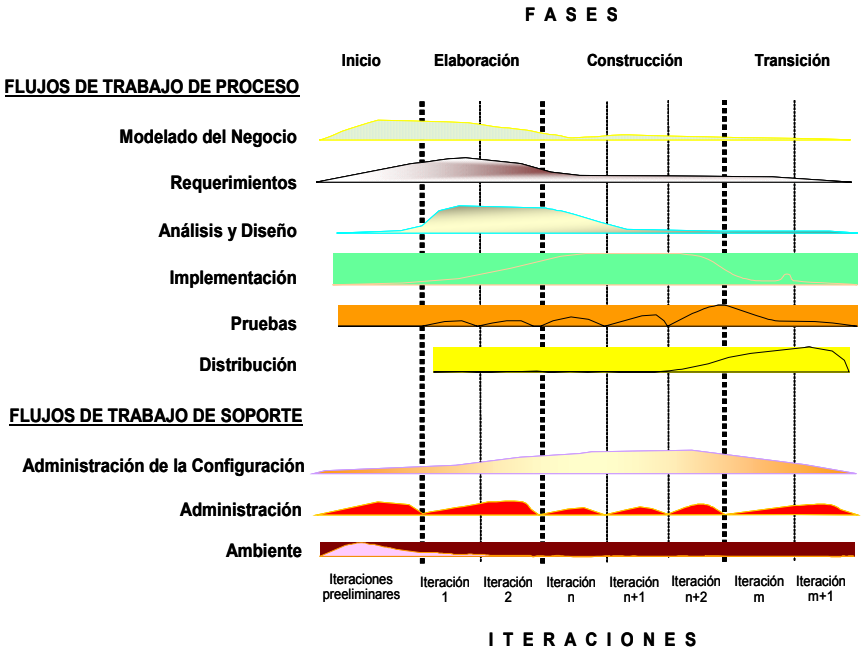


Los pasos a seguir en este modelo son:

- **Planificación**: Se determinan los objetivos, alternativas y restricciones.
- **Análisis de riesgos**: Se identifican los riesgos, analizando alternativas y resolución de los mismos.
- **Ingeniería**: Se desarrolla el producto del siguiente nivel.
- **Evaluación del cliente**: Se evalúan los resultados del paso anterior.

3.3.1.3 Proceso unificado de desarrollo

El proceso unificado es un modelo que combina algunos aspectos del modelo en cascada y del modelo en espiral, además de un enfoque iterativo propuesto inicialmente por Kruchten. Actualmente este modelo ha evolucionado a los que se conoce como Proceso Unificado, desarrollado principalmente por Ivar Jacobson en Ericsson y luego en Rational.



El proceso unificado considera los siguientes elementos:

- **Fases.** Son las diversas etapas que se dan a lo largo del tiempo del proyecto. inicio, elaboración, construcción y transición.
- **Flujos de trabajo**. Las actividades son organizadas en flujos de trabajo. Cada flujo es un conjunto de actividades relacionadas para obtener un producto.
- **Iteraciones**. Dentro de cada fase, en el proyecto se realizan iteraciones. Una iteración es una secuencia de actividades planeadas y con criterios de evaluación que da como resultado algún producto.

Durante la fase de INICIO, los analistas identifican la mayoría de los casos de uso para delimitar el sistema y el alcance del proyecto y para detallar los más importantes (menos del 10%)

Durante la fase de ELABORACIÓN, los analistas capturan la mayoría de los requisitos restantes para que los desarrolladores puedan estimar el esfuerzo. El objetivo es haber capturado un 80% de los requisitos y haber descrito la mayoría de los casos de uso al final de esta fase. (Sólo entre un 5 y 10% debería estar implementado)

Los requisitos restantes se capturan (e implementan) durante la fase de CONSTRUCCIÓN.

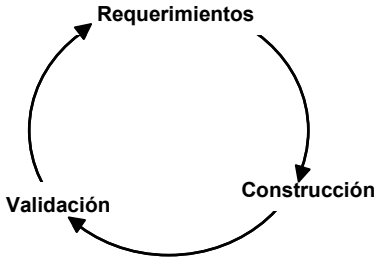
Casi no hay captura de requisitos en la fase de TRANSICIÓN, a menos que haya requisitos que cambien.

Este modelo permite tener un sistema con un funcionamiento parcial en una fase inicial y que los usuarios y otros interesados hagan sugerencias sobre él o señalen requisitos que se nos pueden haber escapado. El plan –presupuesto y calendario– aún no es inamovible, de forma que los desarrolladores pueden incorporar revisiones con más facilidad. En el modelo unidireccional en cascada, los usuarios no ven un sistema funcionando hasta la integración y la prueba. En este momento los cambios, incluso aquellos que son valiosos o parecen ser pequeños, impactan de manera importante. Por lo tanto, el ciclo de vida iterativo hace más sencillo a los clientes el observar la necesidad de modificaciones o requisitos adicionales y facilita a los desarrolladores el trabajo.

Como se puede observar en este proceso existe un flujo de trabajo enfocado a los requerimientos, el cual se realiza principalmente en las fases de inicio y elaboración.

3.3.1.4 Desarrollo por prototipos

Como se mencionó anteriormente, un prototipo es un programa que implementa algunas partes de los requerimientos del sistema. El desarrollo por prototipos es una estrategia de desarrollo de software que permite desarrollar rápidamente alguna parte del sistema de software para que los usuarios evalúen y recomienden cambios a la definición de requerimientos.



Este modelo consta de 3 actividades principales:

- **Requerimientos.** Se obtienen las características generales de lo que los usuarios solicitan.
- **Construcción.** Se implementa una parte de las características identificadas.
- **Validación.** Se presenta a los usuarios el prototipo para obtener retroalimentación y en su caso, nuevos requerimientos.

El paradigma de creación de prototipos puede ser cerrado o abierto. El enfoque cerrado se denomina a menudo prototipo desechable. Este prototipo sirve como demostración de los requisitos; después se desecha. Un enfoque abierto, prototipo evolutivo, emplea el prototipo como primera parte de una actividad de análisis.

Los prototipos según Alan M. Davis pueden ser: prospectos, evolutivos u operacionales, verticales, horizontales, de interfaces de usuario, algorítmicos. Para efectos de la obtención de requerimientos, generalmente se opta por un prototipo: prospecto, horizontal, de interfaces de usuario.

3.3.1.5 Programación extrema

Programación extrema (XP) es una disciplina de desarrollo de software desarrollada por Kent Beck en 1996. Se basa en cuatro valores:

- **Comunicación.** Se establece una continua comunicación entre el cliente y el equipo de desarrollo, mediante la presencia de éste en el ciclo de vida de desarrollo. El cliente decide qué construir y en qué orden.
- **Simplicidad.** El equipo de desarrollo propicia la reutilización de componentes simples.
- **Retroalimentación.** Muchas versiones y pruebas unitarias continuas constituyen los mecanismos de retroalimentación.
- **Coraje.** Hacer las cosas bien, aún cuando no sea lo más importante. Implica ser honesto en que podemos y que no podemos hacer.

Para apoyar estos valores, la programación extrema lleva a cabo las siguientes 12 prácticas:

- **Planeación audaz.** Determinar las características más prioritarias para la próxima versión.
- **Pequeñas versiones.** Liberar el software frecuentemente al usuario en pequeñas versiones incrementales.
- **Metáfora.** Una simple descripción de cómo trabaja el sistema.
- **Diseño simple.** Mantener diseños simples para lograr código simple. Remover continuamente la complejidad.
- **Pruebas.** El cliente escribe pruebas para los diversos escenarios. Los programadores escriben pruebas para probar cualquier cosa que pueda fallar en el código. Las pruebas son escritas antes de que el código sea escrito.
- **Revisión del código.** El objetivo es remover la duplicidad y la complejidad del código.
- **Programación en pares.** Equipos de dos programadores en una sola computadora desarrollan todo el código. Uno escribe el código, mientras el otro lo revisa para que sea correcto y comprensible.

- **Propiedad colectiva.** Todos y cada uno son propietarios de todo el código. Esto significa que cualquiera pueden cambiar el código en cualquier momento.
- **Integración continua.** Construir e integrar el código varias veces en un día.
- **Cuarenta horas a la semana.** Los programadores no pueden trabajar eficientemente si están cansados.
- **Cliente en el sitio.** Un cliente o usuario trabaja en el desarrollo de tiempo completo para ayudar a definir el sistema, escribir pruebas y responder preguntas.
- **Estándares de codificación.** Loa programadores adoptan un estándar de codificación consistente.

Julio César Sampaio, identifica varios inconvenientes en el enfoque de XP, relacionados con requerimientos, los más relevantes son:

- El negocio es representado sólo por un usuario.
- La falta de consideración de los requerimientos no funcionales del punto de vista del negocio.

3.3.2 Relación con modelos de madurez de procesos

3.3.2.1 CMM (Capability Maturity Model)

El Modelo de Madurez de Capacidades ha surgido como un estándar para un proceso de calidad más profundo y comprensible que otros estándares, por lo que está siendo ampliamente adoptado por la industria del software. CMM orienta a los desarrolladores y a la organización a mejorar su proceso de software con el objetivo de lograr repetición, control y medición. El aspecto de administración de requerimientos es fundamental en CMM. A continuación se presenta una tabla con el resumen de los niveles y las áreas clave de proceso de CMM:

	NIVEL	AREAS CLAVE DEL PROCESO
Inicial	Proceso ad hoc, a veces caótico; el éxito depende de héroes y esfuerzos individuales.	-
Repetible	Administración de proyectos básica para dar seguimiento a la funcionalidad de la aplicación, y al costo y tiempo del proyecto.	**Administración de requerimientos** Planeación de proyectos de software Supervisión y seguimiento de proyectos de software Administración de subcontratación de software Aseguramiento de la calidad del software Administración de la configuración del software
Definido	El proceso de administración e ingeniería es documentado, estandarizado e integrado. Todos los proyectos usan una versión del proceso aprobada y adaptada.	Enfoque en el proceso de la organización Definición del proceso de la organización Programa de capacitación Administración del software integrada **Ingeniería del producto de software** Coordinación entre grupos Revisiones pares
Administrado	Se colecta métricas detalladas del proceso de software y de su calidad. Tanto el proceso como los productos de software son comprendidos y controlados.	Administración cuantitativa del proceso Administración de la calidad del software
Optimizado	Mejoramiento continuo del proceso es posible por el uso de métricas y por pilotear ideas y tecnologías nuevas.	Prevención de defectos Administración del cambio tecnológico Administración del cambio del proceso

Administración de requerimientos es una actividad clave del proceso enfocada a asegurar que el producto de software satisfaga las necesidades del usuario tanto en funcionalidad como en calidad.

La primer área clave de proceso que debe ser considerada para pasar de nivel 1 a nivel 2 es la **administración de requerimientos;** representa una parte integral en el desarrollo de software.

Administración de requerimientos en CMM nivel 2

> *El propósito de la administración de requerimientos es establecer una comprensión común entre el cliente y el equipo de desarrollo, de los requerimientos del usuario.*

Esta comprensión común sirve como base para el acuerdo entre el cliente y el equipo de desarrollo, como tal, es el documento central que define y controla la actividad a seguir. Los requerimientos son usados para establecer líneas base. Los lineamientos de CMM especifican que todas las actividades, planes, calendarios y los artefactos de software serán desarrollados y modificados como sea necesario para ser consistentes con los requerimientos. De esta manera CMM guía a la organización a que la actividad técnica de requerimientos sea consistente con los planes y actividades. Para ello este proceso debe ser documentado y revisado por todos los grupos afectados.

La especificación de requerimientos de software sirve como documento central del proyecto. Los requerimientos incluyen tanto requerimientos técnicos como no técnicos. Los criterios de aceptación también serán establecidos y documentados.

Para cumplir estos objetivos se deben proveer los recursos necesarios para la administración de requerimientos. Los miembros del grupo de ingeniería de software y otros deben ser capacitados en las actividades de la administración de requerimientos.

CMM especifica que los requerimientos deben ser manejados y controlados y deben servir como base para los planes, productos y actividades de trabajo del software. Los cambios deben ser revisados, incorporados en los planes y se debe evaluar el impacto para negociarlo con los grupos involucrados.

CMM sugiere métricas para verificar la implementación de las actividades de la administración de requerimientos, por ejemplo: el estatus de cada requerimiento, el número acumulado de cambios, total de cambios abiertos, propuestos, aprobados e incorporados en una línea base. El estatus de cada requerimientos en el ciclo de vida y el total de requerimientos por categoría.

Dentro de CMM la administración de requerimientos no es simplemente un proceso de documentación, sino una actividad central del proceso de desarrollo. Si bien está explícitamente ubicado en el nivel 2, está guarda relación con todos los niveles y con la mayoría de las áreas clave de proceso.

En CMM también se menciona la relación de los documentos y la administración de la configuración, así como el rastreo.

En resumen CMM explícitamente sugiere lo siguiente:

- *Los requerimientos de software deben ser documentados.*
- *Los requerimientos de software deben ser controlados para establecer una línea base para la ingeniería y la administración.*
- *Los miembros del equipo deben ser capacitados para realizar las actividades de la administración de requerimientos.*
- *Deben establecerse métricas que incluyan el estado de cada requerimiento.*

Administración de requerimientos en CMM nivel 3

El proceso de administración de requerimientos aparece virtualmente en todos los niveles del modelo y dentro de varias áreas clave de proceso.

- *Los requerimientos de software son desarrollados, mantenidos, documentados y verificados por el análisis sistemático de los requerimientos de acuerdo al proceso de software definido al proyecto.*
- *Los requerimientos deben ser analizados para determinar si son completos, consistentes y se pueden probar.*
- *Los documentos de requerimientos de software deben ser considerados en la administración de la configuración.*

Rastreo de requerimientos

La consistencia es mantenida a través de los productos de trabajo de software, incluyendo los planes, las descripciones de procesos, los requerimientos asignados, el diseño de software, el código, los planes de pruebas y los procedimientos de pruebas.

- *Los requerimientos de software, el diseño, el código y los casos de prueba son derivados de su fuente y hacia los productos de la actividad subsecuente.*

3.3.2.2 SPICE (ISO 15504)

Este estándar es un modelo de referencia que describe los procesos que una organización puede realizar ya sea para adquirir, proveer, desarrollar, operar, evolucionar y soportar algún software, así como los atributos que caracterizan la capacidad de cada uno de los procesos.

Los procesos se agrupan en tres ciclos de vida:

- Primario.
- Organizacional.
- Soporte.

Estos contienen cinco categorías de proceso, las cuales son:

- CUS. Categoría de Procesos Cliente-Proveedor.
- ENG. Categoría de Procesos de Ingeniería.
- SUP. Categoría de Procesos de Soporte.
- MAN. Categoría de Procesos de Administración.
- ORG. Categoría de Procesos de Organización.

Principalmente en tres categorías (CUS, ENG y SUP) se encuentran aspectos relacionados con la ingeniería de requerimientos.

4 MARCO METODOLÓGICO

4.1 Variables

Las variables planteadas inicialmente en el marco problemático de este trabajo son las siguientes:

Independientes	Dependientes
Actividades, técnicas y herramientas de la Ingeniería de Requerimientos	▪ Mayor control del proyecto ▪ Mejor estimación de la dimensión del proyecto ▪ Mayor calidad del software ▪ Mejor comunicación con clientes y usuarios ▪ Desarrollar sistemas que realicen la funcionalidad requerida por el usuario ▪ Mejorar la comunicación entre los equipos involucrados en el desarrollo ▪ Alcanzar un mayor nivel de competitividad en el desarrollo de software ▪ Cubrir los principales aspectos de los modelos de madurez de procesos

4.2 Variables de control

En el presente trabajo no se incluyen variables de control, de tipo interviniente, ni distorisionante, debido a que no se identificaron otros factores considerablemente importantes en la investigación para tratarlos de esta forma.

4.3 Hipótesis definitiva

Tomando como base el problema planteado inicialmente en este trabajo, la hipótesis preliminar y el desarrollo del marco teórico y conceptual, se presenta a continuación la hipótesis definitiva:

La realización de las actividades de la ingeniería de requerimientos, mediante el uso de técnicas y herramientas en un proyecto de desarrollo de software permite:

- **Especificar de manera clara, precisa, y completa los requerimientos de un sistema informático**
- **Desarrollar sistemas con la funcionalidad y características requeridas por el usuario**
- **Planear y realizar estimaciones más precisas de los proyectos**
- **Lograr una mayor calidad del software**
- **Cumplir con los lineamientos de los modelos internacionales de madurez de procesos**

4.4 Definición del universo

En este tema de investigación, el universo es difícil de especificar, debido a que no se tiene una delimitación precisa y cuantitativa de las personas encuestadas. Además no es el objetivo realizar pruebas estadísticas para aprobar o desaprobar la hipótesis, sino obtener opiniones y juicios de valor.

Cabe mencionar sin embargo, que dicho universo estuvo integrado por personas involucradas de alguna manera con el desarrollo de software en México, tanto de organizaciones públicas como privadas., como son: gerentes de sistemas, líderes de proyecto, consultores informáticos, investigadores, analistas, programadores y probadores de software.

4.5 Determinación de la muestra

Se utilizó una muestra no probabilística, denominada de juicio o intencionada, en la que se seleccionaron 14 personas, que laboran en diversas organizaciones que desarrollan sistemas sin ninguna tendencia en particular. De las respuestas y opiniones obtenidas se realizó un análisis y se emitieron conclusiones.

4.6 Instrumento

El instrumento de investigación seleccionado fue el cuestionario, el cual se aplicó vía Internet. Se seleccionó este instrumento por permitir obtener las respuestas de varias personas a la vez desde sus lugares de trabajo, considerando que todos ellos cuentan con el perfil para contestarlo y enviarlo por Internet.

El cuestionario estuvo integrado por 16 preguntas, la mayoría de éstas cerradas, para facilitar y precisar las respuestas de los encuestados. No obstante se permitió completar con justificaciones y comentarios adicionales.

4.7 Costo de la investigación

A continuación se resumen diversos costos relacionados con la investigación:

Concepto	Costo
Recursos Humanos 10 meses con sueldo Mensual de $ 17,500	175,000.00
Transporte a los sitios de recopilación de información	4,500.00
Material de investigación (libros, revistas, fotocopiado)	2,800.00
Asistencia a eventos y cursos	6,000.00
Uso de equipo de cómputo	3,200.00
Papelería y artículos diversos de oficina	2,500.00
Impresión y terminado	3,000.00
Comunicación (teléfono, fax, Internet)	2,800.00
Otros varios	1,300.00
Total	**201,100.00**

4.8 Cuestionario definitivo

Objetivo: Comprobar si el uso de lineamientos, técnicas y herramientas de la Ingeniería de Requerimientos, contribuyen al éxito de los proyectos de desarrollo de software.

1	**¿Considera que la ingeniería de requerimientos es una actividad clave dentro del ciclo de vida de desarrollo de software?** ❍ Sí, es la más importante ❍ Sí, es importante ❍ No es importante El objetivo de esta pregunta es conocer el nivel de la importancia que se da a las actividades de ingeniería de requerimientos.
2	**¿El software que desarrolla cumple con todas las características que el usuario necesita?** ❍ Sí, siempre ❍ Sí, casi siempre ❍ Sí, Solo a veces ❍ Nunca Esta pregunta se hizo con la finalidad de que el encuestado reflexione acerca de si sus sistemas de software desarrollados, satisfacen a sus clientes y usuarios, ya que esta es una tarea clave en la ingeniería de requerimientos.
3	**¿Los proyectos de desarrollo de software se terminan a tiempo y con los recursos inicialmente estimados?** ❍ Sí, siempre ❍ Sí, a veces ❍ Casi Nunca ❍ Nunca Con esta pregunta se trata de comprobar que las estimaciones iniciales de recursos en la mayoría de los casos se sobrepasan, siendo una de las causas principales la inadecuada ingeniería de requerimientos.

4	**Si conoce los modelos de madurez como CMM o SPICE ¿En qué nivel considera que se encuentra su organización respecto a alguno de estos?** Nivel ❑ de CMM (1-Inicial, 2-Repetible, 3-Definido, 4-Administrado, 5-Optimizado) Nivel ❑ de SPICE (0-Incompleto, 1-Realizado, 2-Administrado, 3-Establecido, 4-Predecible, 5-Optimizado) ❑ Desconozco los modelos y/o el nivel La finalidad de esta pregunta es por un lado comprobar si los modelos de madurez de procesos son conocidos en México, y en qué nivel creen encontrarse las organizaciones, ya que la ingeniería de requerimientos es fundamental en ambos modelos para garantizar la calidad del software.
5	**¿Qué técnicas utiliza y considera efectivas para la identificación y obtención de los requerimientos con el usuario?** ❑ Entrevistas ❑ Cuestionarios ❑ Mesas de Trabajo ❑ Lluvia de Ideas ❑ Juego de Roles ❑ Guía de preguntas y aspectos clave ❑ Otras,¿cuales? _____ El objetivo de esta pregunta es conocer las técnicas y herramientas que los encuestados utilizan para obtener requerimientos de los usuarios e involucrados.
6	**¿Realiza un análisis de los requerimientos con el fin de depurarlos, clasificarlos, ponderarlos y definir así claramente el alcance de la solución?** ❍ Sí, siempre ❍ A veces ❍ No

	El objetivo de esta pregunta es conocer si se realiza la actividad de análisis de requerimientos dentro del proceso de la organización del encuestado.
7	**Si realiza este análisis, ¿Utiliza alguna guía para este análisis de requerimientos?** ❍ Sí ❍ No En caso de ser afirmativa la pregunta anterior, el objetivo de ésta es conocer si se utilizan guías para realizar el análisis de requerimientos.
8	**¿Utiliza alguna guía de contenido estándar para la documentación/especificación de requerimientos?** ❍ Se tiene y se utiliza ❍ Se tiene pero no se utiliza ❍ No se tiene El objetivo de esta pregunta es conocer si se cuenta con guías, estándares o plantillas para especificar los requerimientos en un documento.
9	**¿Qué técnicas utiliza y considera efectivas para el modelado y especificación de requerimientos?** ❍ No se modelan ni especifican por escrito ❍ Se modelan utilizando: ❑ Lenguaje natural ❑ Casos de Uso ❑ Pseudocódigo ❑ Prototipos ❑ Análisis estructurado ❑ Otras, ¿cuáles? _____

	El objetivo de esta pregunta es conocer las técnicas y herramientas que los encuestados utilizan para especificar y modelar los requerimientos.
10	**¿Cómo considera que son las especificaciones de requerimientos que se realizan en su organización?** ❑ Claras ❑ Completas ❑ Concisas ❑ Comprensibles ❑ Simples ❑ Viables ❑ Flexibles ❑ Ambiguas ❑ Se pueden probar ❑ Se puede rastrear la dependencia de/con otras ❑ Con un identificador único ❑ Independientes de la implementación Esta pregunta se formuló con la finalidad de que el encuestado reflexione acerca de las características de los requerimientos que desarrolla.
11	**¿Realiza una validación y verificación de requerimientos para garantizar que sean correctos (que representen lo que el usuario necesita, que expresen lo que tengan que decir, que sean los que deben de ser, que se especifiquen de la manera correcta)?** ❍ Sí ❍ No El objetivo de esta pregunta es conocer si se realizan las actividades de validación y verificación de requerimientos dentro del proceso de la organización del encuestado.

12	**¿En promedio de qué tamaño son los sistemas o el software que se desarrolla en su organización?** ❍ Pequeños. Menos de: 10 requerimientos de sistema/casos de uso. Menos de 2,500 LOC ❍ Mediano. Entre 10 y 50 requerimientos de sistema/casos de uso. Entre 2,500 y 25,000 LOC ❍ Grande. Entre 50 y 200 requerimientos de sistema/casos de uso. Entre 25,000 y 100,000 LOC ❍ Muy Grande. Más de 200 requerimientos de sistema/casos de uso. Más de 100,000 LOC **Además, el software que desarrolla es de tipo:** ❍ Critico ❍ Administrativo/Gestión de Información ❍ Comercial Con esta pregunta se pretende clasificar a la organización del encuestado de acuerdo a la dimensión y tipo de los sistemas que desarrolla.
13	**¿Cuenta con políticas para la administración de requerimientos (organización, control y seguimiento de los cambios en los requerimientos)?** ❍ Sí ❍ No El objetivo de esta pregunta es conocer si se realiza la actividad de administración de requerimientos dentro del proceso de la organización del encuestado.
14	**¿Utiliza alguna herramienta de software para la administración de requerimientos?** ❍ Sí ❍ No La finalidad es conocer si se utilizan herramientas de software para administrar requerimientos

15	**¿Qué aspectos de la ingeniería de requerimientos considera muy importantes para una mayor calidad del software y para el éxito de los proyectos?** El objetivo de esta pregunta es conocer otros aspectos que los encuestados consideran importantes relacionados con la ingeniería de requerimientos que les han coadyuvado al éxito de sus proyectos.
16	**¿Considera las actividades de la ingeniería de requerimientos factor clave para el éxito o fracaso de los proyectos de desarrollo de software?** Esta pregunta nos permitirá conocer el juicio de los encuestados sobre la influencia de la ingeniería de requerimientos en el éxito o fracaso de los proyectos, y comprobar en gran medida la hipótesis del trabajo.

4.8.1 Matriz de respuestas

		Adrián Galindo Hernández	Alberto Cortés Ulloa	Alejandro Moreno Rivas	Alejandro Pérez Hernández	Antonio Malpica	Carlos Lozano H.	Diego H. Zavala GC	Edgar Mendoza Arreola	Javier Elizondo Campos	José de Jesús Hernández Suárez	Juan Carlos Segundo Elías	Marco Dorantes Martínez	Roberto Chávez Quintero	Sergio Cardoso
		Intermec Technologies de Mexico	iquatro	ING Comercial América	Qualitas Compañía de Seguros, S.A. de C.V.	Desarrollo Creativo	ING Comercial América	Hildebrando, SA de CV	DA, Organismo Integrado de Distribución S.A. de C.V.	ING Comercial América	Dirección de Sistemas-UNAM	SHCP-TESOFE	Microsoft Consulting	ING Comercial América	ISOFT Ingeniería de Software
1	Sí, es la más importante		✓	✓	✓			✓		✓	✓	✓			✓
	Sí, es importante	✓				✓	✓		✓				✓	✓	
	No es importante														
2	Sí, siempre												✓		
	Sí, casi siempre	✓	✓		✓	✓		✓		✓	✓				✓
	Sí, sólo a veces			✓			✓		✓			✓		✓	
	Nunca														
	Porque:		Porque está basado en un estudio previo y por contener muchas de las necesidades actuales de las organizaciones.	Cambios en el alcance del proyecto antes de poder terminarlo	Porque por lo general siempre se le consulta.	Es difícil, incluso para el usuario, determinar siempre lo que exactamente necesita.	No existe una definición adecuada de las necesidades del usuario		Porque con frecuencia el usuario no sabe lo que quiere	Debido a un énfasis en el análisis de requerimientos en conjunto con el usuario.	En ocasiones se da una mala comunicación con el usuario y a su vez una mala administración de los requerimientos.	El usuario no tiene pleno conocimiento de qué es lo que realmente quiere	Porque el usuario-cliente define los criterios de aceptación funcionales al inicio de cada ciclo de entrega	No siempre está bien especificado y documentado qué necesita el usuario.	

		Adrián Galindo Hernández	Alberto Cortés Ulloa	Alejandro Moreno Rivas	Alejandro Pérez Hernández	Antonio Malpica	Carlos Lozano H.	Diego H. Zavala GC	Edgar Mendoza Arreola	Javier Elizondo Campos	José de Jesús Hernández Suárez	Juan Carlos Segundo Elías	Marco Dorantes Martínez	Roberto Chávez Quintero	Sergio Cardoso
		Intermec Technologies de Mexico	iquatro	ING Comercial América	Qualitas Compañía de Seguros, S.A. de C.V.	Desarrollo Creativo	ING Comercial América	Hildebrando, SA de CV	DA, Organismo Integrado de Distribución S.A. de C.V.	ING Comercial América	Dirección de Sistemas-UNAM	SHCP-TESOFE	Microsoft Consulting	ING Comercial América	ISOFT Ingeniería de Software
3	Sí, siempre												✓		
	Sí, a veces		✓		✓	✓		✓				✓			✓
	Casi nunca	✓		✓			✓		✓	✓	✓			✓	
	Nunca														
	Porque:	Usuarios, cambios no previstos en un principio.	Porque en ocasiones existen algunas adaptacio-nes, aunque regular-mente son de diseño gráfico.	Mala definición desde el inicio, adecuacio-nes a medio proyecto, alcance diferente.	Algunas veces, nos salen imponderables, que son difíciles de solucionar inmediata-mente.	Sólo cuando se hace una correcta estimación y ambas partes convienen en ella.	Por una mala planeación de las cosas.			Es difícil debido a la falta de una metodología certera en el seguimiento a proyectos.	Mala estimación de tiempos con base a lo que real-mente se requiere desarrollar.	Siempre obtenemos nuevos requerimien-tos a destiempo.	Porque el tiempo y recursos se estiman continua-mente de acuerdo a las priori-dades y cambios de negocio y porque no estimamos más de dos meses de trabajo por ciclo de entrega.	Mala planeación, malos entendidos, ambigüedades falta de control y seguimiento.	Por lo subjetivo del software y la dificultad para evaluar el desempeño humano en su desarrollo.

		Adrián Galindo Hernández	Alberto Cortés Ulloa	Alejandro Moreno Rivas	Alejandro Pérez Hernández	Antonio Malpica	Carlos Lozano H.	Diego H. Zavala GC	Edgar Mendoza Arreola	Javier Elizondo Campos	José de Jesús Hernández Suárez	Juan Carlos Segundo Elías	Marco Dorantes Martínez	Roberto Chávez Quintero	Sergio Cardoso
		Intermec Technologies de Mexico	iquatro	ING Comercial América	Qualitas Compañía de Seguros, S.A. de C.V.	Desarrollo Creativo	ING Comercial América	Hildebrando, SA de CV	DA, Organismo Integrado de Distribución, S.A. de C.V.	ING Comercial América	Dirección de Sistemas-UNAM	SHCP-TESOFE	Microsoft Consulting	ING Comercial América	ISOFT Ingeniería de Software
4	Nivel de CMM		3					3			2		3		4
	Nivel de SPICE		4								2		4		
	Desconozco los modelos y/o el nivel	✓		✓	✓	✓	✓		✓	✓		✓		✓	
5	Lluvia de ideas	✓		✓						✓	✓	✓		✓	✓
	Entrevista	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
	Roles										✓	✓	✓		
	Cuestio-narios			✓	✓							✓			
	Guías	✓	✓			✓		✓			✓	✓		✓	✓
	Mesas		✓	✓		✓	✓		✓	✓	✓	✓	✓		✓
	Otras					Presentación de propuestas, revisión conjunta y retroalimentación. Prototipos.			Casos de Uso			Muestra de prototipos.	Especificación funcional por medio de pruebas de aceptación automatizadas.	Modelado de procesos y tareas.	

		Adrián Galindo Hernández	Alberto Cortés Ulloa	Alejandro Moreno Rivas	Alejandro Pérez Hernández	Antonio Malpica	Carlos Lozano H.	Diego H. Zavala GC	Edgar Mendoza Arreola	Javier Elizondo Campos	José de Jesús Hernández Suárez	Juan Carlos Segundo Elías	Marco Dorantes Martínez	Roberto Chávez Quintero	Sergio Cardoso
		Intermec Technologies de Mexico	iquatro	ING Comercial América	Qualitas Compañía de Seguros, S.A. de C.V.	Desarrollo Creativo	ING Comercial América	Hildebrando, SA de CV	DA, Organismo Integrado de Distribución S.A. de C.V.	ING Comercial América	Dirección de Sistemas-UNAM	SHCP-TESOFE	Microsoft Consulting	ING Comercial América	ISOFT Ingeniería de Software
6	Sí, siempre			✓		✓				✓		✓	✓		✓
	A veces	✓	✓		✓		✓	✓	✓					✓	
	No										✓				
	Por qué		Por falta de información			Si ya es bastante difícil llevar el proyecto a buen término haciéndolo, imagine usted si no.	Por la mala planeación				Por los tiempos tan cortos para desarrollo que demandan los proyectos en los que estoy involucrado.		El cliente los ordena por prioridad de negocio e implementamos modelos ejecutables para verificar aspectos clave.	Las prisas, la presión.	Optimización de tiempos y recursos
7	Sí		✓		✓			✓		✓		✓	✓		✓
	No	✓		✓		✓	✓		✓					✓	

		Adrián Galindo Hernández	Alberto Cortés Ulloa	Alejandro Moreno Rivas	Alejandro Pérez Hernández	Antonio Malpica	Carlos Lozano H.	Diego H. Zavala GC	Edgar Mendoza Arreola	Javier Elizondo Campos	José de Jesús Hernández Suárez	Juan Carlos Segundo Elías	Marco Dorantes Martínez	Roberto Chávez Quintero	Sergio Cardoso
		Intermec Technologies de Mexico	iquatro	ING Comercial América	Qualitas Compañía de Seguros, S.A. de C.V.	Desarrollo Creativo	ING Comercial América	Hildebrando, SA de CV	DA, Organismo Integrado de Distribución S.A. de C.V.	ING Comercial América	Dirección de Sistemas-UNAM	SHCP-TESOFE	Microsoft Consulting	ING Comercial América	ISOFT Ingeniería de Software
8	Se tiene y se utiliza							✓		✓	✓		✓		✓
	Se tiene pero no se utiliza														
	No se tiene	✓	✓	✓	✓	✓	✓		✓			✓		✓	
	Por qué	Nunca se ha hecho y hace falta una investigación	Falta de iniciativa						Finalmente no se utilizan estándares sino técnicas			Tenemos que adaptar-nos a la naturaleza del sistema	Casos de uso breves (como narraciones)		Base funda-mental para el desarrollo del proyecto.
9	Lenguaje natural	✓	✓	✓	✓	✓		✓				✓		▪	✓
	Casos de uso	✓	✓			✓			✓	✓	✓	✓	✓		
	Pseudo-codigo													✓	
	Prototipos	✓		✓	✓	✓		✓	✓	✓	✓	✓			✓
	Análisis estructurado			✓				✓			✓	✓			
	Otras									Metodo-logía de ingeniería de procesos		Diagramas de flujo sencillos	Pruebas funcio-nales automati-zadas		

		Adrián Galindo Hernández	Alberto Cortés Ulloa	Alejandro Moreno Rivas	Alejandro Pérez Hernández	Antonio Malpica	Carlos Lozano H.	Diego H. Zavala GC	Edgar Mendoza Arreola	Javier Elizondo Campos	José de Jesús Hernández Suárez	Juan Carlos Segundo Elías	Marco Dorantes Martínez	Roberto Chávez Quintero	Sergio Cardoso
		Intermec Technologies de Mexico	iquatro	ING Comercial América	Qualitas Compañía de Seguros, S.A. de C.V.	Desarrollo Creativo	ING Comercial América	Hildebrando, SA de CV	DA, Organismo Integrado de Distribución S.A. de C.V.	ING Comercial América	Dirección de Sistemas-UNAM	SHCP-TESOFE	Microsoft Consulting	ING Comercial América	ISOFT Ingeniería de Software
10	Claras				✓			✓			✓	✓	✓		
	Simples	✓	✓							✓		✓	✓		
	Probables				✓						✓	✓	✓		✓
	Identificador unico											✓	✓		
	Completas							✓		✓			✓		
	Viables				✓					✓		✓	✓		
	Concisas							✓		✓	✓	✓	✓		✓
	Flexibles		✓							✓		✓			
	Rastreables		✓							✓	✓	✓			✓
	Indepen-dientes		✓					✓					✓		
	Compren-sibles					✓		✓		✓		✓	✓		✓
	Ambiguas	✓		✓			✓		✓					✓	
11	Sí		✓		✓	✓	✓	✓		✓	✓	✓	✓	✓	✓
	No														

		Adrián Galindo Hernández	Alberto Cortés Ulloa	Alejandro Moreno Rivas	Alejandro Pérez Hernández	Antonio Malpica	Carlos Lozano H.	Diego H. Zavala GC	Edgar Mendoza Arreola	Javier Elizondo Campos	José de Jesús Hernández Suárez	Juan Carlos Segundo Elías	Marco Dorantes Martínez	Roberto Chávez Quintero	Sergio Cardoso
		Intermec Technologies de Mexico	iquatro	ING Comercial América	Qualitas Compañía de Seguros, S.A. de C.V.	Desarrollo Creativo	ING Comercial América	Hildebrando, SA de CV	DA, Organismo Integrado de Distribución S.A. de C.V.	ING Comercial América	Dirección de Sistemas-UNAM	SHCP-TESOFE	Microsoft Consulting	ING Comercial América	ISOFT Ingeniería de Software
12	Pequeños				✓		✓								
	Medianos	✓	✓			✓					✓				✓
	Grandes								✓	✓			✓		
	Muy Grandes			✓				✓				✓		✓	
	Crítico			✓			✓	✓		✓	✓	✓	✓		
	De gestión	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓
	Comercial					✓	✓				✓		✓		
	Comentarios								Sistemas con fines de comercialización masiva por lo que no se basan en requerimientos específicos de un solo cliente.		Sitios de Comercio Electrónico	El Sistema Integral de Admón. Financiera Federal es uno de los proyectos más importantes en el país	Es posible debido a las técnicas de administración de la complejidad en las prácticas de los métodos ágiles		

		Adrián Galindo Hernández	Alberto Cortés Ulloa	Alejandro Moreno Rivas	Alejandro Pérez Hernández	Antonio Malpica	Carlos Lozano H.	Diego H. Zavala GC	Edgar Mendoza Arreola	Javier Elizondo Campos	José de Jesús Hernández Suárez	Juan Carlos Segundo Elías	Marco Dorantes Martínez	Roberto Chávez Quintero	Sergio Cardoso
		Intermec Technologies de Mexico	iquatro	ING Comercial América	Qualitas Compañía de Seguros, S.A. de C.V.	Desarrollo Creativo	ING Comercial América	Hildebrando, SA de CV	DA, Organismo Integrado de Distribución S.A. de C.V.	ING Comercial América	Dirección de Sistemas-UNAM	SHCP-TESOFE	Microsoft Consulting	ING Comercial América	ISOFT Ingeniería de Software
13	Sí					✓		✓		✓		✓	✓		✓
	No	✓	✓	✓	✓		✓		✓		✓	▪		✓	
13a	Sí	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	No														
14	Sí		✓				✓	✓			✓	✓	✓		✓
	No	✓		✓	✓	✓			✓	✓				✓	▪
14a	Sí	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	No														

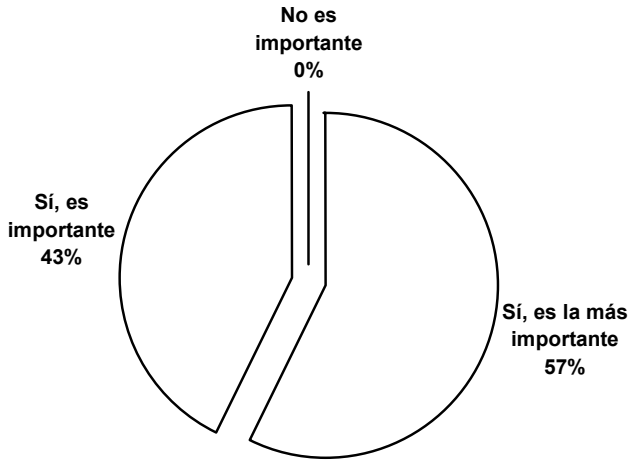
		Adrián Galindo Hernández	Alberto Cortés Ulloa	Alejandro Moreno Rivas	Alejandro Pérez Hernández	Antonio Malpica	Carlos Lozano H.	Diego H. Zavala GC	Edgar Mendoza Arreola	Javier Elizondo Campos	José de Jesús Hernández Suárez	Juan Carlos Segundo Elías	Marco Dorantes Martínez	Roberto Chávez Quintero	Sergio Cardoso
		Intermec Technologies de Mexico	iquatro	ING Comercial América	Qualitas Compañía de Seguros, S.A. de C.V.	Desarrollo Creativo	ING Comercial América	Hildebrando, SA de CV	DA, Organismo Integrado de Distribución S.A. de C.V.	ING Comercial América	Dirección de Sistemas-UNAM	SHCP-TESOFE	Microsoft Consulting	ING Comercial América	ISOFT Ingeniería de Software
15		Comunicación entre los integrantes. Conocimiento y dominio del problema por parte de los jefes.	La comprobación de requerimientos con el usuario y la existencia de formatos predeterminados de documentación y requerimientos.	Definiciones claras y concretas, objetivos y alcances bien definidos.	Una buena definición, junto con el usuario	El seguimiento sobre el avance del desarrollo del cumplimiento de los requerimientos.	La definición clara, precisa y concreta de los requerimientos.	Objetos pre-construidos.		Compromiso e involucra-miento de los usuarios. Administración de proyectos	Control sobre los requerimientos iniciales, control sobre los cambios de requerimientos para observar el impacto, estimación correcta del proyecto para tiempos y recursos.	Comprensión del objetivo del proyecto de sistemas, investigación preliminar, contacto estrecho con él.	El cliente-usuario es parte del equipo de desarrollo y esta 100% disponible para los programadores.		Definición de requeri-mientos y plan de pruebas
16	Sí	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	No														

4.9 Análisis de los resultados

A continuación se presentan los resultados obtenidos, desde distintas perspectivas: por pregunta, por entrevistado y por último de forma global.

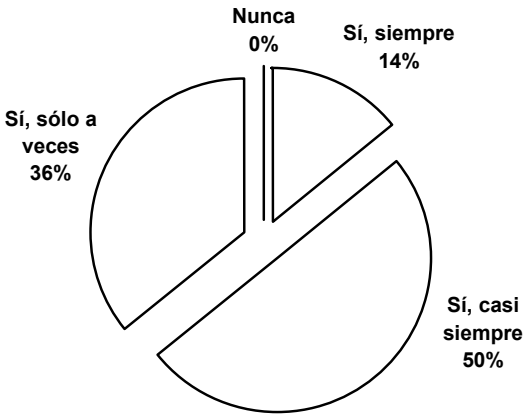
4.9.1 Por pregunta

1. ¿Considera que la ingeniería de requerimientos es una actividad clave dentro del ciclo de vida de desarrollo de software?



Todos consideran importantes las actividades de la ingeniería de requerimientos. Esto es buen indicio de que finalmente existe la convicción de que los requerimientos son medulares en el desarrollo de software. Actualmente de hecho, existe una tendencia por invertir más tiempo en las primeras fases del ciclo de desarrollo (requerimientos y diseño) que en la fase de codificación. Hay que reconocer todas y cada una de las fases y actividades en el desarrollo de software (por ejemplo: diseño, pruebas, codificación) tienen un objetivo específico y contribuyen en conjunto a obtener un producto final; todas son necesarias, importantes y se retroalimentan entre sí, pero lo relacionado con los requerimientos, por ser en donde se define lo que se va a hacer y por realizarse primeramente es la base para las demás actividades y productos.

2. ¿El software que desarrolla cumple con todas las características que el usuario necesita?



Afortunadamente ninguno en los encuestados ha participado en proyectos en los que no se satisfagan completamente las necesidades de los usuarios, caso muy extremo en el que nada de lo que realiza el sistema es útil. La realidad y los resultados obtenidos nos indican, sin embargo, que tampoco en la mayoría de los proyectos los usuarios quedan completamente satisfechos.

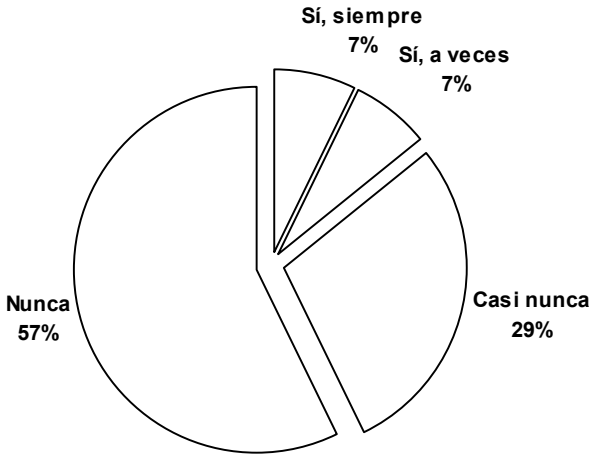
Ahora bien, los encuestados expresaron su punto de vista; sería interesante conocer el punto de vista del usuario para validar estos resultados.

Este factor depende de todas las actividades de la ingeniería de requerimientos mencionadas. En la obtención se identifican todas las necesidades reales de los usuarios. En el análisis y la negociación se revisan, ponderan y se establecen acuerdos sobre el alcance. La especificación permite establecer por escrito lo que se va a hacer y esos documentos son utilizados de alguna forma por todos los involucrados. En la validación el cliente y/o usuario juega un papel fundamental para determinar si los requerimientos son los correctos.

En todas estas actividades es importante reconocer que ambas partes tienen responsabilidad, tanto los clientes y usuarios como el equipo de desarrollo. Por el lado de los clientes y usuarios, el no proporcionar la información real de forma oportuna, el hecho de que existan intereses ajenos al proyecto, la falta de disposición y la resistencia al cambio son algunos de los factores más comunes.

Por otra parte, el equipo de desarrollo debe ser honesto en la viabilidad de los requerimientos y los recursos necesarios, con base en necesidades reales del usuario y no en otro tipo de intereses.

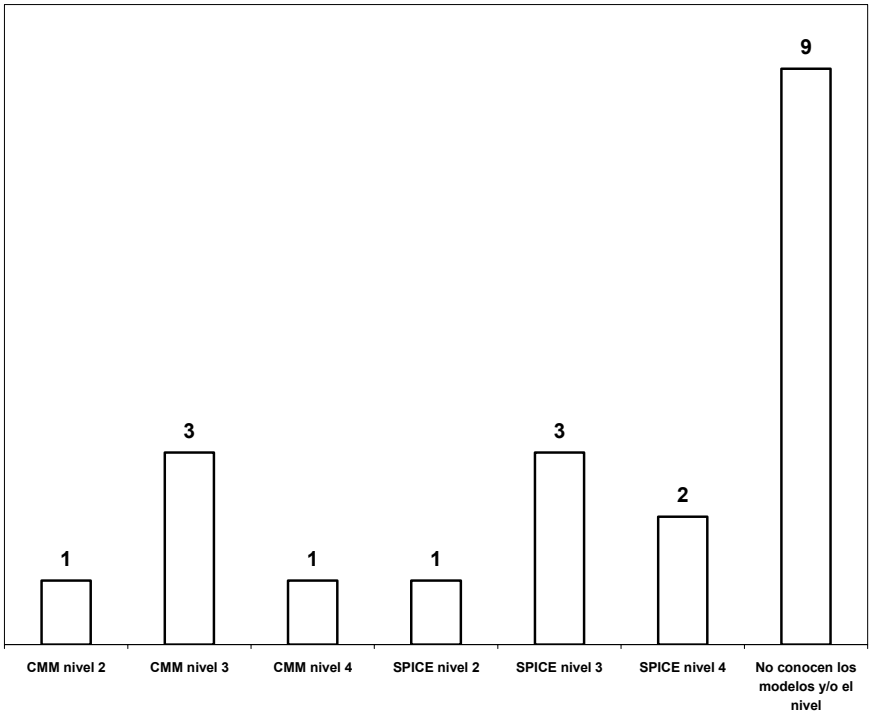
3. ¿Los proyectos de desarrollo de software se terminan a tiempo y con los recursos inicialmente estimados?



Los resultados de esta gráfica coinciden con diversas estadísticas generales y criterios de diversos autores mencionados en el marco conceptual, sobre los costos y tiempos sobrepasados. Si bien lo relacionado con los requerimiento no es el único factor que contribuye a estas desviaciones, porque existen otros aspectos como: -mala planeación ye estimación, adaptación a nuevas tecnologías, tiempo y recursos limitados, entre otras-, es una de las más importantes.

Es evidente que si no se tiene una clara definición de lo que se va hacer en un proyecto, no se puede planear y estimar adecuadamente; si no se tiene por escrito lo que se va a hacer, no hay manera de comprobar los acuerdos y alcances iniciales, no existe una comprensión común entre los involucrados, se depende de personas y todo ello constituye un factor riesgo para que los proyectos fracasen.

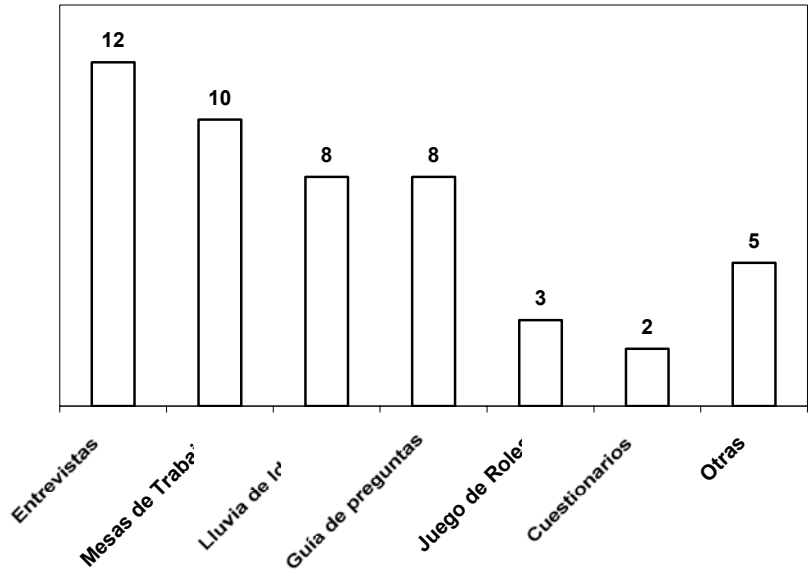
4. Si conoce los modelos de madurez como CMM o SPICE ¿En qué nivel considera que se encuentra su organización respecto a alguno de estos?



Solo 6 de cada 10 personas encuestadas conocen los modelos de madurez de procesos, los cuales se están convirtiendo en un estándar de la industria de desarrollo de software en el mundo. En México desafortunadamente son contadas las organizaciones con niveles comprobables de madurez de procesos para considerarse competitivas en el ámbito internacional y aprovechar la oportunidad de tener cerca al mayor consumidor de software en el mundo: Estados Unidos.

La información proporcionada por la mayoría de los encuestados respecto a los niveles en los que consideran a sus organizaciones, llega a ser muy cuestionable, ya que alcanzar los niveles de madurez no es sencillo; implica realizar diversas prácticas e invertir mucho tiempo y recursos en la definición de procesos. Sin embargo esto no es imposible, y hay algunas organizaciones en México que cumplen con ello.

5. ¿Qué técnicas utiliza y considera efectivas para la identificación y obtención de los requerimientos con el usuario?

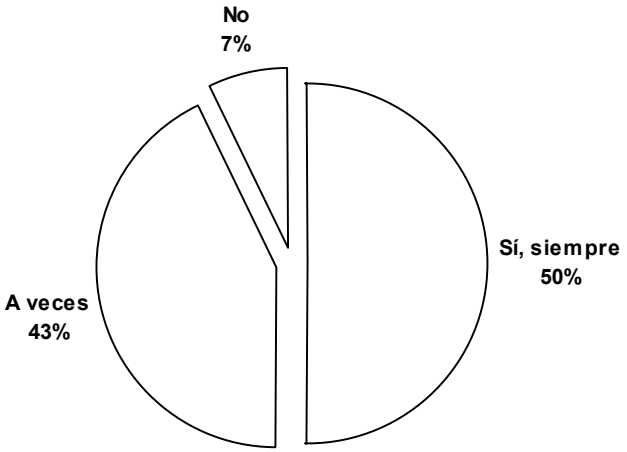


Las entrevistas son la técnica más común de recopilación de información. El hecho de estar frente a frente con los clientes y usuarios finales del sistema permite percibir el ambiente de la organización, observar la actitud de las personas y la forma de operación de sus procesos. Si bien una entrevista debe esta planeada de acuerdo a un objetivo, permite flexibilidad y da oportunidad a tratar otros puntos de interés no contemplados.

Casi todos los encuestados las utilizan, además de las mesas de trabajo con los clientes y usuarios, la lluvia de ideas como complemento a éstas, el juego de roles, las guías de preguntas y los cuestionarios específicos. Independientemente de la tecnología y de las herramientas de desarrollo y comunicación, éstas técnicas clásicas de obtención de información se siguen utilizando y son de gran utilidad.

Dentro del rubro de otras técnicas se encuentran los prototipos, utilizados por algunos de los encuestados.

6. ¿Realiza un análisis de los requerimientos con el fin de depurarlos, clasificarlos, ponderarlos y definir así claramente el alcance de la solución?

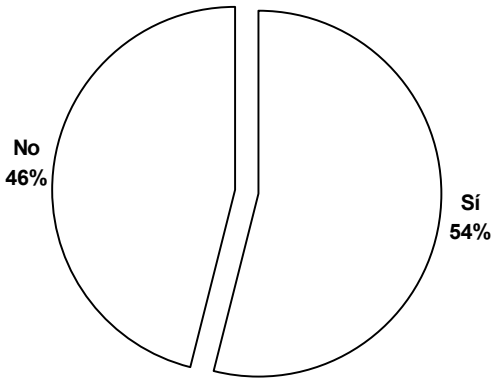


La mayoría de los encuestados realiza un análisis de los requerimientos, sin embargo no se obtuvo en la encuesta qué tan adecuado o profundo es dicho análisis. Considero que el análisis que supuestamente se realiza no se hace de manera formal o no contempla los puntos tratados en este trabajo, como son asignar: riesgo, prioridad, estado, estabilidad, recursos, entre otros. O bien el término es nuevo para los encuestados.

No es común encontrar en la bibliografía clásica de ingeniería de software algo específico sobre análisis de requerimientos. Podría confundirse con "análisis del sistema" pero estrictamente no es lo mismo; el análisis de requerimientos es muy específico. Anteriormente –y todavía– suele llamarse análisis de sistemas a esa actividad en la que especificamos el "qué" de un sistema informático; esto es correcto, sin embargo actualmente existe la tendencia de enfocarse a procesos y apoyarse en las herramientas de la ingeniería de software, por lo que el enfoque está evolucionando hacia la ingeniería de requerimientos.

Análisis de requerimientos, como se expuso en el Marco Conceptual de este trabajo, es una parte de la ingeniería de requerimientos que permite descubrir conflictos, traslapes, omisiones e inconsistencias para obtener un conjunto acordado de requerimientos.

7. Si realiza este análisis, ¿Utiliza alguna guía para ello?



En complemento con esta cuestión, se preguntó: "Si realiza este análisis, ¿utiliza alguna guía para ello?", se obtuvieron que un poco más de la mitad sí utiliza una guía.

El contar con una guía simple para el análisis de requerimientos es importante para realizarlo de manera adecuada y para que todos los involucrados los interpreten de la misma forma. Todos los encuestados coinciden en que una guía es o sería útil para esta actividad.

Una guía completa debería contener los lineamientos para clasificar y ponderar los requerimientos, así como describir el procedimiento para hacerlo. De hecho como tal no se encontró nada en las fuentes información. No obstante en el Marco Conceptual de este trabajo se presentan los principales atributos y aspectos a considerar en esta actividad, además se presenta un "checklist" o lista de verificación con una serie de preguntas que pueden servir para orientar el proceso de análisis de requerimientos.

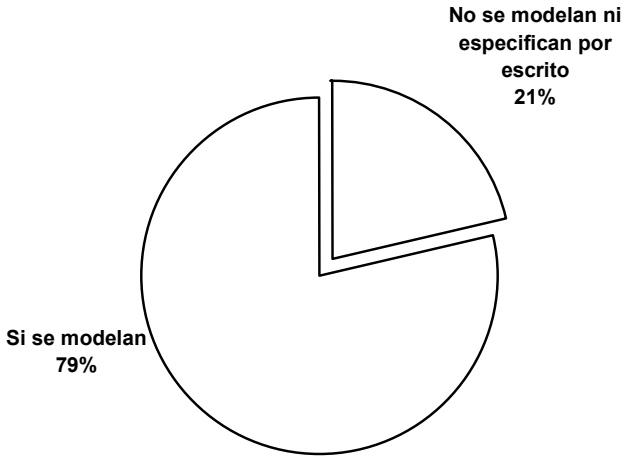
8. ¿Utiliza alguna guía de contenido estándar para la documentación/especificación de requerimientos?

La mayoría de los encuestados no cuenta con un estándar para la documentación de requerimientos. Si bien no existe un guía general que sea la mejor para todos los casos, debido a que la naturaleza de los proyectos y organizaciones es distinta, es importante cubrir mínimo algunos puntos fundamentales en la especificación de requerimientos. Generalmente cuando se documentan los requerimientos se hace de manera "ad hoc"

Internamente, un estándar permite lograr un mejor comunicación entre los equipos involucrados en el desarrollo, se tendría una guía para que los encargados de requerimientos los especifiquen de la misma forma.

Externamente, el poder contar con un estándar permitirá una mejor interacción, por ejemplo entre una organización que requiere un sistema y una empresa que los desarrolla. Todas las posibles empresas que se dedican al desarrollo del sistema comprenderían de la misma manera el documento; la organización que los solicita podría utilizar el mismo documento para solicitar diversas propuestas de diseño e implementación.

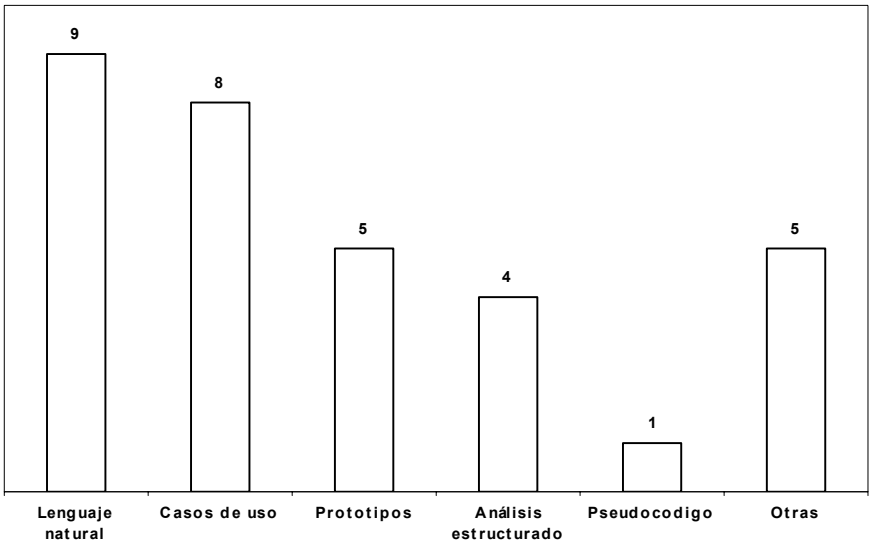
9. ¿Qué técnicas utiliza y considera efectivas para el modelado y especificación de requerimientos?



La mayoría de los encuestados realizan un modelado y especificación de requerimientos. Esto confirma la convicción por parte de las áreas y organizaciones de desarrollo de sistemas y software, sobre la importancia de especificar de alguna manera los requerimientos, y que sirva como una base para establecer el alcance, para conducir el desarrollo y para la negociación.

Independientemente de la técnica, metodología o enfoque que se utilice, estamos convencidos que de alguna manera es necesario especificar los requerimientos. Algunos "nuevos" enfoque para el desarrollo de software como XP (Extreme Programming) no comparten la filosofía del modelado y documentación tradicional. Sin embargo, como su nombre lo dice, es un enfoque extremo, que puede resultar muy efectivo para algunos casos, siempre y cuando los clientes y usuarios participen convencidos bajo esa filosofía.

Retomando los enfoques más tradiciones de modelado y especificación, en la gráfica siguiente se indican las técnicas más utilizadas por los encuestados.



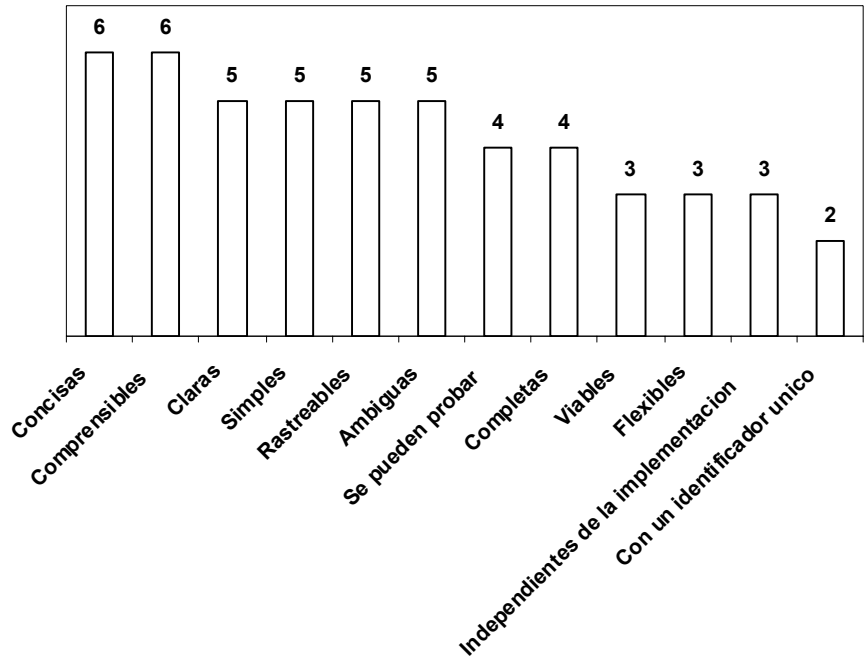
Las especificaciones basadas en lenguaje natural son las más utilizadas, ya que esto permite que tanto el equipo de desarrollo como los clientes y usuarios comprendan lo que el sistema de debe realizar. Finalmente la mayoría de las diversas técnicas de especificación y modelado van complementadas con alguna descripción o explicación en lenguaje natural.

Los casos de uso son una técnica muy utilizada en el modelado y especificación de los requerimientos. Son comprensibles por todos los involucrados y como se mencionó en el Marco Conceptual ofrecen diversas ventajas. Con los casos de uso por un lado se modelan los requerimientos funcionales y sus actores (modelo de casos de uso) y también se especifican por escrito en lenguaje natural cada uno de estos.

Los prototipos también representan una técnica de especificación de requerimientos. Muchas veces se usan en complemento con otras.

El análisis estructurado es una técnica que cada vez está siendo menos utilizada, debido a que las herramientas actuales de desarrollo presentan un enfoque más bien de orientación a objetos.

10. ¿Cómo considera que son las especificaciones de requerimientos que se realizan en su organización?

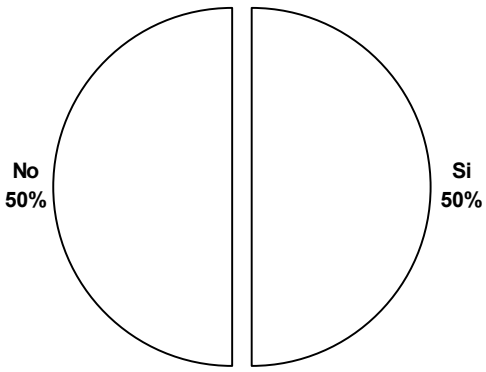


En esta pregunta cabe mencionar que 11 de las 12 características propuestas son positivas, es decir, son recomendables que las contengan las especificaciones. Solo una es negativa: la que se refiere a la ambigüedad. No obstante se observa que la mayoría se encuentran en una frecuencia similar, tanto las positivas como la negativa.

Llama la atención que la mayoría de los encuestados, hayan comprendido a los que se refería la característica de "rastreable", ya que no si bien es un término común en la administración de requerimientos, en general no es tan conocido.

Los resultados no reflejaron alguna característica en la que hayan coincidido todos los encuestados.

11. ¿Realiza una validación y verificación de requerimientos para garantizar que sean correctos (que representen lo que el usuario necesita, que expresen lo que tengan que decir, que sean los que deben de ser, que se especifiquen de la manera correcta)?

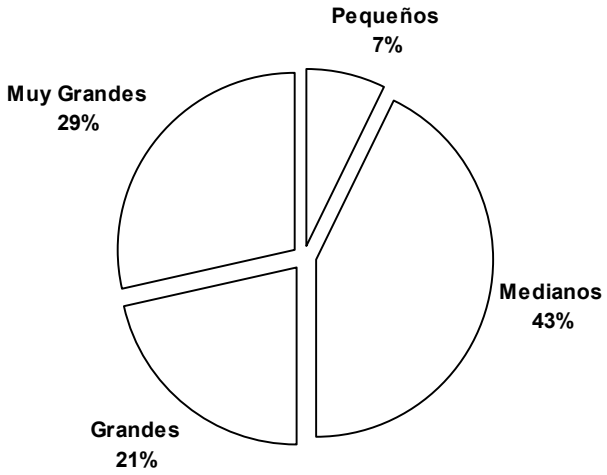


Es interesante observar que los resultados están equilibrados. La mitad sí realiza una validación y verificación de los requerimientos, la otra mitad no. Quizás esta última ni siquiera había considerado la posibilidad de hacerlo.

Si no se realiza una validación de los requerimientos con el usuario, se corre el riesgo de que aun obteniendo, especificando y analizando adecuadamente los requerimientos, no se desarrolle un sistema que satisfaga las necesidades de los usuarios.

Tampoco existe una manera universal de realizar esta validación y verificación, no obstante en este trabajo se ofrece una seria de preguntas y recomendaciones que pueden ayudar a realizar esta importante actividad de la ingeniería de requerimientos.

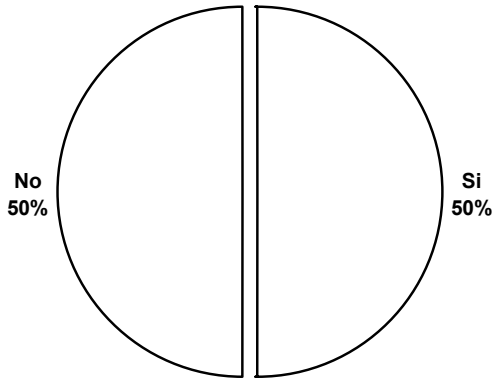
12. ¿En promedio de qué tamaño son los sistemas o el software que se desarrolla en su organización?



"El problema de la ingeniería de requerimientos crece de manera exponencial con respecto a la cantidad de requerimientos". Esto es una gran verdad. Entre menos sea el número de requerimientos se tiene un mayor control sobre los mismos. Entre mayor sea el número puede crecer la complejidad del sistema, el tiempo de desarrollo, los recursos necesarios (humanos, materiales y técnicos) y se requiere un mayor control de los mismos.

Esta pregunta permite ubicar las respuestas de cada encuestado con respecto al tamaño de los sistemas que desarrolla. Se puede observar que la mayoría son sistemas medianos, es decir entre 10 y 50 requerimientos o entre 2,500 y 25,000 líneas de código.

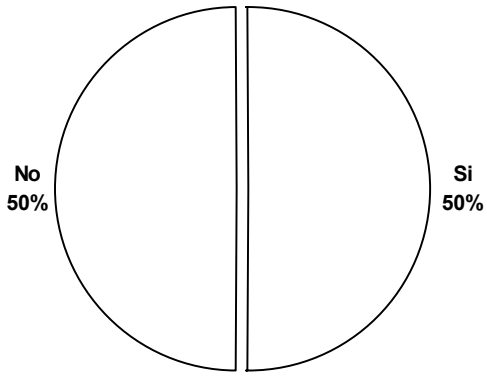
13. ¿Cuenta con políticas para la administración de requerimientos (organización, control y seguimiento de los cambios en los requerimientos)?



También en esta pregunta se observa una igualdad en los porcentajes de respuesta. La mitad sí cuenta con políticas para la administración de requerimientos, la mitad no.

En complemento con la anterior, entre mayor sea el tamaño de los sistemas es más necesarios contar con políticas para la administración de los mismos. Desafortunadamente en la encuesta no se obtuvo el detalle de qué aspectos cubren esas políticas.

14. ¿Utiliza alguna herramienta de software para la administración de requerimientos?



También en esta pregunta se observa una igualdad en los resultados. Está muy relacionada con la anterior, por lo que de alguna manera se validan y complementan.

La necesidad de usar herramientas de software para la administración de requerimientos está directamente relacionada con el número de estos, es decir, con el tamaño de los sistemas que se desarrolla. A mayor número de requerimientos se necesiten controlar, es más necesaria una herramienta para administrarlos.

Las herramientas que se utilizan van desde simples listados y matrices de dependencia entre los requerimientos que se realizan en una hoja de cálculo, hasta herramientas comerciales como las mencionadas en el punto 3.2.6.4.

15. ¿Qué aspectos de la ingeniería de requerimientos considera muy importantes para una mayor calidad del software y para el éxito de los proyectos?

Esta pregunta fue abierta, y de las respuestas de los encuestados resaltan los siguientes aspectos:

- Involucramiento y disponibilidad del cliente y el usuario.
- Aspectos de administración de proyectos.
- Conocimiento del dominio del problema.
- Comunicación entre el equipo de desarrollo.
- Plantillas para la documentación de requerimientos.
- Definición clara, precisa y concreta de los requerimientos.
- Definición clara del alcance.
- Control de cambios en los requerimientos.
- Reutilización.

Definitivamente la participación del cliente y/o usuario es un aspecto que en varias ocasiones mencionaron los encuestados, de hecho desde el sondeo del Marco Problemático, llamó la atención esta insuficiente participación del usuario.

Otro aspecto muy mencionado y que si bien estrictamente no es una actividad de la ingeniería de requerimientos, depende en gran medida, es la administración de proyectos. Si no se estiman y planean adecuadamente las actividades del proyecto, éste fracasará por no terminarse a tiempo. Pero a su vez, si no se tiene requerimientos claros, un alcance bien definido y se administra el impacto de los cambios, no se puede realizar una adecuada planeación y seguimiento de proyectos.

El tercer aspecto muy importante tiene que ver con el conocimiento del dominio del problema. Para proporcionar una solución adecuada y generar requerimientos con base en necesidades reales del negocio, es fundamental conocer el dominio del problema, el ambiente de la organización, los proceso de los usuarios y los objetivos de negocio.

16. ¿Considera las actividades de la ingeniería de requerimientos factor clave para el éxito o fracaso de los proyectos de desarrollo de software?

Todos los encuestados coincidieron en que efectivamente las actividades de la ingeniería de requerimientos son factor clave en el éxito o fracaso de los proyectos de desarrollo de software. Estas opiniones permitirán validar en gran medida la hipótesis del trabajo.

Cabe mencionar que no es él único factor, todas las actividades del ciclo de vida contribuyen finalmente al producto final, pero sí es crítico tener los requerimientos correctos, expresados de manera adecuada y administrar el cambio de éstos.

Afortunadamente también, la visión de los encuestados es favorable hacia invertir tiempo y recursos en la actividad de requerimientos; ya no es tan evidente esa tendencia por la programación y los aspectos técnicos solamente. Se reconoce también la importancia de las actividades de administración de proyectos.

4.9.2 Por encuestado

Adrián Galindo
Intermec Technologies de Mexico

Utiliza lenguaje natural, casos de uso y prototipos. Casi nunca se terminan a tiempo y con los recursos estimados pero se cubren las expectativas del usuario. No conoce los modelos de CMM y SPICE.

Alberto Cortés Ulloa
iquatro

Considera a la ingeniería de requerimientos como lo más importante. Casi siempre cubren las expectativas de los usuarios. Si conoce los niveles de madurez, considera 3 para CMM y 4 para SPICE. Utiliza casos de uso y lenguaje natural, sin embargo no cumplen las características de calidad sus requerimientos.

Alejandro Moreno Rivas
Seguros ING Comercial América

Solo a veces el software cumple con las expectativas del usuario. Casi nunca se terminan a tiempo los sistemas que desarrolla porque se dan constante cambios en los sistemas que desarrolla. Realiza entrevistas con los usuarios. Modela con lenguaje natural, pseudocódigo y análisis estructurado. Los sistemas son muy grandes y no cuenta ni con políticas ni con herramientas de administración.

Alejandro Pérez Hernández
Qualitas Compañía de Seguros, S.A. de C.V.

Las actividades de la ingeniería de requerimientos son lo más importante. Casi siempre se cumplen las expectativas del usuario. Si se realiza una verificación y validación de sus requerimientos. Sus sistemas con pequeños y no cuenta con políticas ni herramientas para la administración de requerimientos.

Antonio Malpica
Desarrollo Creativo

Solo a veces se terminan los proyectos con los recursos estimados. Sus sistemas son medianos y modela con lenguaje natural y prototipos. No conoce los modelos de madurez de procesos.

Carlos Lozano
Seguros ING Comercial América
Casi nunca se terminan a tiempo los sistemas que desarrolla. Solo a veces el software cumple con las expectativas del usuario. NO modela ni especifica los requerimientos por escrito. Sus sistemas son pequeños.

Diego H Zavala GC
Hildebrando, SA de CV
Considera a las actividades de la ingeniería de requerimientos como lo más importante, casi siempre se cubren las expectativas de los usuarios, sin embargo sólo a veces se terminan los proyectos con los recursos estimados. Se considera en nivel 3 de CMM. Utiliza lenguaje natural, prototipos y análisis estructurado.

Edgar Mendoza Arreola
DA, Organismo Integrado de Distribución, S.A. de C.V.
Utiliza casos de uso. Los sistemas son grandes. No cuenta con estándares ni políticas. No se validan los requerimientos por lo que sólo a veces se cumple completamente con las expectativas de los usuarios.

Javier Elizondo
Seguros ING Comercial América
Considera a las actividades de la ingeniería de requerimientos como lo más importante. Sí utiliza casos de uso. Sus sistemas son grandes y casi siempre se terminan a tiempo y casi siempre se cubren las expectativas de los usuarios. No conoce los modelos de madurez de procesos. Considera que el compromiso e involucramiento de los usuarios es vital.

José de Jesús Hernánde Suárez
Dirección de Sistemas, DGSCA, UNAM
Casi nunca se terminan los proyectos con los recursos inicialmente estimados, debido en gran medida a una mala estimación y loa tiempo insuficientes para su desarrollo. Ubica a su organización en nivel 2 de CMM y de SPICE.

Juan Carlos Segundo Elías
SHCP-TESOFE
Los sistemas son muy grandes. Utiliza análisis estructurado y otras técnicas complementarias. Considera a la ingeniería de requerimientos como lo más importante pero sólo a veces se cumplen las expectativas del usuario.

Marco Dorantes
Microsoft Consulting

De acuerdo con las respuestas de Marco Dorantes, los sistemas siempre se terminan a tiempo y dentro del presupuesto estimado. Lo atribuyen principalmente a los ciclos cortos de entrega. Considera a las actividades de la ingeniería de requerimientos como importantes. Estima nivel 3 de CMM y 4 de SPICE. Utiliza pruebas de aceptación automatizadas. Utiliza casos de uso y herramientas para la administración de requerimientos. Sus sistemas son grandes. Considera al usuario como parte del equipo de desarrollo. Las respuestas de este encuestado sugieren una buena visión y madurez en sus procesos de desarrollo.

Roberto Chávez Quintero
Seguros ING Comercial América

Solo a veces el software cumple con las expectativas del usuario. Casi nunca se terminan a tiempo los sistemas que desarrolla. Considera a las actividades de la ingeniería de requerimientos como importantes. Desconoce los niveles de CMM y SPICE. Realizan las especificaciones por pseudocódigo. Los sistemas son muy grandes y sus requerimientos son ambiguos. No cuenta con políticas de administración ni con una guía estándar para la especificación de requerimientos.

Sergio Cardoso
ISOFT Ingeniería de Software

Se considera en nivel 3 de CMM, no obstante sus especificaciones no cumplen con atributos de calidad. Le da gran importancia a las pruebas.

4.10 Conclusiones generales sobre los resultados

Es satisfactorio concluir que existe una gran conciencia de que la ingeniería de requerimientos es una actividad importante dentro del ciclo de desarrollo. Afortunadamente ya no se tiene la visión de hace algunos años, en que la programación y la corrección de defectos eran los más importante y en lo que se gastaba (no invertía) la mayor parte del tiempo, y que constituía un gran riesgo en el desarrollo de software.

No obstante, dentro de los encuestados sigue existiendo el problema de que los proyectos no se terminan a tiempo ni con los recursos estimados, por lo que si bien los requerimientos son una parte muy importante para establecer el alcance, manejar líneas base y negociar los recursos del proyecto, existen otros factores que afectan y que salen del alcance de este trabajo.

Muchos de los encuestados usan y les funcionan los casos de uso como una técnica de modelado y especificación de requerimientos, técnica que detallamos en el Marco Conceptual de este trabajo por considerar que tiene diversas ventajas y con la cuál también se ha trabajo en diversos proyectos.

Todos los encuestados coincidieron en que efectivamente las actividades de la ingeniería de requerimientos son factor clave en el éxito o fracaso de los proyectos de desarrollo de software. Además, los dos aspectos más mencionados que también intervienen en el éxito de los proyectos son: la participación de los usuarios y la administración de proyectos. Se percibe más debilidad en aspectos administrativos que técnicos.

La naturaleza de las organizaciones encuestadas es distinta, no obstante operan todas en México y presentan situaciones y características similares. Es de especial interés analizar las respuestas de aquellas conocen los modelos de madurez de procesos, debido a que conocen las tendencias actuales en la ingeniería de software y en determinado momento podrían ser más competitivas. Desgraciadamente son menos de la mitad.

4.11 Aprobación de la hipótesis

De acuerdo con los resultados a continuación se comenta el grado de aceptación de las variables dependientes con respecto a la dependiente:

Variable independiente:

La realización de las actividades de la ingeniería de requerimientos, mediante el uso de técnicas y herramientas en un proyecto de desarrollo de software permite:

Variables dependientes:

Variable	Porcentaje de aceptación
Especificar de manera clara, precisa, y completa los requerimientos de un sistema informático	Las técnicas y herramientas de la ingeniería de requerimientos sí permiten lograr claridad, precisión y completez en los requerimientos, especialmente aquellas que incluyen un modelo gráfico y una descripción textual en lenguaje natural. **Porcentaje de aceptación: 100%**
Desarrollar sistemas con la funcionalidad y características requeridas por el usuario	Definitivamente para el cumplimiento de las expectativas del usuario, las actividades de la ingeniería de requerimientos sí contribuyen a lograrlo. **Porcentaje de aceptación: 100%**
Planear y realizar estimaciones más precisas de los proyectos	El contar con requerimientos precisos, documentados, clasificados y ponderados ayuda a establecer líneas de base para planear y estimar los recursos del proyecto. Sin embargo en algunos casos, a pesar de esto, los proyectos

	no se terminan a tiempo y dentro de los recursos estimados inicialmente. **Porcentaje de aceptación: 75%**
Lograr una mayor calidad del software	Los dos principales factores de la calidad son software son: que el software tenga la funcionalidad requerida por el usuarios y que lo haga bien. Las actividades de la ingeniería de requerimientos sí coadyuvan a lograr la primera, no obstante en la segunda intervienen factores de otras fases y actividades del ciclo de desarrollo. **Porcentaje de aceptación: 50%**
Cumplir con los lineamientos de los modelos internacionales de madurez de procesos	Los principales modelos de madurez de procesos consideran a las actividades de la ingeniería de requerimientos como parte de los requisitos para lograr determinado nivel de madurez, por lo que al realizar ingeniería de requerimientos sí se cumplen los lineamientos; no obstante estas actividades si bien son una parte muy importante en ambos modelos, sólo constituyen una aspecto. **Porcentaje de aceptación: 75%**

Las respuestas obtenidas permiten considerar por lo tanto a la hipótesis en un 80% aceptada.

5 MARCO INSTRUMENTAL

5.1 Propuestas de acción

Como resultado del presente trabajo de investigación, y con el fin de difundir información sobre el tema, a continuación se exponen las actividades a realizar.

A) Publicación de un artículo en la publicación "Algo mas" de la Facultad de Contaduría y Administración, con el propósito de dar a conocer de manera general a la comunidad de la facultad, lo que es la ingeniería de requerimientos, la importancia de la participación de los usuarios en el desarrollo de sistemas informáticos y el papel de los Licenciados en Informática en esta área.

B) Elaboración de una propuesta a la Coordinación de Informática para complementar el plan de estudios la de Licenciatura en Informática con está temática, lo cual permitirá que los egresados tengan las herramientas para cubrir esta importante área de su desarrollo profesional, tomando en cuenta que es un requisito de los estándares internacionales de calidad en el desarrollo de software.

C) Desarrollo y difusión de un sitio en Internet, el primero en México en este tema, que presente información actualizada y de interés para la comunidad informática del país. Como primera fase de este sitio, se pondrá de forma inmediata el presente trabajo de investigación a disposición de las personas interesadas. Cabe mencionar que las encuestas aplicadas en este trabajo y sus resultados detallados, se encuentran ya publicados en Internet.

D) Ofrecer cursos y conferencias sobre el tema de investigación, complementado con la experiencia profesional adquirida en esta área, en los eventos relacionados con la informática y el desarrollo de software, en los que pueda resultar de gran interés al auditorio.

5.2 Plan y programa de trabajo

A) Publicación de un artículo

Respecto a la publicación del artículo, éste fue entregado al Secretario de Divulgación y Fomento Editorial de la Facultad de Contaduría y Administración el día 15 de mayo del 2002, teniendo como respuesta que de acuerdo a la temática y características del artículo, se planearía su publicación en alguno de los próximos números de Junio o Julio de "Algo Más", dependiendo de los espacios y el material programado previamente en la publicación.

Cabe mencionar que el artículo se enfocó principalmente a la importancia de la participación de los usuarios en los proyectos de desarrollo de software, pero además se tocaron algunos aspectos de la ingeniería de requerimientos, tratando de que fuera una lectura breve y ligera pero que despertara interés sobre el tema.

Se anexa la carta de presentación y el contenido del artículo.

La participación de los usuarios: elemento esencial para el éxito en el desarrollo de los sistemas informáticos.

Existen infinidad de problemas relacionados con el desarrollo de software, pero uno de los más graves es que los sistemas no hagan lo que el usuario necesita, es entonces que algunas veces los informáticos decimos que "los usuarios no saben lo que quieren", pero:

¿El usuario no sabe lo que quiere o el informático no sabe identificar sus necesidades reales?

Diariamente interactuamos con sistemas informáticos que fueron desarrollados para realizar muchas de nuestras actividades, sin embargo no siempre nos ofrecen la funcionalidad que deseamos. Nos referimos en esta ocasión a esos sistemas que fueron creados "a la medida", es decir que no son precisamente software comercial o de uso genérico, sino aplicaciones específicas para automatizar algún proceso o administrar cierta información de nuestra organización.

De manera muy general, el proceso común para desarrollar esas aplicaciones comprende las siguientes fases: 1) determinación de requerimientos, 2) diseño de la solución, 3)implementación, 4)pruebas y 5)entrega e implantación. Cada fase es un área de especialización por sí misma.

La actividad más crítica es la determinación de requerimientos, ya que es en donde se especifica **qué** se va a hacer, por lo que sirve de insumo para las demás, y es desafortunadamente en la que ocurren más errores y a la que menos importancia se le da. Para realizar adecuadamente esta actividad, los analistas de sistemas deben tomar conciencia de su importancia y capacitarse en las técnicas y herramientas de la **Ingeniería de Requerimientos.**

La ingeniería de requerimientos se enfoca a la obtención, análisis, negociación, especificación, validación y administración de los

Además, el involucramiento del usuario es un factor principal para el éxito de este tipo de proyectos. Es parte del equipo, y si bien en la mayoría de los casos no es un especialista informático, es el que mejor conoce sus procesos de negocio, el que diariamente realiza las actividades de su área y quien finalmente utilizará y dará el visto bueno al sistema. **El usuario es nuestro cliente.**

De acuerdo con la ingeniería de requerimientos, el cliente y usuario participa de alguna manera en todas las actividades de la ingeniería de requerimientos, principalmente junto con los especialistas informáticos en la obtención, análisis, negociación y validación de requerimientos. A los informáticos les corresponde en mayor medida la especificación por escrito y la administración de los requerimientos.

Por lo tanto, no es responsabilidad de los usuarios especificar exactamente qué es lo quieren desde el punto informático puesto que no son especialistas, no obstante sí debe participar activamente en lo relacionado con los requerimientos, y de hecho de alguna manera en todas las fases del desarrollo, debido a que son los **expertos en su área de negocio** y utilizarán el sistema.

Es responsabilidad principal de los analistas, identificar la problemática, conducir a los usuarios en la determinación de sus requerimientos informáticos, especificar los requerimientos por escrito, analizarlos, validarlos y propiciar la participación continua del usuario. Para todo esto puede auxiliarse de las técnicas y herramientas de la ingeniería de requerimientos. **El especialista informático es también parte del equipo y comparte durante el proyecto los objetivos de negocio del usuario.**

Ma. Teresa Ventura Miranda
Licenciatura en Informática

B) Propuesta de materia optativa para el plan de estudios

El día 9 de Mayo de 2002 se entregó en la Jefatura de la División de Informática, la propuesta de contenido para la creación de una materia optativa sobre Ingeniería de Requerimientos, considerando que es una de las áreas de especialización más idóneas para el licenciado en Informática, considerando que por tener conocimientos técnicos y administrativos es un buen intermediario entre los usuarios y el equipo de desarrollo.

Se anexa la carta de presentación y el contenido del temario.

TEMARIO PARA MATERIA OPTATIVA DE INGENIERÍA DE REQUERIMIENTOS

NOMBRE	Ingeniería de Requerimientos
OBJETIVO:	El participante conocerá diversas técnicas y herramientas para la obtención, análisis, negociación, especificación, validación y administración de los requerimientos de un sistema de información.
PRERREQUISITOS:	Haber cursado alguna asignatura relacionada con el proceso de desarrollo de sistemas, análisis y diseño de sistemas o ingeniería de software.
HORAS:	2 horas por clase. 2 clases por semana.
CONTENIDO:	**I. INTRODUCCIÓN** **1. Problemática relacionada con los requerimientos** **2. Conceptos de la ingeniería de requerimientos** 2.1 Clasificación de los requerimientos (funcionales y no funcionales) 2.2 Niveles de requerimientos (De usuario, de sistema y de software) 2.3 Proceso general de la ingeniería de requerimientos **3. Participantes en la ingeniería de requerimientos** 3.1 Rol de los clientes y usuarios 3.2 Rol de los analistas 3.3 Otros involucrados (líder de proyecto, programadores, probadores) **II. OBTENCIÓN DE REQUERIMIENTOS** **1. Fuentes de requerimientos** 1.1 Estudios de mercado 1.2 Manuales de organización y procedimientos 1.3 Observaciones y medidas del sistema actual 1.4 Entrevistas con el cliente y los usuarios 1.5 Documentación de los sistemas actuales 1.6 Modelos y prototipos **2. Problemática en la obtención de requerimientos** **3. Técnicas para la obtención de requerimientos** 3.1 Entrevistas y cuestionarios 3.2 Talleres 3.3 Lluvia de ideas 3.4 Presentaciones 3.5 Juego de roles 3.6 Prototipos

III. ANÁLISIS Y NEGOCIACIÓN DE REQUERIMIENTOS

1. **Análisis de problema**
2. **Definición del alcance del proyecto**
3. **Clasificación y ponderación de los requerimientos**
   - 3.1 Tipo
   - 3.2 Criticidad
   - 3.3 Importancia
   - 3.4 Estado

IV. MODELADO Y ESPECIFICACIÓN DE REQUERIMIENTOS

1. **Características de calidad en las especificaciones de requerimientos**
   - 1.1 Correcto
   - 1.2 Sin ambigüedad
   - 1.3 Completo
   - 1.4 Consistente
   - 1.5 Verificable
   - 1.6 Modificable
   - 1.7 Posibilidad de rastreo
   - 1.8 Comprensible

2. **Técnicas para el modelado y especificación**
   - 2.1 Casos de Uso (UML)
   - 2.2 Prototipos
   - 2.3 Otras

V. VALIDACIÓN Y VERIFICACIÓN DE LOS REQUERIMIENTOS

1. **Validación**
   - 1.1 Concepto de validación
   - 1.2 Métodos de validación
2. **Verificación**
   - 2.1 Concepto de verificación
   - 2.2 Métodos de verificación

VI. ADMINISTRACIÓN DE REQUERIMIENTOS

1. **Tipos de cambios en los requerimientos**
2. **Rastreabilidad (Traceability)**
3. **Herramientas de software para la administración de requerimientos**

BIBLIOGRAFÍA:

- Leffingwell, Dean y Wodrig, Don. **MANAGING SOFTWARE REQUIREMENTS**. Ed. Addison Wesley. EUA, 2000.
- Thayer, Dorfman. **SYSTEM AND SOFTWARE REQUIREMENTS ENGINEERING**. IEEE. 1990.
- Sommerville, Ian y Sawyer, Pete. **REQUIREMENTS ENGINEERING**. A Good Practice Guide. Ed. John Wiley & Sons.

C) Sitio en Internet

Actualmente la primera versión del sitio se encuentra en: http://genesis.dcaa.unam.mx/requerimientos y cuenta con los siguientes módulos:

- **Inicio.** Introducción al sitio, a quién está dirigido y cuál es su objetivo.
- **Actividades de la Ingeniería de Requerimientos.** Descripción de cada una de las actividades de la ingeniería de requerimientos tratadas en este trabajo.
- **Documentos.** Se podrán a disposición de los visitantes los principales documentos que se obtuvieron durante el trabajo de investigación y que fueron principalmente encontrados en Internet, pero con el valor agregado de presentarlos en un solo lugar especializado en la temática, organizados y con una breve descripción de su contenido.
- **Comentarios.** Posibilidad de enviar por correo sugerencias.

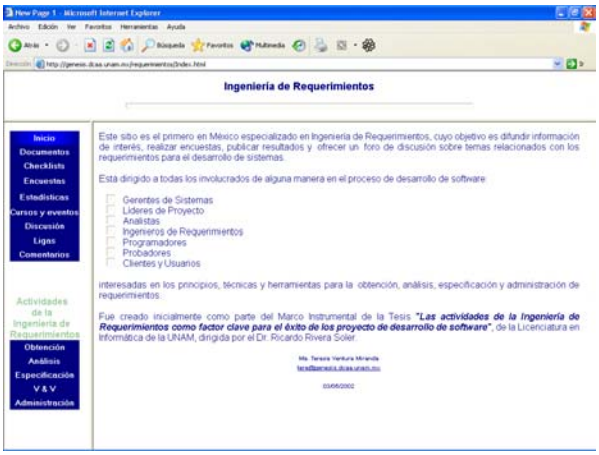
En la segunda fase, que se terminará el 17 de junio del 2002 se incorporará:

- **Ligas.** Referencias en Internet de interés sobre el tema.
- **Cheklists.** Las guías de preguntas clave para la obtención, análisis y negociación, validación y verificación, y administración de requerimientos, presentadas en este trabajo, se pondrán también a disposición de los interesados.
- **Cursos y eventos** en México relacionados con el tema.
- **Encuestas y Estadísticas.** Encuestas breves y abiertas de temas de interés. Se encuentran publicados los resultados de las encuestas realizadas en este trabajo (algunos ya se encontraban publicados a petición de los encuestados). Además, en fases posteriores se crearán nuevas encuestas de aspectos más específicos de la ingeniería de requerimientos que resultaron de interés.

En la tercera fase, que se terminará el 30 de agosto del 2002 se incorporará:

- **Discusión.** Se creará para fases posteriores una lista de discusión, la primera en México sobre el tema, para los especialistas interesados.

Presentación del sitio Web.
http://genesis.dcaa.unam.mx/requerimientos/index.html



6 CONCLUSIONES

6.1 Capítulo 1 (Marco Problemático)

Se reconoció la existencia de diversos problemas asociados a los requerimientos. Cuando se está involucrado de alguna manera en el desarrollo de software, es común experimentar estos problemas, la mayoría de los encuestados coincidieron en ello. Algunos ya están haciendo algo al respecto, aunque es difícil cuando el tiempo es una constante restricción, se da prioridad a otras cosas y desafortunadamente el problema se sigue arrastrando.

No es el único problema en el desarrollo de software pero sí uno de los más importantes. El desarrollo de software no es un proceso fácil, intervienen muchas variables y la naturaleza del software es muy especial por ser un producto intangible. Por lo mismo hay infinidad de problemas asociados a este proceso. Los requerimientos son uno de los más importantes debido a que constituyen la base para todas las fases y actividades subsecuentes en el desarrollo. Si no se tiene bien definido lo que quiere y lo que se va a hacer, no se puede hacer mucho, o no se van a tener resultados satisfactorios.

Existe conciencia de la importancia de los requerimientos en el ciclo de desarrollo. Afortunadamente la gran mayoría reconoce que los requerimientos son importantes, ya que algunos años atrás se le daba una mayor importancia y se gastaba más tiempo en codificar y corregir errores, lo que llevaba al fracaso de los proyectos y a estar manteniendo, adaptando y corrigiendo continuamente los sistemas.

El tema despertó interés entre los participantes de la encuesta. Fue muy satisfactorio recibir las respuestas de los encuestados y aparte comentarios adicionales por medio del correos electrónico sobre el gran interés en este tema de algunas personas y organizaciones. Como respuesta a este interés se ofreció a dichas personas información relacionada, el resultado de la encuesta y la posibilidad de consultar este trabajo por Internet una vez concluido.

La encuesta hizo reflexionar a los encuestados. Algunas personas no estaban tan concientes del la efectividad o ineficacia de sus proceso de requerimientos. A veces aunque se tenga la información, la conciencia y el conocimiento de que algo es importante, no lo aplicamos. Las preguntas permitieron reflexionar a los encuestados al respecto y despertó su interés.

Se estableció contacto con colegas de otras organizaciones. Con motivo del presente trabajo se conocieron personas que se dedican al desarrollo de sistemas en diversas organizaciones, lo que permitió percibir distintos puntos de vista, formas de trabajo, niveles de madurez y calidad, pero sobre todo un interés común: mejorar la forma de desarrollar sistemas.

La problemática resultó de mi interés. El tema de obtener requerimientos siempre resultó de mi interés: el determinar qué hacer, el modelar las soluciones a alto nivel e interactuar con los clientes y usuarios. En mayor o menor medida también había experimentado algunos aspectos del problema, lo que me llevó con mayor razón a tratar de encontrar causas raíces y ofrecer algunas soluciones.

6.2 Capítulo 2 (Marco Teórico)

La gran mayoría del material (libros, revistas, Internet) está en inglés. La ingeniería de requerimientos no es un tema precisamente nuevo, no obstante gran parte del material está en inglés y no hallé ningún intento de traducción de alguno de estos. Tampoco es algo extraño o inconveniente, considerando que el inglés, especialmente en el ámbito de cómputo e informática, es el lenguaje más utilizado.

No existen libros especializados de autores mexicanos al respecto. La gran mayoría del material consultado, en especial los libros, son de autores extranjeros. No existe un libro en México especializado en esta temática. Hay material relacionado con el análisis y diseño de sistemas pero no con la visión específica de la ingeniería de requerimientos.

Internet fue una fuente de información muy importante. En Internet se encontró suficiente información. Mediante el acceso a diversos sitios se obtuvo información actualizada y de gran interés, por ejemplo, sitios de empresas proveedoras de herramientas para la administración de requerimientos, universidades extranjeras, revistas en línea e incluso hasta seminarios por Web.

Existe suficiente material sobre el tema. Se observaron dos tendencias en este: material académico-formal, y material comercial. Del primero cabe mencionar es información de aproximadamente 10 años atrás. El material más reciente tiene un enfoque comercial, es publicado por empresas proveedoras de herramientas software para apoyar el ciclo de desarrollo de software.

EL IEEE (Instituto de Ingenieros Eléctrico y Electrónicos) es una fuente clave en el tema. Fue precisamente este instituto quien formalizó el concepto de Ingeniería de requerimientos. Existen diversos y muy importantes materiales editados al respecto por el IEEE, está conformado por muchas áreas de especialidad, una de las cuales es el cómputo.

Los seminarios por Internet (webminars) son una nueva e importante alternativa de acceso a la información. De gran interés y muy prácticos resultaron los seminarios por Internet que consisten en presentaciones (con audio y video) que ofrecen organizaciones y personas expertas en diversos temas de valor, y que resultan gratuitos y de fácil acceso para los usuarios del Internet.

El marco teórico fue trabajado prácticamente en todo el desarrollo del trabajo. Desde la concepción inicial del trabajo, hasta las últimas fases del mismo, se estuvo realizando labor de recopilación de información, estando al tanto de las tendencias en el tema. Cabe mencionar que se describieron los materiales más relevantes y claves para la investigación.

Se tuvo la oportunidad de asistir a cursos y eventos relacionados. Por el interés que siempre he tenido sobre el tema y para contribuir a este trabajo de investigación, tuve la oportunidad de asistir a diversos eventos de interés sobre el tema que fueron de gran utilidad para el contenido del trabajo.

EL IEEE, Rational y Karl Wiegers, las principales fuentes. De todas las fuentes consultadas se obtuvo información de interés, no obstante considero que estas tres fueron las principales y las más conocidas. La IEEE por ser quien formalizó el concepto; Rational por el éxito de sus herramientas y Karl Wiegers por generar libros artículo actuales y de gran interés.

6.3 Capítulo 3 (Marco Conceptual)

Los autores coinciden en la problemática. Todos los autores y estudios coinciden en que una de las principales causas de los problemas en el desarrollo de software tiene que ver con requerimientos. Esto complementa lo detectado en el Marco Problemático y justifica en alguna medida la realización de este trabajo.

Administración vs. Ingeniería de Requerimientos. Hoy en día y comercialmente es más conocido el término "Administración de Requerimientos" que "Ingeniería de Requerimientos". Como se mencionó en el trabajo, dentro del concepto formal de Ingeniería de Requerimientos, la administración es complemento de ésta. Estrictamente la ingeniería es la construcción, la administración es la organización y gestión. Sin embargo en la práctica estos términos suelen utilizarse como sinónimos.

Los casos de uso son une técnica efectiva de especificación. La técnica de especificación que se describió con más detalle en el trabajo fueron los Casos de Uso, de UML, ya que en la práctica resultan efectivos. Además diversos autores sobre el tema coinciden en su utilidad y se está convirtiendo en una técnica muy adoptado por la comunidad de desarrollo de sistemas. Existe mucha información sobre de casos de uso, aquí sólo se resumieron sus elementos y forma de construirlos.

Los requerimientos tienen que ver con la calidad del software. Si bien la calidad del software depende de muchos factores, los requerimientos son elemento importante para lograrla. Cuando se habló de la validación, es decir, que los requerimientos digan lo que realmente se quiere, se está colaborando a la calidad. Además la relación de los requerimientos con los modelos de madurez de procesos: CMM y SPICE complementan también esta afirmación.

Hay muchos enfoques de procesos y ciclos de desarrollo de software. Se incluyo la parte de procesos y ciclos de desarrollo por considerar importante la existencia, ubicación e importancia de los requerimientos en cada una de éstos. Hay enfoques de procesos y ciclos de desarrollo, no obstante se observó que comparten características y algunos son adaptaciones de otros. En este trabajo se mencionaron los más representativos, considerando incluso la programación extrema (Extreme Programming), un nuevo enfoque que está teniendo mucho auge para cierto tipo de proyectos.

Existen muchos temas relacionados de interés. Alrededor de la ingeniería de requerimientos hay una serie de temas, aspectos, técnicas, etc. Que por sí mismos pueden ser objeto de nuevos trabajos de investigación y evaluación. Por ejemplo por mencionar algunos: la administración de proyectos de software, las métricas para el tamaño de los sistemas de software, los modelos y estándares de calidad, las técnicas de obtención y especificación, el análisis de las herramientas de software existente, entre otros.

6.4 Capítulo 4 (Marco Metodológico)

Hubo respuestas muy interesantes que podrían ser analizadas como un tema de investigación específico. Como resultado de la aplicación del cuestionario, los encuestados expresaron una serie de inquietudes y aspectos de interés que bien podrían ser tema de otras investigaciones, no sólo de aspectos informáticos sino también administrativos y psicológicos.

Se confió en el juicio de los encuestados. La hipótesis fue comprobada con base en las opiniones de los encuestados, las cuales resultaron de gran interés y aportaron elementos interesantes.

Posteriormente se desea ahondar en algunas respuestas de los encuestados que salen del alcance del Marco Metodológico de este trabajo. El cuestionario fue aplicado con un objetivo específico, pero vale la pena recopilar información de aspectos relacionados que aportarían resultados de gran valor para conocer la situación actual del software en México.

Los resultados del Marco Metodológico coinciden otros estudios especializados. Por ejemplo, el *Standish Group* sugiere que los cuatro factores de éxito para un proyecto de Tecnologías de Información son: involucrar al cliente/usuario, apoyo de la alta dirección, definición clara de los requerimientos y desarrollar una adecuada planeación; aspectos que los encuestados mencionaron explícitamente.

El involucramiento del usuario es clave. La participación de los usuarios fue un aspecto que mencionaron los encuestados en varias ocasiones. Por esa razón se escribió un artículo al respecto como parte del Marco Instrumental.

6.5 Capítulo 5 (Marco Instrumental)

El artículo escrito resulto de interés para el área editorial de la Facultad de Contaduría y Administración. Se trato proyectar un enfoque ligero y no técnico para que resultara comprensible y de interés para la comunidad de la FCA en general. Al hablar de los usuarios y de los sistemas informáticos, la gran mayoría nos sentimos identificados de alguna manera. Se enfoco a la importancia de la participación los usuarios, debido a que resaltó como un punto de inquietud en la mayoría de los encuestados, e incluso los estudios que existen al respecto lo consideran.

Se pretende mejorar el sitio Web. Como se planteó en el Marco Instrumental, actualmente se encuentra una primera versión del sitio Web sobre Ingeniería de Requerimientos en México, no obstante se pretende crecer, debido a que sería una referencia de interés para la comunidad de personas relacionadas con el desarrollo de sistemas.

La difusión del tema debe de ser continua. Por diversos medios como el sitio Web y la participación en eventos, es de mi interés difundir más información sobre la temática ye esto debe hacerse de una manera continua.

6.6 Conclusiones Generales

La hipótesis se comprobó. Con base en las opiniones de los encuestados, la hipótesis tuvo un 80% de aceptación. Definitivamente las actividades, técnicas y herramientas de la ingeniería de requerimientos permiten desarrollar sistemas que realicen la funcionalidad requerida por el usuario, una mejor estimación de los proyectos, y mejor comunicación entre los involucrados, principalmente.

El tema despertó interés en otras personas. Las diversas fuentes que se consultaron, el apoyo solicitado para la realización de los cuestionarios, y la asistencia a eventos relacionados, fueron actividades que influyeron en que otros compañeros y colegas reflexionaran sobre la importancia de esta temática y me solicitarán información al respecto.

El trabajo final ha sido solicitado por diversos interesados. En complemento con el punto anterior, cabe mencionar que existen diversos interesados (sobre todo personas que amablemente contestaron los cuestionarios) en conocer el trabajo final, por lo que será en su momento, publicado en el Web y se enviarán las referencias de clasificación asignadas por la biblioteca de la Facultad de Contaduría y Administración.

La ingeniería de requerimientos es una buena área de especialización para un licenciado en Informática. Como se mencionó en varias ocasiones, el perfil del licenciado en Informática es idóneo para ser el intermediario entre los usuarios y el equipo de desarrollo, y hacer por lo tanto ingeniería de requerimientos. Cuenta con los conocimientos técnicos y administrativos que por un lado le permiten comprender los procesos de una organización y por otro lado diseñar soluciones informáticas adecuadas.

Es posible alcanzar una mayor especialización en tema. Si bien uno de los objetivos personales de este trabajo fue precisamente especializarme en el tema, el cual considero en cierta medida está cumplido, la labor no termina aquí. Es necesario mantenerse al tanto de las nuevas tendencias y contribuir a la verdadera aplicación de estas técnicas y herramientas.

El trabajo constituye una valiosa aportación para los involucrados en el desarrollo de sistemas. Para cualquier interesado en el tema de requerimientos este trabajo puede servirles de referencia para tener una visión más adecuada de todo lo que implica la ingeniería de requerimientos, y que no es sólo obtener información de los usuarios. Puede además, despertar el interés para la especialización en tema e incluso servir como guía para diversos aspectos del proceso de desarrollo de software.

El desarrollo de un trabajo de tesis no es sencillo. Para llegar al término de trabajo hubo que invertir mucho esfuerzo y dedicación. No solo se limita a hacer una investigación documental, si no que se realizan esfuerzos complementarios como son: entrevistas, visitas, observaciones, relacionarse con colegas y expertos en el tema, asistir a diversos eventos, y en este caso en particular, aplicación de algunas técnicas aquí expuestas en la Dirección de Sistemas de la UNAM.

Aplicación en el ámbito laboral. Lo mencionado en este trabajo no sólo se limita a lo teórico, sino que se aportan elementos y conocimientos con los que se ha trabajado en la vida real, y que por lo tanto se ha comprobado su utilidad, beneficios e importancia en el desarrollo de sistemas.

Los especialistas en Ingeniería de Requerimientos serán ampliamente demandados con el crecimiento de la industria de software en México. Con las iniciativas actuales de impulsar el crecimiento de la industria de software y considerando a una fábrica de software como una entidad donde con base en especificaciones se desarrollan productos de software, habrá demanda de especialistas en esta área. Alguien debe de elaborar esas especificaciones para que las fábricas las implementen.

7 GLOSARIO

Término	Significado
Actor	Representa cualquier cosa que interactúe con un sistema.
Administración de la configuración	Conjunto de actividades relacionadas con la identificación y control de las características y cambios en un punto del tiempo, de los productos generados durante el desarrollo de software.
AMCIS	Asociación Mexicana para la Calidad en Ingeniería de Software.
Calidad	EL grado en que un sistema, componente o proceso cumple con los requerimientos especificados y con las expectativas de un cliente o usuario.
Característica	Particularidad. Una característica es un servicio que el sistema proporciona para satisfacer una o más necesidades de los involucrados.
Caso de Uso	Un caso de uso es una descripción de un conjunto de acciones que el sistema realiza para ofrecer algún resultado de valor a un actor en particular.
Ciclo de vida del software	El período de tiempo que empieza cuando un producto de software se concibe y termina cuando el software no está disponible más para el uso.
CMM	*Capability Maturity Model*. Modelo de madurez de capacidades para evaluar los procesos de desarrollo de software de una organización. Consta de 5 niveles y fue desarrollado por el *Software Engineering Institute*.
CRC	*Class-Responsability-Collaboration*. Técnica originalmente para identificar las responsabilidades de las clases en el análisis orientado a objetos. Consisten en tarjetas de 5 x 6 cm., en las que cada participante asume el papel de una clase y anota sus responsabilidades e interacciones.
CUA	*Common User Access.* Conjunto de estándares para interfaces de usuario desarrollado por IBM.
Estándar	Los requisitos obligatorios empleados y reforzados para lograr un enfoque uniforme y disciplinado al desarrollo de software.
Éxito	Resultado feliz de un negocio o proyecto.
GUI	*Graphic User Interface.* Interfaces gráfica de usuario. Término aplicado

a los sistemas en ambiente gráfico.

ID Identificador único de un requerimiento. Clave.

IEEE *Institute of Electrical and Electronics Engineers.*

Ingeniería Se refiere al uso de principios de manera sistemática para obtener diseños que cubran los requerimientos considerando restricciones de recursos y bajo condiciones de incertidumbre.

Ingeniería de requerimientos La ciencia y disciplina relacionada con el análisis y documentación de requerimientos, incluyendo análisis de necesidades, análisis de requerimientos y especificación de requerimientos. También proporciona los mecanismos adecuados para facilitar las actividades de análisis, documentación y verificación.

ISO *Internacional Organization for Standardization.*

JAD *Joint Application Development.* Método de obtención de requerimientos que reúne a los analistas, desarrolladores y representantes de los clientes/usuarios en una mesa de trabajo.

Línea base Un conjunto detallado de características, o requerimientos que se pretenden entregar en una versión específica de la aplicación.

Método Un conjunto razonablemente completo de reglas y criterios que establecen una forma precisa y repetible de realizar una tarea y llegar a un resultado deseado.

Metodología Una colección de métodos, procedimientos y normas.

OMT *Object Modeling Technique*

Técnica de modelado de objetos desarrollada principalmente por Rumbaugh cuyos elementos se integraron posteriormente en UML.

OOA/OOD *Object Oriented Analysis/ Object Oriented Design*

Análisis/diseño orientado a objetos.

PAISLey Es un lenguaje de especificación ejecutable el cual es acompañado por un conjunto de métodos de especificación, técnicas de análisis y herramientas de software. Los lenguajes de especificación ejecutables son lenguajes especializados los cuales pretenden modernizar el proceso de desarrollo de software.

Proyecto Un conjunto de tareas que requiere la coordinación y distribución de un conjunto de esfuerzos y recursos para lograr un objetivo en un tiempo

finito.

Proyecto de (desarrollo de) software Una tarea que requiere une esfuerzo concertado que se enfoca a analizar y especificar, diseñar, desarrollar, probar y/o mantener los componentes del software de un sistema y la documentación asociada.

PSL/PSA *Problem Statement Language/Problem Statement Analyzer.*

PSP *Personal Software Process.*

RAD Rapid Application Development. Un sistema de programación que permite a los programadores construir programas de manera rápida. En general, las herramientas RAD se basan en interfaces gráficas de usuario.

Rastreo/ Rastreabilidad Término que hace referencias a las ligas y dependencias lógicas entre requerimientos y otros componentes derivados del desarrollo de software.

Requerimiento Un requerimiento es una condición o capacidad que debe satisfacer un sistema.

Riesgo Peligro. Posibilidad de que ocurra un contratiempo o evento indeseado.

Rol Una unidad de responsabilidades definidas que pueden ser asumidas por uno o más individuos.

RSL/REVS *Requirements Statement Language/Requirement Engineering and Validation System*

RUP *Rational Unified Process.* Proceso de desarrollo de software bidimensional desarrollado por Ivar Jacobson en Ericsson y posteriormente refinado en Rational.

SA/SADT *Structured Analysis and Desing Techniques.* Técnicas estructuradas de análisis y diseño.

Sistema Conjunto de hardware, software, datos, personas, facilidades y procedimientos organizados para lograr algún objetivo en común.

Software Término general para el conjunto de programas usados para operar computadoras y dispositivos relacionados. Elemento primordial en los sistemas informáticos juntos con el hardware y la información. (v. sistema)

SPICE Modelo de referencia que describe los procesos que una organización puede realizar ya sea para adquirir, proveer, desarrollar, operar,

evolucionar y soportar algún software, así como los atributos que caracterizan la capacidad de cada uno de los procesos.

SREM *Software Requirements Engineering Methodology.*

Metodología de de ingeniería de requerimientos de software desarrollada por TRW Corp.

SRS *Software Requirements Specification*

Es un documento que contiene una descripción completa de lo que hará el software, sin describir cómo lo hará. <IEEE>

SSADM *Structured Systems Analysis and Design Methodology*

Metodología usada en el análisis y diseño del desarrollo de sistemas. Usa tres técnicas clave: modelado de datos, modelado de flujo de datos y modelado de entidades-eventos.

Stakeholder Individuos y organizaciones involucrados en un proyecto de desarrollo de software y que tienen influencia directa o indirecta sobre los requerimientos, o cuyos intereses se ven afectados por el proyecto.

Storyboard Presentación animada de algún producto.

TSP *Team Software Process.*

UML Unified Modeling Language. Lenguaje que proporciona una notación estándar para el modelado de software.

Validación Proceso de asegurar que lo que se está construyendo corresponde con los realmente requerido, que sea completo, consistente y de acuerdo con los requisitos.

Verificación Proceso de determinar si los productos de una determinada fase del ciclo de desarrollo de software, cumplen o no los requerimientos establecidos durante el inicio de la fase.

8 BIBLIOGRAFÍA

8.1 Libros

A user's guide for defining software requirements
Carolyn Shamlin
Editorial Massachussets
Wellesley, 1989
ISBN 0-89435-278-4

El proceso unificado de desarrollo de software
Ivar Jacobson, Grady Booch y James Rumbaugh
Editorial Addison Wesley
Primera edición en español. 2000

IEEE Guide to Software Requirements Specifications. ANSI-IEEE 830-1984
IEEE. 1984

Ingeniería de software. Un enfoque práctico
Roger S. Pressman
Editorial Mc Graw Hill.
México. Cuarta edición, 1998
84-7615-222-1

Managing software requirements
Dean Leffingwell y Don Wodrig
Ed. Addison Wesley
Massachussets, Estados Unidos. Primera edición, 2000
ISBN 0201615932

Modelo de referencia de procesos y capacidades, modelo de evaluación de procesos y guía de indicadores según ISO 15504 (SPICE)
M. en C. María del Pilar Angeles
Asociación Mexicana para la Calidad en Ingeniería de Software (AMCIS)
Noviembre, 2000.

Requirements engineering. A good practice guide
Ian Sommerville y Pete Sawyer
Editorial John Wiley & Sons
Chichester, England. Primera edición, 1997
ISBN 0-471-97444-7

Software requirements & specifications: a lexicon of practice, principles and prejudices
Michael Jackson
Editorial Addison-Wesley
1a. Edición. 1995
ISBN 0201877120

Software requirements : objects, functions, and states
Alan M. Davis
Editorial Prentice-Hall
Englewood Cliffs, New Jersey. Primera edición, 1993.
ISBN 0-13-805763-X

Software Requirements: Analysis and specification
Michael Davis
Editorial Prentice Hall
New Jersey, 1990
ISBN 0-13-824673-4

Software that works
Michael Ward
Editorial Academic
San Diego, 1990.
ISBN 0-12-735040-3

Specification and transformation of programs. A formal approach to software development
Helmut A. Partsch
Editorial *Springer-Verlag Berlin Heidelberg*
New York. 1ª. Edición. 1990
ISBN 3-540-52356

System and software requirements engineering
Richard H. Thayer y Merlin Dorfman
IEEE Computer Society
1990
ISBN 0-8186-8921-8

8.2 Revistas

Software Development Magazine
http://www.sdmagazine.com

STQE Magazine
Http://www.stqemagazine.com

8.3 Internet

http://www.computer.org
http://www.construx.com
http://www.cutter.org
http://www.incose.org
http://www.processimpact.com
http://www.processimprovement.com
http://www.rational.com
http://www.rej.com
http://www.rspa.com
http://www.sei.cmu.edu
http://www.spc.ca
http://www.telelogic.com
http://www.therationaledge.com
http://www.itera.com.mx
http://biblioteca.dgsca.unam.mx
http://www.dgbiblio.unam.mx