Ingeniería de Requisitos
umh2818-TADS

En la ingeniería de sistemas y la ingeniería de software, la **Ingeniería de requisitos** o **Ingeniería de requerimientos**[1] comprende todas las tareas relacionadas con la determinación de las necesidades o de las condiciones a satisfacer para un software nuevo o modificado, tomando en cuenta los diversos requisitos de las partes interesadas, que pueden entrar en conflicto entre ellos.

Muchas veces se habla de requerimientos en vez de requisitos; esto se debe a una mala traducción del inglés. La palabra *requirement* debe ser traducida como requisito, mientras que requerimiento se traduce al inglés como *request*.

El propósito de la ingeniería de requisitos es hacer que los mismos alcancen un estado óptimo antes de alcanzar la fase de diseño en el proyecto. Los buenos requisitos deben ser medibles, comprobables, sin ambigüedades o contradicciones, etc.

1 Implicaciones

La Ingeniería de Requisitos implica todas las actividades del ciclo de vida dedicadas a:

- La educción (a veces llamada "elicitación", debido a una mala traducción de "elicitation") de los requisitos de usuario.
- El análisis y negociación de requisitos para derivar requisitos adicionales.
- La documentación de los requisitos como especificación.
- La validación de los requisitos documentados contra las necesidades de usuario.
- Así como los procesos que apoyan estas actividades.

2 Fases de implementación

Desde un punto de vista conceptual, las actividades son de cinco clases.

- **Obtener requisitos:** a través de entrevistas o comunicación con clientes o futuros usuarios, para saber cuáles son sus expectativas.
- **Analizar requisitos:** detectar y corregir las carencias o falencias comunicativas, transformando los requisitos obtenidos de entrevistas y requisitos, en condiciones apropiadas para ser tratados en el diseño.
- **Documentar requisitos:** igual que todas las etapas, los requisitos deben estar debidamente documentados.
- **Verificar los requisitos:** consiste en comprobar la implementación de los requisitos.
- **Validar los requisitos:** comprobar que los requisitos implementados sean funcionales para lo que inicialmente se construyó el producto.

3 Técnicas principales

La ingeniería de requisitos puede ser un proceso largo y arduo para el que se requiere de habilidades psicológicas. Los nuevos sistemas cambian el entorno y las relaciones entre la gente, así que es importante identificar a todos los actores involucrados, considerar sus necesidades y asegurar que entienden las implicaciones de los nuevos sistemas. Los analistas pueden emplear varias técnicas para obtener los requisitos del cliente. Históricamente, esto ha incluido técnicas tales como las entrevistas, o talleres con grupos para crear listas de requisitos. Técnicas más modernas incluyen los prototipos, y utilizan casos de uso. Cuando sea necesario, el analista empleará una combinación de estos métodos para establecer los requisitos exactos de las personas implicadas, para producir un sistema que resuelva las necesidades del negocio.

3.1 Entrevistas

Las entrevistas son un método común. Por lo general no se entrevista a toda la gente que se relacionará con el sistema, sino a una selección de personas que represente a todos los sectores críticos de la organización, con el énfasis puesto en los sectores más afectados o que harán un uso más frecuente del nuevo sistema.

3.2 Talleres

Los requisitos tienen a menudo implicaciones cruzadas desconocidas para las personas implicadas individuales y que a menudo no se descubren en las entrevistas o quedan incompletamente definidas durante la misma. Estas implicaciones cruzadas pueden descubrirse realizando en un ambiente controlado, talleres facilitados por un analista del negocio, en donde las personas implicadas participan en discusiones para descubrir requisitos, analizan sus detalles y las implicaciones cruzadas. A menudo es útil la selección de un secretario dedicado a la documentación de la discusión, liberando al analista del negocio para centrarse en el proceso de la definición de los requisitos y para dirigir la discusión.[*cita requerida*]

Existen dos técnicas de este tipo que son las más utilizadas: *Brainstorming* (Lluvia de ideas) y JAD (*Joint Application Development*, Diseño de Aplicación Conjunta). La diferencia que existe entre ambas técnicas es que durante la tormenta de ideas se tiene como objetivo generar una gran cantidad de ideas, es desestructurada y la información que se obtiene puede ser visual, textual ó coloquial; mientras que en el JAD el taller es ordenado, la información que se obtiene es visual y su objetivo es generar requisitos y la interfaz del sistema.[*cita requerida*]

Durante una sesión de **Lluvia de ideas**, todos los participantes pueden aportar distintas ideas en un ambiente libre de prejuicios. Ningún participante debe juzgar negativamente la propuesta de otros, sino que se anotan todas las ideas en una pizarra y serán evaluadas al final de la sesión. El principio básico es no descartar de manera apresurada ningún planteo, de modo que existe la posibilidad de que surjan otras ideas derivadas de la idea original y se generan varios puntos de vista del problema.[*cita requerida*]

En el Joint application development se trabaja directamente sobre los documentos a generar, las temáticas que se tratan durante las reuniones siguen un esquema y se busca que la misma sea ordenada y racional. Se define una agenda con los puntos principales a tratar durante la jornada. Este tipo de taller tiene el inconveniente de que es muy difícil poder reunir a todas los participantes, es costoso y generalmente es necesaria más de una reunión para establecer los requisitos del sistema.[*cita requerida*]

3.3 Forma de contrato

En lugar de una entrevista, se pueden llenar formularios o contratos indicando los requisitos. En sistemas muy complejos éstos pueden tener centenares de páginas.

3.4 Objetivos medibles

Los requisitos formulados por los usuarios se toman como objetivos generales, a largo plazo, y en cambio se los debe analizar una y otra vez desde el punto de vista del sistema hasta determinar los objetivos críticos del funcionamiento interno que luego darán forma a los comportamientos apreciables por el usuario. Luego, se establecen formas de medir el progreso en la construcción, para evaluar en cualquier momento qué tan avanzado se encuentra el proyecto.

3.5 Prototipos y Casos de uso

Un prototipo es una pequeña muestra, de funcionalidad limitada, de cómo sería el producto final una vez terminado. Ayudan a conocer la opinión de los usuarios y rectificar algunos aspectos antes de llegar al producto terminado.

Un caso de uso es una técnica para documentar posibles requisitos, graficando la relación del sistema con los usuarios u otros sistemas. Dado que el propio sistema aparece como una caja negra, y sólo se representa su interacción con entidades externas, permite omitir dichos aspectos y determinar los que realmente corresponden a las entidades externas. El objetivo de esta práctica es mejorar la comunicación entre los usuarios y los desarrolladores, mediante la prueba temprana de prototipos para minimizar cambios hacia el final del proyecto y reducir los costes finales. Esta técnica se enfrenta a los siguientes peligros potenciales.

- A los directivos, una vez que ven un prototipo, les cuesta comprender que queda mucho trabajo por hacer para completar el diseño final.
- Los diseñadores tienden a reutilizar el código de los prototipos por temor a “perder el tiempo” al comenzar otra vez.
- Los prototipos ayudan principalmente a las decisiones del diseño y de la interfaz de usuario. Sin embargo, no proporcionan explícitamente cuáles son los requisitos.
- Los diseñadores y los usuarios finales pueden centrarse demasiado en el diseño de la interfaz de usuario y demasiado poco en producir un sistema que sirva el proceso del negocio.

Los prototipos pueden ser: diagramas, aplicaciones operativas con funcionalidades sintetizadas. Los diagramas, en los casos donde se espera que el software final tenga diseño gráfico, se realizan en una variedad de documentos de diseño gráficos y a menudo elimina todo el color del diseño del software (es decir utilizar una gama de grises). Esto ayuda a prevenir la confusión sobre la apariencia final de la aplicación.

4 Especificación de requisitos del software

Una especificación de requisitos del software es una descripción completa del comportamiento del sistema a desarrollar. Incluye un conjunto de casos de uso que describen todas las interacciones que se prevén que los usuarios tendrán con el software. También contiene requisitos no funcionales (o suplementarios). Los requisitos no funcionales son los requisitos que imponen restricciones al diseño o funcionamiento del sistema (tal como requisitos de funcionamiento, estándares de calidad, o requisitos del diseño).

Las estrategias recomendadas para la especificación de los requisitos del software están descritas por IEEE 830-1998. Este estándar describe las estructuras posibles, contenido deseable, y calidades de una especificación de requisitos del software.

Los requisitos se dividen en tres:

- Funcionales: son los que el usuario necesita que efectúe el software. Normalmente se identifican como los requisitos que responden a la pregunta ¿qué hace? e.g. El sistema debe emitir un comprobante al asentar la entrega de mercadería.
- No funcionales: son los "recursos" para que trabaje el sistema de información (redes, tecnología). Ej: el soporte de almacenamiento a usar debe ser MySQL. Normalmente se identifican como los requisitos que responden a la pregunta ¿cómo lo hace? e.g. rápido, fácil etc.
- Empresariales u Organizacionales: son el marco contextual en el cual se implantará el sistema para conseguir un objetivo macro. Ej: abaratar costos de expedición.

5 Identificación de las personas involucradas

Debido a que los cambios que introduce un sistema nuevo tienden a afectar a más de un tipo de usuario, los analistas de requisitos han de tomar en consideración a todos los implicados para que se obtengan y depuren sus requisitos de la forma más fidedigna posible. Entre las personas implicadas hay que considerar:

- Organizaciones que integran la organización del analista que está diseñando el sistema
- Organizaciones o sistemas de respaldo
- Dirección
- Usuarios.

6 Problemas

6.1 Relacionados con las personas involucradas

Las vías que pueden dificultar la determinación de los requisitos son:

- Los usuarios no tienen claro lo que desean
- Los usuarios no se involucran en la elaboración de requisitos escritos
- Los usuarios insisten en nuevos requisitos después de que el coste y la programación se hayan fijado.
- La comunicación con los usuarios es lenta
- Los usuarios no participan en revisiones o son incapaces de hacerlo.
- Los usuarios no comprenden los problemas técnicos
- Los usuarios no entienden el proceso del desarrollo

Esto puede conducir a la situación donde las exigencias del consumidor cambian, incluso cuando el desarrollo del producto ya está en marcha.

6.2 Relacionados con los analistas

La correcta redacción de las Especificaciones de requisitos del Software es imprescindible para el correcto desarrollo del proyecto. Por ello, en su redacción hay que evitar:

- Uso de terminología ambigua en la redacción de los documentos de requisitos
- Sobreespecificación de los requisitos
- Escritura poco legible, voz pasiva, abuso de negaciones
- Uso de verbos en condicional, expresiones subjetivas
- Ausencia de términos y verbos del dominio de la aplicación

6.3 Relacionados con los desarrolladores

Los problemas posibles causados por los desarrolladores durante análisis de requisitos son:

- El personal técnico y los usuarios finales pueden tener diversos vocabularios y pueden llegar a creer incorrectamente que están de acuerdo, no dándose cuenta del desacuerdo hasta que se provee el producto final.

- Los desarrolladores pueden intentar encajar el sistema en un modelo existente, en vez de desarrollar un sistema adaptado a las necesidades del cliente.
- El análisis de requisitos se puede realizar a menudo por los ingenieros o programadores, en vez de personal con las habilidades de relación con la gente y el conocimiento apropiados para entender las necesidades de un cliente correctamente.

7 Soluciones aplicadas

Una solución aplicada en los problemas de comunicaciones ha sido emplear a especialistas en análisis del negocio o del sistema.

Las técnicas introducidas en los años 90 tienden al uso de prototipos, lenguaje unificado de modelado, casos de uso, y el desarrollo ágil de software.

Otros tipos de herramientas aplicadas para salvar las diferencias entre los usuarios y las organizaciones de tecnología de la información y que permiten la comprobación de las aplicaciones son:

- pizarras electrónicas para bosquejar los algoritmos y para probar alternativas
- capacidad de capturar la lógica del negocio y los datos necesarios
- capacidad de generar los prototipos que imitan fielmente el producto final
- interactividad
- la capacidad para agregar requisitos contextuales y otro comentarios
- capacidad para que usuarios remotos y distribuidos operen con el prototipo

Por último, se requieren herramientas que permitan medir, de forma objetiva, la calidad de una especificación de requisitos.

8 Referencias

[1] Investigación IT (28 de octubre de 2013). «¿Requisitos o requerimientos?». Archivado desde el original el 23 de noviembre de 2015. Consultado el 2 de noviembre de 2013.

8.1 Bibliografía

- McConnell, Steve (1996). Rapid Development: Taming Wild Software Schedules, 1st ed., Redmond, WA: Microsoft Press. ISBN 1-55615-900-5.
- Wiegers, Karl E. (2003). Software Requirements 2: Practical techniques for gathering and managing requirements throughout the product development cycle, 2nd ed., Redmond: Microsoft Press. ISBN 0-7356-1879-8.
- Landgraf, Katja (2011) Requirement Management in Product Development, Symposion Publishing ISBN 978-3-939707-84-4
- Andrew Stellman and Jennifer Greene (2005). Applied Software Project Management. Cambridge, MA: O'Reilly Media. ISBN 0-596-00948-8.
- IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications - Description

9 Origen del texto y las imágenes, colaboradores y licencias

9.1 Texto

- **Ingeniería de requisitos** *Fuente:* https://es.wikipedia.org/wiki/Ingenier%C3%ADa_de_requisitos?oldid=90147106 *Colaboradores:* Enric Naval, Banfield, Gabriel Acquistapace, Roberto Fiadone, Isha, Gacq, VolkovBot, Muro Bot, Abel.orian, Mafores, Farisori, UA31, AVBOT, Adelpine, Diegusjaimes, Luckas-bot, Acasson, ArthurBot, Xqbot, Nail2001, Shadow440, EmausBot, ChuispastonBot, WikitanvirBot, M0000g, Elvisor, Helmy oved, Addbot, 10xor, Ejfdelgado, Jarould y Anónimos: 46

9.2 Imágenes

- **Archivo:Commons-emblem-question_book_yellow.svg** *Fuente:* https://upload.wikimedia.org/wikipedia/commons/d/dd/Commons-emblem-question_book_yellow.svg *Licencia:* CC BY-SA 3.0 *Colaboradores:* + *Artista original:* GNOME icon artists, Linfocito B

9.3 Licencia del contenido

- Creative Commons Attribution-Share Alike 3.0