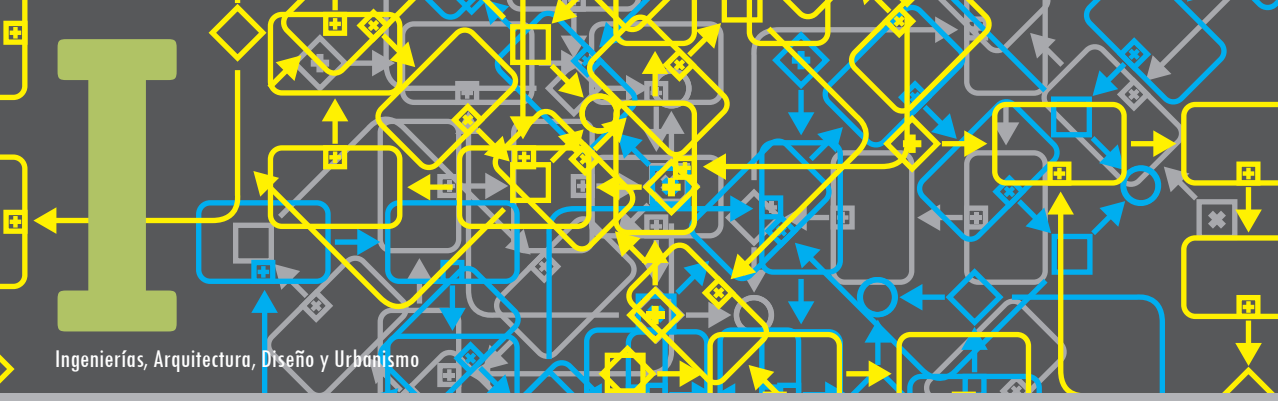


Ingeniería de requerimientos

Colección Libros de Investigación
Vicerrectoría de Investigación
Pontificia Universidad Javeriana

Vanesa Carolina Loaiza Carvajal
Laura Catalina Zorro Jiménez
Miguel Eduardo Torres Moreno
Rafael Andrés González Rivera
María Patricia Amórtegui Vargas

INGENIERÍA DE REQUERIMIENTOS

INGENIERÍA DE REQUERIMIENTOS

Vanesa Carolina Loaiza Carvajal
Laura Catalina Zorro Jiménez
Miguel Eduardo Torres Moreno
Rafael Andrés González Rivera
María Patricia Amórtegui Vargas

Pontificia Universidad JAVERIANA Bogotá

editorial Pontificia Universidad JAVERIANA 25 AÑOS

Colección Libros de Investigación
Vicerrectoría de Investigación

Primera edición: Bogotá, D. C., febrero de 2018
ISBN: 978-958-781-172-8
Número de ejemplares: 300
Impreso y hecho en Colombia
Printed and made in Colombia

Editorial Pontificia Universidad Javeriana
Carrera 7 n.° 37-25, oficina 1301
Teléfono 3208320 ext. 4752
www.javeriana.edu.co/editorial
editorialpuj@javeriana.edu.co
Bogotá, D. C.

Corrección de estilo:
Ella Suárez

Diagramación y montaje de cubierta:
Nathalia Rodríguez

Impresión:
Javegraf

Pontificia Universidad Javeriana | Vigilada Mineducación. Reconocimiento como Universidad: Decreto 1297 del 30 de mayo de 1964. Reconocimiento de personería jurídica: Resolución 73 del 12 de diciembre de 1933 del Ministerio de Gobierno.

MIEMBRO DE LA RED DE EDITORIALES UNIVERSITARIAS DE AUSJAL
AUSJAL
www.ausjal.org

Torres Moreno, Miguel Eduardo, autor
Ingeniería de requerimientos / Miguel Eduardo Torres Moreno [y otros cuatro]. -- Primera edición. -- Bogotá : Editorial Pontificia Universidad Javeriana, 2017. (Colección libros de investigación)

182 páginas : ilustraciones ; 24 cm
Incluye referencias bibliográficas.
ISBN : 978-958-781-172-8

1. Ingeniería de requisitos. 2. Ingeniería de software. 3. Desarrollo de programas para computador. I. González Rivera, Rafael Andrés, autor. II. Loaiza Carvajal, Vanesa Carolina, autora. III. Zorro Jiménez, Laura Catalina, autora. IV. Amórtegui Vargas, María Patricia, autora. V. Pontificia Universidad Javeriana.

CDD 001.6425 edición 19

Catalogación en la publicación - Pontificia Universidad Javeriana. Biblioteca Alfonso Borrero Cabal, S.J.

inp 05 / 12 / 2017

CONTENIDO

FIGURAS

TABLAS

PRÓLOGO

Miguel Eduardo Torres Moreno

La ingeniería de requerimientos es, al igual que la ciencia que la agrupa (la ingeniería de software), una ciencia bastante joven con quizás no más de cuarenta años. Si bien la problemática de los requerimientos de software siempre fue relegada a ser una parte no muy trascendental del proceso de construcción, donde dependiendo del paradigma de desarrollo que se use se aplicaban algunas técnicas que buscaban dar una visión de alto nivel para el equipo de desarrollo y de alguna manera clarificar los resultados esperados por parte del cliente, solo hasta los años ochenta, con el trabajo tanto de Alexander y Stevens [1] como de Wiegers [2], se comenzó a profundizar en el papel de los requerimientos en el momento de comprender, abstraer y modelar el problema y, por supuesto, usarlos como herramienta de gestión y control del proyecto. Esto abrió un nuevo mundo a los ingenieros de software, quienes ahora debían acercase más al negocio y a las necesidades reales de los clientes, con lo cual podían dar una respuesta más acertada a dichas necesidades.

La profesionalización de la ingeniería de software durante los años setenta y ochenta dio como resultado la aparición de estándares (por ejemplo, los de IEEE/ISO) y mejores prácticas (Modelo de Capacidad y Madurez [CMM], Mejoramiento de Procesos de Software [SPICE], Proceso Unificado Racional [RUP], etc.), tendientes a mejorar la calidad de los productos, así como asegurar la repetición exitosa de los procesos y su estandarización. Estos estándares y prácticas proporcionan recomendaciones y soluciones a problemas comunes del desarrollo de software, que para el caso de la ingeniería de requerimientos es frecuentemente la traducción de un problema de negocio a una solución de software, de tal manera que el equipo de desarrollo entienda cuál era el comportamiento deseado del producto final.

Más recientemente, gracias a las metodologías de desarrollo ágil, se propuso acercar al cliente al equipo de desarrollo, de manera que esa comprensión del negocio siempre estuviera al alcance del equipo; sin embargo, aún se siguen usando modelos y abstracciones basadas en paradigmas de desarrollo que acercan el negocio al software esperado.

Observamos entonces dos caminos para llegar al desarrollo de software actual: los tradicionales y los ágiles, que si bien filosóficamente se perciben diferentes, son simplemente formas de ver la vida de un proyecto de desarrollo de software, siempre pensando en el "depende": del cliente, del alcance, de la tecnología, de los atributos de calidad esperados del producto y tantas otras variables que llevan a preguntarse: ¿cuál es la mejor solución?

Otro elemento que ha puesto la atención en la gestión de requerimientos de software es el uso extendido del Project Management Institute (PMI) y su cuerpo de conocimiento (Project Management Body of Knowledge [PMBOK]) [3] alrededor de la gestión de proyectos y en el cual el principal elemento que da forma y caracteriza un proyecto es su alcance (*scope*). Este concepto conecta perfectamente con la idea de la ingeniería de software, donde antes de desarrollar un producto de software, debe existir una verdadera problemática de negocio enmarcada en sus procesos, las personas y sus roles en dichos procesos y el rol de las tecnologías de información que respaldan dichos procesos. El análisis inicial de la organización se desarrolla a partir del concepto *análisis de negocio*, el cual busca comprender las organizaciones, sus problemáticas y la caracterización de su subsistencia en el entorno. Se observa entonces una relación innegable entre los "requerimientos de negocio", el "alcance" en el contexto de la gestión de proyectos y los requerimientos de software.

Este trabajo busca precisamente ayudar a ingenieros industriales y de procesos a comprender de mejor manera cómo expresar sus necesidades en el momento de buscar una solución con tecnologías de la información. Por otra parte, está dirigido a ingenieros de sistemas y de software que están comenzando a comprender que la construcción de software requiere un diálogo más fluido y constante entre las necesidades de un negocio y lo que la tecnología le puede aportar para solucionar dichas necesidades.

Aun cuando no hay una receta o secuencia de pasos totalmente cierta y válida para todos los casos, es importante conocer diferentes estrategias y los aportes que ellas pueden brindar al contexto del proyecto, idiosincrasia del cliente y complejidad del problema. Este es uno de los fuertes del presente texto: mostrar diferentes métodos y estrategias para la identificación, la especificación, el análisis y la gestión aplicable tanto a los requerimientos de negocio como de software.

Este texto presenta, inicialmente, en el capítulo 1 una introducción y justificación de estado del arte alrededor del área de conocimiento de la ingeniería de requerimientos, descrita de tal forma que su uso y aplicación pueda ser más eficiente por parte de un equipo de analistas de sistemas, arquitectos de software o ingenieros industriales.

Posteriormente, los capítulos 2 y 3 presentan los procesos recomendados para la recolección, el análisis, la especificación y la validación de los requerimientos, así como técnicas aplicables y recomendaciones para un uso más eficiente y adecuado de ellas.

A continuación, los capítulos 4 y 5 presentan acciones encaminadas a apoyar la gestión de los requerimientos para asegurar el correcto desarrollo de la solución, así como la descripción de una herramienta de gestión de requerimientos.

Finalmente, se presenta un modelo basado en los elementos presentados a lo largo del texto con el fin de mostrar que los procesos planteados son generales y que pueden y deben ser adaptados en contexto del problema, la tecnología, el usuario, el negocio y la idiosincrasia de la organización, al aplicarlo en un caso de estudio que típicamente se estima como uno de los más complejos, como lo es el entorno hospitalario.

Referencias

[1] I. Alexander y R. Stevens, *Writing Better Requirements*, 1.ª ed. Londres: Addison-Wesley Professional, 2002.

[2] K. Wiegers, "First things first", *Softw. Dev.*, vol. 7, no. 9, pp. 48-53, set. 1999.

[3] Project Management Institute, *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)–Fifth Edition*, 5.ª ed. Newtown Square, Pennsylvania: Project Management Institute, 2013.

CAPÍTULO 1
INTRODUCCIÓN Y JUSTIFICACIÓN

Laura Catalina Zorro Jiménez
Vanesa Carolina Loaiza Carvajal
Miguel Eduardo Torres Moreno

La ingeniería de requerimientos, como área de conocimiento, ha evolucionado en la última década. Originalmente, desde la ingeniería de software y luego desligándose de ella hacia otras áreas, debido a las diversas aplicaciones que presenta para diferentes tipos de proyectos de implementación o implantación de productos. Desde la construcción de software, con el paso del tiempo, ha comenzado a identificarse como una disciplina clave para el éxito en los proyectos de software, pues apoya diferentes procesos como los que se exponen a continuación:

- Mostrar qué resultados quieren los *stakeholders*[1] (clientes, usuarios o participantes) y *stockholders*[2] (inversionistas o personas de la alta gerencia que no necesariamente serán usuarios del producto final).
- Darles oportunidad de opinión a los *stakeholders.*
- Representar diferentes puntos de vista.
- Verificar el diseño del producto.
- Permitir que el cliente pueda aceptar los productos con criterios precisos y medibles.
- Posibilitar que el equipo de desarrollo esté seguro de que está resolviendo los problemas importantes.
- Permitir que los probadores sepan qué probar del producto en contexto de las necesidades reales del cliente.

---

[1] *Stakeholder*: dícese de cualquier persona involucrada en el proceso de desarrollo del sistema [1].

[2] *Stockholder*: dícese de cualquier persona interesada (típicamente desde el punto de vista económico o financiero) en el proceso de desarrollo e implantación del sistema [1].

- Asegurar que los gerentes de proyecto puedan tener información adecuada de su estado y avance.

Para poder presentar la ingeniería de requerimientos es necesario introducir algunos conceptos y elementos que permitan contextualizar esta área. El presente capítulo presenta diferentes conceptos que serán luego puestos en contexto en los capítulos 2 y 3.

1.1. ¿Qué es un requerimiento?

A través de los años se han expuesto diferentes definiciones de la palabra *requerimiento* dentro del campo de la ingeniería de software, que además ha mostrado su permanente evolución. A continuación, se presentan algunas de estas definiciones:

Según Karl E. Wiegers [2], en la industria del software existe un problema: la falta de definiciones que describan aspectos del trabajo de los ingenieros que sean comunes para todos; por lo tanto, no existe una definición universal acerca de qué es un requerimiento. Una definición la presenta Ralph R. Young en su libro *The Requirements Engineering Handbook*, quien menciona que un requerimiento "es un atributo necesario dentro de un sistema, una declaración que identifica la capacidad, característica o factor de calidad de un sistema para que tenga valor y utilidad a un cliente o usuario" [3].

En este caso se tomará como base la definición dada por Sommerville [4], la cual especifica que los requerimientos son la descripción de lo que debería ser implementado [2], de los servicios proporcionados [4] y de cómo debería comportarse el sistema que se va a desarrollar.

Adicional a la anterior definición, se tiene la de Wohlin et ál., la cual indica que un requerimiento es:

> (1) Una condición o capacidad que necesita un usuario para resolver un problema o alcanzar un objetivo, (2) Una condición o capacidad que se debe cumplir o que posea un sistema o componente para satisfacer un contrato, norma, especificación, u otros documentos, impuestos formalmente. Una representación documentada de una condición o capacidad como en (1) o (2). [5]

Los autores ya mencionados indican que no solo son necesidades de usuario las que se deben documentar, sino que también se deben tener en cuenta las políticas organizacionales, el gobierno y los estándares del sector donde se desempeñan. En algunos casos también es útil indicar qué NO hace el sistema, con el fin de apoyar la

definición del alcance que se dará al sistema o al producto que se está implementando. Adicionalmente, es importante tener un conocimiento claro de por qué es necesario el sistema o producto y cómo este afectará a la organización que lo usará; esto generalmente se realiza por medio de estudios de análisis de negocio (*business analysis*).

Finalmente, es importante aclarar que, en la lengua castellana, el término *requerimiento*, según la Real Academia Española, se define como: "la acción y efecto de requerir" [6]; mientras que *requisito* se define como "circunstancia o condición necesaria para algo" [6]. Sin embargo, desde el uso técnico, estos dos términos se usan indistintamente (pero sí de manera consistente); por tanto, en ese contexto y para la lectura subsecuente de este texto usaremos el término *requerimiento* para hacer referencia a una necesidad o un atributo del sistema propuesto.

1.2. Atributos de calidad del requerimiento

La mayor dificultad al usar los requerimientos como el elemento guía en la construcción de un proyecto es precisamente su naturaleza ambigua, dada por la forma en que están representados (típicamente, lenguaje natural). Para que los requerimientos sean realmente útiles en el contexto del desarrollo y gestión de un proyecto, deben exhibir una serie de cualidades que permitirán asegurar la calidad del requerimiento y del producto generado. Algunos atributos de calidad esperados de un requerimiento son [3], [7], [8]-[10]:

1. Completo: cada requerimiento debe describir completamente la funcionalidad esperada. Debe contener toda la información necesaria para diseñarla e implementarla.
2. Correcto: debe describir exactamente la funcionalidad que se va a construir (la mejor fuente para definir la correctitud de un requerimiento es el usuario final o un requerimiento de negocio de más alto nivel).
3. Independiente de la implementación (abstracto): el requerimiento no describe ni incluye restricciones de diseño o arquitectura al sistema, esto es, "el requerimiento expresa lo que se requiere, no como el requerimiento debe cumplirse" [1].
4. Factible: debe ser posible implementarlo dentro de las capacidades y limitaciones del sistema y su entorno operativo, financiero y legal.
5. Necesario: el requerimiento debe documentar una capacidad o funcionalidad que el cliente realmente NECESITE. En ese sentido, los requerimientos deben originarse de una fuente que efectivamente tenga la autoridad para ello y, además, debe poder trazarse (seguir) su origen hacia una regla de negocio, caso de uso o proceso que dé valor agregado al cliente.

6. Priorizado: asignar una prioridad a cada requerimiento, característica o caso de uso indica qué tan esencial es para una entrega o versión particular del producto. Si todos los requerimientos fueran tratados de la misma forma, sería muy difícil para el gestor del proyecto poder responder a cortes de presupuesto, atrasos en el cronograma, pérdidas de personal o nuevos requerimientos solicitados por el cliente.
7. No ambiguo (claro): cualquier lector del requerimiento debe llegar a la misma conclusión, entendimiento o interpretación. Por ello, los requerimientos deben escribirse en sentencias cortas y directas, usando un lenguaje apropiado al dominio del problema, típicamente apoyado por un glosario de términos.
8. Único (singular, atómico): el requerimiento debe expresar una única idea que permita ser trazada a lo largo del proyecto y los artefactos producto de él. Típicamente esto se logra identificando conjunciones (*y*, *o*, *además*, etc.) y analizando el contexto de uso de ellas para si es el caso separar el requerimiento en dos nuevos requerimientos.
9. Verificable: se busca que la estructura del requerimiento permita a un ingeniero probador diseñar casos de prueba o cualquier aproximación de verificación que permita recolectar información que demuestre que el sistema satisface efectivamente el requerimiento, así como determinar si el requerimiento está siendo correctamente implementado.

Cuando los requerimientos se agrupan, bien sea en una especificación de requerimientos o en subconjuntos para ser lanzados o usados en una iteración de desarrollo, estos deben además presentar las siguientes características [1], [2], [9]:

1. Completo: ningún requerimiento o información necesaria está ausente de la especificación. Adicionalmente, se entiende que en el conjunto de requerimientos no hay requerimientos ambiguos o por determinar (TBD: *to be determined*) por parte de los *stakeholders*.
2. Consistente: no existen conflictos entre requerimientos (se soluciona identificando las fuentes originales de los requerimientos bien sea personas o documentos) y no hay requerimientos repetidos. De manera complementaria, el lenguaje usado para expresar el conjunto de requerimientos usa siempre los mismos términos y definiciones (por ejemplo, evitar el uso de sinónimos).
3. Modificable: esto no solo implica que debe poderse reescribir un requerimiento, sino que, además, debe existir un historial de cambios hecho a los requerimientos. De ello también deriva que un requerimiento pueda ser identificado y diferenciado de otros.

4. Delimitado: el conjunto de requerimientos se mantiene dentro de un alcance específico en contexto del sistema y sus resultados esperados (esto es, no hace más de lo exigido o estipulado en el contrato para satisfacer las necesidades de los *stakeholders*). Esta característica permite controlar el crecimiento indiscriminado de las especificaciones, así como del calendario, el presupuesto y la calidad del sistema [1].
5. Factible: la especificación o el conjunto de requerimientos pueden ser satisfechos con una solución que es posible producir dentro de las restricciones dadas del proyecto y del ciclo de vida del sistema (esto es, costo, calendario, técnico, legal, operativo, regulatorio, etc.) [1].
6. Trazable: el requerimiento es trazable hacia su origen, esto es, se puede relacionar de manera única con documentos de especificación de necesidades de usuario, procesos de negocio, etc., y hacia "abajo" a otros artefactos resultantes del proceso de construcción del sistema, como son: requerimientos derivados, elementos de diseño (modelos estáticos y dinámicos, arquitectura, etc.), construcción y pruebas del sistema.
7. Estructurado y modular: la especificación está debidamente organizada y estructurada, de tal manera que los requerimientos estén, además, agrupados en subsistemas que permitan relacionar y encontrar requerimientos que deben presentarse o tratarse de manera conjunta [1].

1.3. Documentos de especificación de requerimientos

Los requerimientos son generalmente agrupados en especificaciones. Documentos que agrupan y organizan los requerimientos para que puedan ser fácilmente gestionados y actualizados por los desarrolladores del proyecto. Al igual que los requerimientos, estas especificaciones deben estar completas y ser consistentes, modificables o trazables.

Estas especificaciones son la base del trabajo posterior de desarrollo, pues sin una especificación de requerimientos no se podría:

- Diseñar el software, porque no será claro qué debe hacer.
- Validar el software, porque no será claro qué debe hacer.
- Gestionar el proyecto de software, porque cuando los requerimientos cambian (y lo hacen muy a menudo), el incremento del costo y el calendario asociado son difíciles de determinar y justificar.

En particular, el Estándar ISO/IEC/IEEE 29148 de 2011 [1] presenta una descripción general de los procesos que deben llevarse a cabo en contexto de ingeniería de sistemas y de productos de software, así como su ciclo de vida; de igual manera, presenta un conjunto de buenas prácticas para la ejecución de procesos relacionados con los requerimientos.

El Estándar 29148 también describe tres documentos asociados a dichos procesos:

1. *Documento de especificación de requerimientos de* stakeholder: describe los requerimientos de negocio y la motivación tras la construcción del sistema (en este contexto el sistema implica no solo software, sino también *hardware*, infraestructura y personal). Incluye descripciones de procesos, políticas y reglas de negocio que serán afectados por él.
2. *Documento de especificación de requerimientos de sistema*: describe el sistema, su entorno y las formas en las cuales interactuará con los usuarios y otros sistemas. Incluye elementos asociados a restricciones, requerimientos no funcionales (usabilidad, desempeño, seguridad, etc.).
3. *Documento de especificación de requerimientos de software*: es el documento de uso más común y se centra exclusivamente en la descripción del software, sus restricciones, operaciones y requerimientos. En el contexto de este texto, cuando se mencione el *Documento de especificación* o *La especificación* se hace referencia al *Documento de especificación de requerimientos de software*.

Estos documentos pueden usarse como base para la gestión y control del proyecto, así como para asegurar el cumplimiento del contrato. Los diferentes modelos presentados a lo largo de este texto pueden apoyar la definición de estos documentos.

1.4. Atributos de un requerimiento

Con el fin de construir, organizar, evaluar, cuantificar y gestionar los requerimientos, algunos autores han propuesto elementos adicionales descriptivos que pueden ser de utilidad [2], [3], [11], [12]:

1. Identificador único: este elemento permite realizar un mejor seguimiento de los requerimientos a medida que se ejecuta el proyecto.
2. Origen: identifica documentos legales, de procesos y normativos, bien sea de la organización o del gobierno que rigen la organización. En algunos casos es recomendable identificar roles de personas dentro de la organización que puedan ser entrevistados acerca de la validez de lo indicado en dichos documentos.

3. Razón de ser: descripción que permite justificar el requerimiento en contexto del objetivo del negocio, así como de los documentos que lo originan.
4. Prioridad: elemento cuantitativo o cualitativo que permite ordenar y relacionar los requerimientos de acuerdo con la importancia dada por el cliente, la complejidad técnica percibida o por riesgos de negocio o del producto relacionados con el requerimiento.
5. Dueño: en contraste con el origen, este atributo hace referencia a una persona dentro de la organización que desarrolla el producto, la cual se responsabiliza de realizar el respectivo seguimiento y gestión del requerimiento durante el desarrollo del proyecto.
6. Versión: este atributo se utiliza para llevar un historial de los cambios realizados a un requerimiento, generalmente cuando dichos cambios son producidos por el cliente, cuando posee conflictos con otros requerimientos o cuando diferentes clientes aún no están de acuerdo con él. Este atributo es útil para asegurar el cumplimiento del contrato y el historial de negociación de requerimientos altamente volátiles o conflictivos.
7. Estado: este atributo identifica, como su nombre lo indica, el estado del requerimiento y sus cambios durante las diferentes fases de desarrollo del producto, indicando si el requerimiento ha sido tenido en cuenta en las diferentes etapas de construcción del sistema, por ejemplo: diseño del sistema, diseño detallado, construcción, pruebas, entrega al cliente, etc. Este atributo es actualizado y mantenido por el dueño del requerimiento.
8. Trazabilidad: es un mecanismo que permite hacerle seguimiento al requerimiento (estado), así como identificar dónde se encuentra en los diferentes artefactos generados durante la ejecución del proyecto (véase sección "4.2.2.2. Trazabilidad").
9. Criterio de aceptación: es una descripción que indica las características que debe cumplir el sistema para indicar que efectivamente el requerimiento está siendo cumplido por él. Este elemento es de gran ayuda para los probadores del sistema e incluso para apoyar el conocimiento y entendimiento del sistema por parte del cliente y del equipo de desarrollo.
10. Conflictos: este atributo permite identificar qué conflictos tiene el requerimiento con otros requerimientos o con elementos externos, por ejemplo, reglas de negocio, procedimientos, etc.

El uso de estos atributos no es enteramente obligatorio, pero su uso en contexto de la dificultad y envergadura del proyecto puede marcar la diferencia entre el éxito y el fracaso.

1.5. Tipos de requerimientos

Para construir la especificación de requerimientos que plasma los servicios que prestará el sistema, es necesario que el analista de requerimientos defina cuáles son los que él va a utilizar en el proyecto [3]. Existen diferentes maneras de clasificar los requerimientos, pues depende del ámbito de alcance del sistema mismo. De hecho, en algunas ocasiones los grupos de proyecto crean su propia clasificación con base en las clasificaciones estándar que proponen varios autores como Ralph Young [3] o Aybüke Aurum y Claes Wohlin [11].

En el contexto de este libro se presenta una forma de clasificación de requerimientos basada en las clasificaciones dadas por autores como Karl Wiegers [2], Ian Sommerville [12] y Martin Glinz en *The Guide to the Business Analysis Body of Knowledge* [13] (figura 1.1).

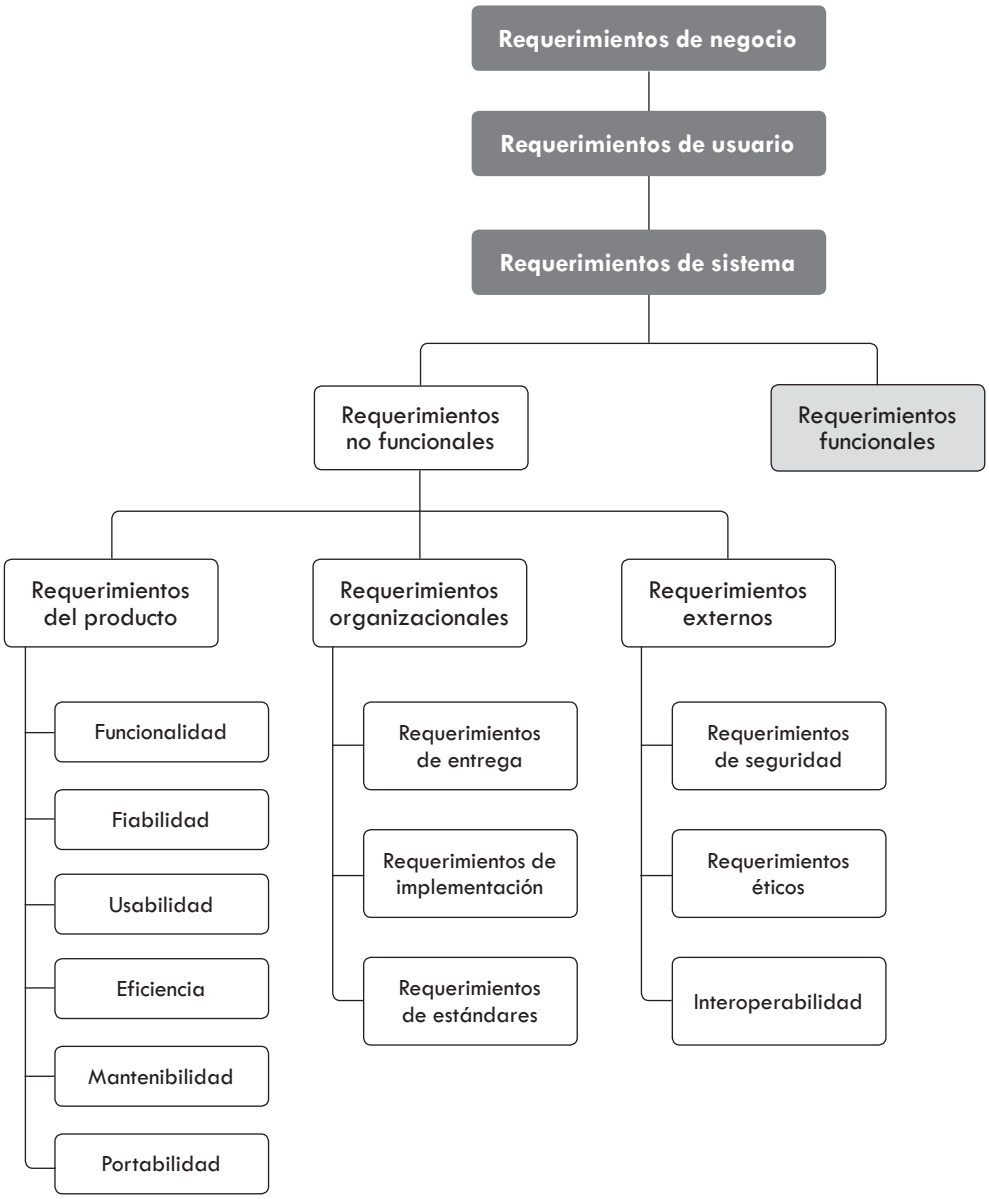
1.5.1. Requerimientos de negocio

Antes de definir los tipos de requerimientos, es necesario definir qué son las *reglas de negocio*. Estas se refieren a políticas, condiciones, restricciones, conocimientos o estándares de la industria donde se desenvuelve la compañía [17], leyes gubernamentales o aspectos del negocio, los cuales deben ser respaldados por el sistema. Las reglas de negocio permiten identificar los requerimientos de negocio [4].

De ellos se describen dos elementos muy importantes para la organización del cliente. En primer lugar, se tiene la *visión del producto* [1], la cual representa lo que es en sí mismo el producto y lo que podrá ser para la organización, especificando además el contexto de las decisiones que se deben tomar alrededor de la construcción, implantación, uso y mantenimiento del producto, lo que permite alinear los deseos y necesidades de los *stakeholders*. El otro elemento es el *alcance del proyecto,* el cual define la porción de la visión que el proyecto actual busca implementar [9].

A partir de dicho origen se definen los *requerimientos de negocio* como los requerimientos de alto nivel que describen los objetivos de la organización y cuyo cumplimiento le dará valor agregado, ya que se derivan de las reglas de negocio. Estos indican por qué se está desarrollando el proyecto. Además, permiten observar cuál es el contexto general donde se va a desarrollar el proyecto [3]. Adicionalmente, los requerimientos de negocio pueden abarcar uno o más proyectos dentro de la organización.

FIGURA 1.1. CLASIFICACIÓN DE REQUERIMIENTOS



Fuente: elaboración propia a partir de [3], [4], [13]-[16].

Según Karl Wiegers, este tipo de requerimientos "describen cómo mejoraría el mundo para ciertas comunidades si el producto estuviera disponible" [17]. Para que el concepto de requerimiento de negocio sea aún más claro, es necesario revisar el siguiente ejemplo (tomado de Lam [18]):

Se tienen las siguientes reglas de negocio:

- El conductor de un vehículo debe tener una licencia de conducir válida.
- La licencia debe tener una fecha de expiración posterior a la fecha de inspección.

De las anteriores reglas de negocio se derivan los siguientes requerimientos:

- El oficial de policía debe revisar la licencia de conducir del conductor.
- El escáner debe leer la fecha de expiración de la licencia.

1.5.2. Requerimientos de usuario

Como se nota en la figura 1.1, los requerimientos de negocio permiten identificar el entorno de la organización, sus procesos de negocio y, ante todo, los roles de las personas o entidades intervinientes, bien sea como usuarios o como beneficiarios de alguna acción del sistema. Es entonces cuando se definen los requerimientos de usuario. Una vez se han identificado los usuarios del sistema y sus beneficiarios, se pueden definir los requerimientos de usuario, que en líneas generales describen las tareas que realizarán los usuarios y que respaldará el sistema o producto en construcción. Estos requerimientos deben ser descritos de tal manera que todos los *stakeholders* del proyecto los puedan entender aun sin tener un conocimiento técnico detallado. Se debe tener en cuenta que solo deben describir el comportamiento del sistema [4].

El objetivo de recolectar los requerimientos de usuario es entender cuáles son las clases de usuarios del sistema y qué es lo que el sistema debe permitirles hacer, consultar, etc. para que se puedan apoyar y cumplir los requerimientos de negocio [17].

Para una mejor comprensión de cuáles pueden llegar a ser los requerimientos de usuario, se presentan los siguientes ejemplos basados en la construcción de un videojuego:

- El sistema debe permitir reanudar un juego que se encuentra guardado.
- El sistema debe permitir al jugador realizar movimiento de partida, únicamente cuando este se encuentre en su turno.

1.5.3. Requerimientos del sistema

Estos requerimientos son versiones ampliadas de los requerimientos de usuario, que les permiten a los ingenieros de software agregar detalles al diseño del sistema. Deben reflejar el comportamiento y las restricciones del sistema en términos de lo que hará como un todo (hardware y software) y su relación con las reglas y procesos de negocio, pero deben evitar a toda costa describir el diseño [4]. Esto se debe a que existe un punto en los proyectos donde los procesos de análisis y recolección de requerimientos se tornan caóticos y se pierde el alcance de lo que realmente el cliente quiere y necesita [14]. Por esto, es de vital importancia tener en cuenta los requerimientos de usuario, ya que son la base de los requerimientos del sistema, pues contienen la visión y las necesidades del usuario y, además, permiten acabar de definir a cabalidad el alcance del sistema.

Por lo tanto, los requerimientos del sistema permiten definir qué es lo que el sistema debe hacer; pero no solo hacen referencia a las características funcionales, las cuales se conocen como los *requerimientos funcionales* (véase sección "1.5.4. Requerimientos funcionales"), sino que también se deben definir las cualidades del sistema, las cuales se denominan *requerimientos no funcionales* (véase sección "1.5.5. Requerimientos no funcionales") [19].

Algunos ejemplos de requerimientos de sistema se aprecian a continuación:

- El sistema debe validar que el nombre de usuario y la contraseña coincidan con los datos almacenados en el sistema mismo.
- El sistema debe conocer en todo momento la cantidad de barcos que ocupan una isla.

La tabla 1.1 presenta una breve diferenciación de los requerimientos de negocio y de sistema.

1.5.4. Requerimientos funcionales

Los requerimientos funcionales son un tipo de requerimiento del sistema, y son los encargados de describir las necesidades reflejadas en los requerimientos de negocio y los requerimientos de usuario de tal manera que los desarrolladores tengan todos los detalles de cada funcionalidad, necesaria para implementar de manera correcta el sistema. Es decir, describen cuál es el comportamiento que debe tener el sistema (producto de software). Este comportamiento se expresa en forma de los servicios o funcionalidades que prestará el sistema a los usuarios [20]. También se conocen con el nombre de *requerimientos operacionales*, ya que indican cuáles son las entradas y salidas del sistema y el comportamiento de las relaciones entre ellas [3].

TABLA 1.1. COMPARACIÓN REQUERIMIENTOS DE NEGOCIO *VS.* SISTEMA

Característica	Requerimiento de negocio	Requerimiento de sistema
Lenguaje	Negocio/vista de usuario	Producto/operacional
Nivel	Conceptual	Concreto, descripción de alto nivel del producto
¿Qué?	Qué resultados de negocio deben ser generados que permitan cumplir los objetivos del negocio	Cómo debe funcionar el producto (independiente de elementos de diseño) que permita cumplir con los requerimientos de negocio esperados
¿Quién?	Ya existen dentro del negocio	Son especificados por el grupo de analistas
Posible solución	Muchas posibles soluciones	Representa una posible solución

Fuente: adaptado de Young [3].

Se recomienda que la especificación de este tipo de requerimientos sea completa [1], lo que significa que todos los servicios que el usuario quiere que preste el sistema deben estar claramente definidos y ser consistentes, lo cual indica que no debe verse ningún tipo de contradicción entre los requerimientos [4].

Algunos ejemplos de requerimientos funcionales son:

- El sistema debe permitir que únicamente un anfitrión cree una nueva partida.
- El sistema debe establecer el uso de dos dados en el momento de defenderse con más de dos barcos.

Koelsch [21] propone, además, diversos tipos de requerimientos funcionales que permiten estructurar y agruparlos, entre ellos: reglas de negocio, transacciones, funciones administrativas, autenticación, auditoría, interfaces externas, certificaciones, búsqueda y reportes, datos históricos, archivado, etc.

1.5.5. Requerimientos no funcionales

Otro tipo de requerimiento de sistema son los requerimientos no funcionales, también conocidos como *atributos de calidad*, ya que hacen referencia a las propiedades que debe cumplir el sistema al prestar sus servicios. Estos se derivan, al igual que los requerimientos funcionales, de los requerimientos de negocio y los requerimientos de usuario. Sin embargo, se diferencian de los funcionales debido a que cada uno de estos afecta a un conjunto de funciones o al sistema como un todo [11].

No obstante, no solo hacen referencia al sistema que se va a desarrollar, también pueden considerar algunas restricciones, es decir, se encargan de definir aspectos como qué estándares de calidad se deben seguir en el desarrollo del sistema o en qué tipo de hardware o entorno operacional debe funcionar el producto [4].

Este tipo de requerimientos, como se ve en la figura 1.1, también se pueden clasificar en tres diferentes categorías que son: requerimientos del producto [4] (véase sección 1.5.6), requerimientos organizacionales (véase sección 1.5.7) y requerimientos externos [3] (véase sección 1.5.8) [2].

1.5.6. Requerimientos del producto

Estos requerimientos sirven para describir cómo debe ser el comportamiento del sistema [4] cuando está realizando alguna funcionalidad [11]. Este tipo de requerimientos, como se ve en la figura 1.1, también se pueden clasificar en seis diferentes categorías. Cada una de estas representa un atributo de calidad referente al comportamiento del sistema, los cuales se encuentran definidos en el estándar ISO 25010:2011 [15].

Estos seis atributos son:

- Funcionalidad (véase sección 1.5.6.1).
- Fiabilidad (véase sección 1.5.6.2).
- Usabilidad (véase sección 1.5.6.3).
- Eficiencia (véase sección 1.5.6.4).
- Mantenibilidad (véase sección 1.5.6.5).
- Portabilidad (véase sección 1.5.6.6).

1.5.6.1. Funcionalidad

Este atributo de calidad hace referencia a los requerimientos que permiten que el sistema provea las funciones descritas con los requerimientos funcionales [22] (véase sección "1.5.4. Requerimientos funcionales"), bajo condiciones de uso especificas [16]. Las características que se deben cumplir con los atributos de funcionalidad son:

- Idoneidad (*suitability*): indica que se deben tener las funciones necesarias para cumplir las tareas requeridas.
- Precisión (*accuracy*): proveer los resultados esperados o acordados con el grado de precisión delimitado por el usuario. Un ejemplo de un requerimiento de precisión puede ser: "El sistema debe realizar los cálculos de los intereses de los créditos teniendo en cuenta dos números decimales significativos".

- Interoperabilidad (*interoperability*): la habilidad que debe tener el sistema para interactuar con uno o más sistemas, por ejemplo: "El sistema debe utilizar el motor de base de datos Oracle".
- Seguridad (*security*): es la habilidad que debe tener el sistema para evitar accesos no autorizados [22], por ejemplo: "El sistema debe cerrar automáticamente la sesión de un usuario cuando esté inactivo por un tiempo igual o superior a 15 minutos".

1.5.6.2. Fiabilidad

Los requerimientos que reflejan la fiabilidad hacen referencia a la capacidad que debe tener el sistema para mantener un nivel de rendimiento [22] bajo las condiciones como velocidad o uso de memoria, precisión [23], establecidas por el cliente en los requerimientos de negocio y lo requerimientos de usuario [16].

Las características de la fiabilidad son:

- Madurez (*maturity*): es la habilidad que tiene el producto para evitar fallos [16].
- Tolerancia a fallos (*fault tolerance*): es la habilidad que debe tener el sistema para mantener un nivel de rendimiento especificado [16], sin importar las fallas que puedan aparecer [23]; por ejemplo: "El sistema debe estar disponible el 99,99 % de las veces que uno de los usuarios lo solicite".
- Recuperable (*recoverability*): esta característica trata dos aspectos: la capacidad de recuperar los datos y la capacidad de reestablecer el rendimiento del sistema [16], después de un fallo; por ejemplo: "El sistema debe realizar un *backup* de la información cada ocho días".

1.5.6.3. Usabilidad

Los requerimientos de usabilidad deben mostrar la capacidad que debe tener el sistema de ser comprendido, aprendido y utilizado bajo las condiciones de uso estipuladas en los requerimientos del sistema [16]. Las características que se deben tener en cuenta al especificar los requerimientos de usabilidad son:

- Comprensibilidad (*understandability*): es la capacidad que presenta el sistema para que el usuario lo pueda entender; por ejemplo: "El producto deberá ser fácil de usar por los miembros de la organización que no leen en idioma inglés".
- Aprendizaje (*learnability*): "la capacidad que tiene el sistema para permitir que el usuario lo aprenda a utilizar" [16]; por ejemplo: "El sistema debe permitir que usuarios expertos aprendan a utilizarlo en un tiempo máximo de tres horas".

- Operabilidad (*operability*): "la capacidad que tiene el sistema para permitir que el usuario lo opere y lo controle" [16]; por ejemplo: "El producto permitirá interacción con usuarios invidentes".

1.5.6.4. Eficiencia

Los requerimientos de eficiencia deben especificar cuál es el rendimiento esperado, teniendo en cuenta la cantidad de recursos y las condiciones establecidas por los *stakeholders* [16]. Las características para definir los requerimientos no funcionales de eficiencia son:

- Tiempo de respuesta: es la capacidad que tiene el sistema de responder de manera correcta ante alguna solicitud de algún usuario; por ejemplo: "El sistema debe responder a una petición en un tiempo promedio de 7 segundos".
- Utilización de recursos: son los recursos que utiliza el sistema en el momento de realizar alguna funcionalidad [22]; por ejemplo: "El sistema debe usar como máximo 512 Mbps para la gestión de transacciones".

1.5.6.5. Mantenibilidad

La mantenibilidad debe reflejar la capacidad que debe tener el sistema de ser modificado, ya sea para actualizar el sistema, incluir correcciones o cambiarlo para adaptarlo a algún cambio del entorno o de los requerimientos funcionales por parte de un equipo capacitado de desarrollo [23]. Dentro de las características de la mantenibilidad se deben tener en cuenta:

- Cambios (*changeability*): capacidad que debe tener el sistema para especificar e implementar una modificación [16], por ejemplo: "El código fuente de la aplicación debe ser documentado con ayuda de la herramienta JavaDoc. Asegurando la correcta compresión, orden y sintaxis de este".
- Estabilidad (*stability*): capacidad que tiene el sistema para evadir los efectos inesperados debidos a las modificaciones realizadas sobre el sistema [16].
- Pruebas (*testability*): capacidad que debe tener el sistema para ser verificado mediante pruebas [16].

1.5.6.6. Portabilidad

La portabilidad refleja la capacidad del sistema de cambiar de un ambiente a otro. El ambiente puede ser organizacional, de hardware o de software [22]. Por esto, los requerimientos no funcionales de portabilidad deben especificar las siguientes características:

- Adaptación (*adaptability*): los requerimientos que quieran reflejar la adaptación deben especificar cómo se debe adaptar el sistema a un nuevo ambiente; por ejemplo: "El sistema debe permitir la instalación por parte del usuario sin ningún tipo de entrenamiento".
- Coexistencia: capacidad que debe tener el sistema para convivir con otros sistemas que existan en el ambiente [16]; por ejemplo: "El sistema debe poder ejecutarse bajo la plataforma de Windows".

1.5.7. Requerimientos organizacionales

Este tipo de requerimientos "son la consecuencia de políticas y procedimientos organizacionales" [4] que existen tanto en la organización del cliente como en la de los desarrolladores del sistema y que típicamente restringen el proceso, las herramientas o el desarrollo mismo del producto. Estos requerimientos también se pueden dividir en las siguientes tres categorías [4], [12]:

- Requerimientos de entrega: deben describir las fechas límites para realizar la entrega del sistema y la documentación de este.
- Requerimientos de implementación: deben especificar cuál es el lenguaje de programación que se va a utilizar o cuál es el método de diseño que es mandatorio utilizar.
- Requerimientos provenientes de las normas: describen cuáles son los estándares de calidad, estándares de documentación y estándares en los procesos de desarrollo.

Un ejemplo de un requerimiento organizacional es: "El proceso de desarrollo del sistema y los documentos que se van a entregar deberán ajustarse al proceso y a los productos definidos en la organización" [4].

1.5.8. Requerimientos externos

Como su nombre lo indica, son los requerimientos "que se derivan de factores que son externos al sistema y su proceso de desarrollo" [4]. Se pueden clasificar en tres ítems que son [4], [12]:

- Requerimientos éticos: se deben especificar, para asegurar que los usuarios aceptarán el sistema [24]. También, corresponden a comportamientos que se esperan del sistema —éticamente—, por ejemplo, el uso correcto de la información personal de los usuarios.

- Requerimientos de interoperabilidad: definen cómo el sistema debe interactuar con otros sistemas de otras organizaciones.
- Requerimientos de seguridad (*safety*): especificarán cuál es la protección que debe tener el sistema en caso de modificaciones o destrucción. Además, debe asegurar la integridad física de los usuarios que interactúan con el sistema en caso que el sistema controle algún dispositivo de hardware (por ejemplo, un torno automatizado) [23].

Ian Sommerville da un ejemplo de un requerimiento externo: "El sistema no deberá revelar al personal de la biblioteca que lo utilice ninguna información personal de los usuarios del sistema aparte de su nombre y número de referencia" [4].

Después de que los equipos de desarrollo deciden cuál es la clasificación de requerimientos que usarán durante el desarrollo del proyecto, se deben empezar a definir roles, procesos, actividades y herramientas para aplicar en general en un proceso de ingeniería de requerimientos (véanse los capítulos 2 y 3).

Referencias

[1] "ISO/IEC/IEEE International Standard - Systems and software engineering - Life cycle processes - Requirements engineering", *ISOIECIEEE 291482011E*, pp. 1-94, dic. 2011.

[2] K. Wiegers y J. Beatty, *Software Requirements*, 3.ª ed. Redmond, WA: Microsoft Press, 2013.

[3] R. R. Young, *The Requirements Engineering Handbook*. Boston: Artech House Print on Demand, 2003.

[4] I. Sommerville, *Software Engineering*, 9.ª ed. Boston: Pearson, 2010.

[5] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell y A. Wesslén, *Experimentation in Software Engineering*. Berlín, Heidelberg: Springer Berlin Heidelberg, 2012.

[6] Real Academia Española. *Diccionario de la lengua española*. [Online]. Disponible: http://dle.rae.es/?w=diccionario. Consultado en: 29 de marzo de 2017.

[7] I. Alexander y R. Stevens, *Writing Better Requirements*, 1.ª ed. Londres: Addison-Wesley Professional, 2002.

[8] K. E. Wiegers, "Writing quality requirements", *Softw. Dev.*, vol. 7, no. 5, pp. 44-48, 1999.

[9] E. Hull, K. Jackson y J. Dick, *Requirements Engineering*, 3.ª ed. Londres: Springer, 2010.

[10] P. Bourque y R. Fairley, eds., *Guide to the Software Engineering Body of Knowledge*. IEEE Computer Society, 2014.

[11] A. Aurum, *Engineering and Managing Software Requirements*. Londres: Springer.

[12] I. Sommerville y P. Sawyer, *Requirements Engineering: A Good Practice Guide*. Haboken, NJ: John Wiley & Sons, 1997.

[13] International Institute of Business Analysis. *BABOK® Guide*. [Online]. Disponible: https://www.iiba.org/babok-guide.aspx. Consultado en: 20 de febrero de 2017.

[14] R. R. Young, *Project Requirements: A Guide to Best Practices*, 1.ª ed. Vienna, Va: Management Concepts, 2006.

[15] International Organization for Standardization. "ISO/IEC 25010:2011: Systems and Software Engineering -- Systems and Software Quality Requirements and Evaluation (SQuaRE) -- System and Software Quality Models". [Online]. Disponible: https://www.iso.org/standard/35733.html. Consultado en: 4 de abril de 2017.

[16] F. Losavio, L. Chirinos, N. Lévy y A. Ramdane-Cherif, "Quality characteristics for software architecture", *J. Object Technol.*, vol. 2, no. 2, pp. 133-150, 2003.

[17] K. Wiegers, *More about Software Requirements: Thorny Issues and Practical Advice*, 1.ª ed. Redmond, WA: Microsoft Press, 2005.

[18] G. Lam, "Business rules vs. business requirements". *Business Rules Community*. [Online]. Disponible: http://www.brcommunity.com/articles.php?id=b290. Consultado en: 4 de abril de 2017.

[19] I. Alexander, "Being clear: Requirements are either needs or specifications", *Requirenautics Q. Newsl. Requir. Eng. Spec. Group Br. Comput. Soc.*, vol. 25, pp. 10-12, 2002.

[20] R. Malan y D. Bredemeyer, "Functional Requirements and Use Cases". [Online], 2001. Disponible: http://www.bredemeyer.com/pdf_files/functreq.pdf

[21] G. Koelsch, *Requirements Writing for System Engineering.* Nueva York: Apress, 2016.

[22] I. Padayachee, P. Kotze y A. van Der Merwe, "ISO 9126 external systems quality characteristics, sub-characteristics and domain specific criteria for evaluating e-Learning systems". [Online], 2015. Disponible: https://www.researchgate.net/publication/228987388_ISO_9126_external_systems_quality_characteristics_sub-characteristics_and_domain_specific_criteria_for_evaluating_e-Learning_systems

[23] M. Barbacci, M. H. Klein, T. A. Longstaff y C. B. Weinstock, "Quality attributes". [Online] Pittsburgh: Carnegie Mellon University, 1995. Disponible: https://www.google.com.co/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0ahUKEwilx72P3eHVAhWCwiYKHaTpAoIQFggmMAA&url=http%3A%2F%2Fwww.dtic.mil%2Fget-tr-doc%2Fpdf%3FAD%3DADA307888&usg=AFQjCNGwxo2lHgEp9EXHduSdGKZEh6OAPQ

[24] K. Wiegers, "First things first", *Softw. Dev.*, vol. 7, no. 9, pp. 48-53, set. 1999.

La ingeniería de requerimientos es, al igual que la disciplina que la agrupa (la ingeniería de software), una ciencia bastante joven. Solo hasta la década de 1980, se comenzó a profundizar en el papel de los requerimientos para comprender, abstraer y modelar problemas y, por supuesto, para usarlos como herramienta de gestión y control de proyectos. Para los ingenieros de software, esto significó acercase más al negocio de sus clientes, para dar una respuesta más completa a sus necesidades.

Ingeniería de requerimientos tiene como objeto ayudar a ingenieros industriales y de procesos a comprender de mejor manera cómo expresar las necesidades de un proyecto para buscar una solución con tecnologías de la información. Aun cuando no hay una secuencia de pasos totalmente cierta y válida para todos los casos, es importante conocer diferentes estrategias y los aportes que ellas pueden brindar al contexto del proyecto. Por eso, el logro de este trabajo es mostrar diferentes métodos y estrategias para la identificación, la especificación, el análisis y la gestión aplicable tanto a los requerimientos de negocio como de software.

Pontificia Universidad
JAVERIANA
Bogotá

editorial
Pontificia Universidad
JAVERIANA
25 AÑOS