



Modalidad Abierta y a Distancia

Ingeniería de Requisitos

Guía didáctica



Facultad de Ingenierías y Arquitectura

Departamento de Ciencias de la Computación y Electrónica

Ingeniería de Requisitos

Guía didáctica

Carrera	PAO Nivel
▪ <i>Tecnologías de la información</i>	VI

Autor:

Manuel Eduardo Sucunuta España



D S O F _ 3 0 4 6

Asesoría virtual
www.utpl.edu.ec

Universidad Técnica Particular de Loja

Ingeniería de Requisitos

Guía didáctica

Manuel Eduardo Sucunuta España

Diagramación y diseño digital:

Ediloja Cía. Ltda.

Telefax: 593-7-2611418.

San Cayetano Alto s/n.

www.ediloja.com.ec

edilojacialtda@ediloja.com.ec

Loja-Ecuador

ISBN digital - 978-9942-39-771-3



**Reconocimiento-NoComercial-CompartirIgual
4.0 Internacional (CC BY-NC-SA 4.0)**

Usted acepta y acuerda estar obligado por los términos y condiciones de esta Licencia, por lo que, si existe el incumplimiento de algunas de estas condiciones, no se autoriza el uso de ningún contenido.

Los contenidos de este trabajo están sujetos a una licencia internacional Creative Commons **Reconocimiento-NoComercial-CompartirIgual 4.0 (CC BY-NC-SA 4.0)**. Usted es libre de **Compartir – copiar y redistribuir el material en cualquier medio o formato. Adaptar – remezclar, transformar y construir a partir del material citando la fuente, bajo los siguientes términos: Reconocimiento- debe dar crédito de manera adecuada, brindar un enlace a la licencia, e indicar si se han realizado cambios.** Puede hacerlo en cualquier forma razonable, pero no de forma tal que sugiera que usted o su uso tienen el apoyo de la licenciatante. **No Comercial-no puede hacer uso del material con propósitos comerciales. Compartir igual-Si remezcla, transforma o crea a partir del material, debe distribuir su contribución bajo la misma licencia del original.** No puede aplicar términos legales ni medidas tecnológicas que restrinjan legalmente a otras a hacer cualquier uso permitido por la licencia. <https://creativecommons.org/licenses/by-nc-sa/4.0/>

Índice

1. Datos de información.....	8
1.1. Presentación de la asignatura	8
1.2. Competencias genéricas de la UTPL	8
1.3. Competencias específicas de la carrera.....	8
1.4. Problemática que aborda la asignatura.....	8
2. Metodología de aprendizaje.....	9
3. Orientaciones didácticas por resultados de aprendizaje	10
Primer bimestre.....	10
Resultado de aprendizaje 1.....	10
Contenidos, recursos y actividades de aprendizaje	10
 Semana 1	10
 Unidad 1. Introducción a la Ingeniería de Requisitos	10
1.1. ¿Qué es la Ingeniería de Requisitos?	10
1.2. Actividades de la Ingeniería de Requisitos	24
Actividad de aprendizaje recomendada	27
 Semana 2	27
1.3. Buenas prácticas de la Ingeniería de Requisitos.....	27
1.4. El analista de negocio	28
Actividades de aprendizaje recomendadas	40
Autoevaluación 1.....	41
 Semana 3	43
 Unidad 2. Definiendo los requisitos de negocio	43
2.1. Interesados	43
2.2. Análisis de interesados.....	51
 Semana 4	56
2.3. Modelado de negocio	56
2.4. Requisitos de negocio.....	57
2.5. Documento de visión y alcance.....	65

2.6. Técnicas para representar el alcance	74
2.7. Mantener el alcance en foco	80
2.8. Visión y alcance en proyectos ágiles	82
2.9. Uso de los objetivos de negocio para determinar la finalización	84
Actividades de aprendizaje recomendadas	85
Autoevaluación 2.....	87
Resultado de aprendizaje 2.....	89
Contenidos, recursos y actividades de aprendizaje	89
Semana 5	89
Unidad 3. Obtención de requisitos	89
3.1. Fuente de los requisitos.....	89
3.2. Obtención.....	90
3.3. Técnicas para obtener requisitos	93
Semana 6	104
3.4. Planificar la obtención	104
Semana 7	106
3.5. Preparación para la obtención.....	106
3.6. Realizar las actividades de obtención	109
3.7. Seguimiento luego de la obtención.....	110
Actividades de aprendizaje recomendadas	111
Autoevaluación 3.....	113
Semana 8	115
Actividades finales del bimestre	115
Actividades de aprendizaje recomendadas	116
Segundo bimestre	117
Resultado de aprendizaje 3.....	117
Contenidos, recursos y actividades de aprendizaje	117

Semana 9	117
 Unidad 4. Análisis de requisitos	117
4.1. El análisis de requisitos	117
4.2. Modelado de requisitos	119
Semana 10	127
4.3. Los diagramas UML	127
4.4. La interfaz de usuario y los requisitos	129
Semana 11	131
4.5. Modelo de casos de uso.....	131
4.6. Modelo ágil	143
4.7. Reglas de negocio.....	146
Actividades de aprendizaje recomendadas	151
Autoevaluación 4.....	152
 Resultado de aprendizaje 4.....	154
Contenidos, recursos y actividades de aprendizaje	154
Semana 12	154
 Unidad 5. Documentando los requisitos.....	154
5.1. ¿Por qué documentar?	154
5.2. Documento de Especificación de Requisitos de Software.....	158
Semana 13	160
5.3. Etiquetado de requisitos	160
5.4. Características de la declaración de requisitos	163
5.5. Lineamientos para escribir requisitos.....	167
Actividades de aprendizaje recomendadas	174
Autoevaluación 5.....	175
Semana 14	177
 Unidad 6. Validando los requisitos	177
6.1. Verificación y validación	177
6.2. Revisión de los requisitos	179
6.3. Prototipos	191

6.4. Probando los requisitos	193
6.5. Validación de requisitos con criterios de aceptación	195
Actividades de aprendizaje recomendadas	200
Autoevaluación 6.....	201
Resultado de aprendizaje 5.....	203
Contenidos, recursos y actividades de aprendizaje	203
Semana 15	203
 Unidad 7. Gestión de requisitos	203
7.1. Proceso de gestión de requisitos.....	203
7.2. La línea base de los requisitos	206
7.3. Control de versiones de requisitos.....	207
7.4. Atributos de requisito.....	209
7.5. Seguimiento al estado de los requisitos.....	210
Actividades de aprendizaje recomendadas	213
Autoevaluación 7.....	214
 Semana 16	216
Actividades finales del bimestre	216
Actividad de aprendizaje recomendada	216
4. Solucionario	217
5. Glosario.....	225
6. Referencias bibliográficas	226
7. Anexos	228



1. Datos de información

1.1. Presentación de la asignatura



1.2. Competencias genéricas de la UTPL

Organización y planificación del tiempo.

1.3. Competencias específicas de la carrera

Modelar procesos de negocio utilizando técnicas y marcos de referencia para identificar problemas, oportunidades de mejora y proponer alternativas que permitan dar soporte a la estrategia del negocio.

1.4. Problemática que aborda la asignatura

La cantidad de proyectos de desarrollo de software exitoso es relativamente baja, es decir, que solo una pequeña parte de los proyectos que empezaron a desarrollarse culminaron en el tiempo previsto, con el presupuesto asignado y con los requisitos establecidos. Una de las causas para que los proyectos

de desarrollo de software fracasen, se debe a la incorrecta especificación de requisitos, por tanto, esta asignatura aborda los temas relacionados con el proceso de desarrollo y gestión de requisitos para lograr a través de modelos, estrategias, técnicas y herramientas definir requisitos con los niveles de calidad que exigen los procesos de desarrollo actuales.



2. Metodología de aprendizaje

Para que el estudiante adquiera las competencias de aprendizaje que se han planificado en la presente asignatura, se utilizará el aprendizaje basado en el análisis sistemático de un caso de estudio, con el objetivo que el estudiante analice situaciones aproximadas a la realidad sobre los temas que se abordarán.

Para lograr el aprendizaje se precisa de una participación del estudiante a la hora de decidir qué, cómo y cuándo debe estudiarse algo. Se espera que el estudiante analice los ejemplos que se indican en la guía didáctica, que le permitan descubrir los principios, conceptos y estrategias que debe estudiar. Este tipo de enseñanza-aprendizaje fomenta la curiosidad y el desarrollo de destrezas que permiten el aprendizaje a lo largo de toda la vida, además de permitir que el estudiante se sienta parte activa de este proceso. En este sentido, y considerando los resultados de aprendizaje, se considera lo siguiente:

- Autoaprendizaje a través del estudio de los conceptos relacionados con los temas que se han planificado en la asignatura.
- Permite al estudiante desarrollar habilidades tanto para resolver problemas, como para tomar decisiones.
- Desarrollo de las actividades recomendadas en los temas que se han especificado.
- Análisis de los ejemplos para sustentar los conceptos teóricos.
- Revisar, analizar y comprender el caso desarrollado y que se comenta en el desarrollo de los contenidos.

Esta metodología es altamente formativo, pero requiere de una gran disciplina tanto del estudiante como del formador (tutor).



3. Orientaciones didácticas por resultados de aprendizaje



Primer bimestre

Resultado de aprendizaje 1

- Identifica a los Interesados del sistema y define sus necesidades.

Este resultado de aprendizaje le permitirá conocer los fundamentos de la Ingeniería de Requisitos para inicialmente comprender la importancia de la participación de los interesados en el desarrollo de un proyecto de desarrollo de software. Adicionalmente, le permitirá conocer la manera en que se debe realizar el modelado de negocio, para definir el visionamiento y alcance del proyecto a desarrollar.

Contenidos, recursos y actividades de aprendizaje



Semana 1

Unidad 1. Introducción a la Ingeniería de Requisitos

Empezamos el estudio de la asignatura comprendiendo los conceptos necesarios de la Ingeniería de Requisitos, las actividades que se desarrollan para especificar los requisitos, las buenas prácticas y sobre el rol del analista de negocio al momento de desarrollar cada una de las actividades de desarrollo y gestión de los requisitos.

1.1. ¿Qué es la Ingeniería de Requisitos?

Para empezar con el estudio de Ingeniería de Requisitos (IR), es necesario conocer la definición del término “requisito”. De acuerdo con la Real

Academia Española, el término requisito se define como una “Circunstancia o condición necesaria para algo”, evidentemente que desde el punto de vista del software los requisitos serán los elementos clave que permitan el desarrollo del software bajo ciertas “restricciones” que deberán ser interpretadas de acuerdo con lo que un cliente necesita. A continuación, se destacan algunos conceptos relevantes en el contexto de la IR.

1.1.1. Ingeniería de software

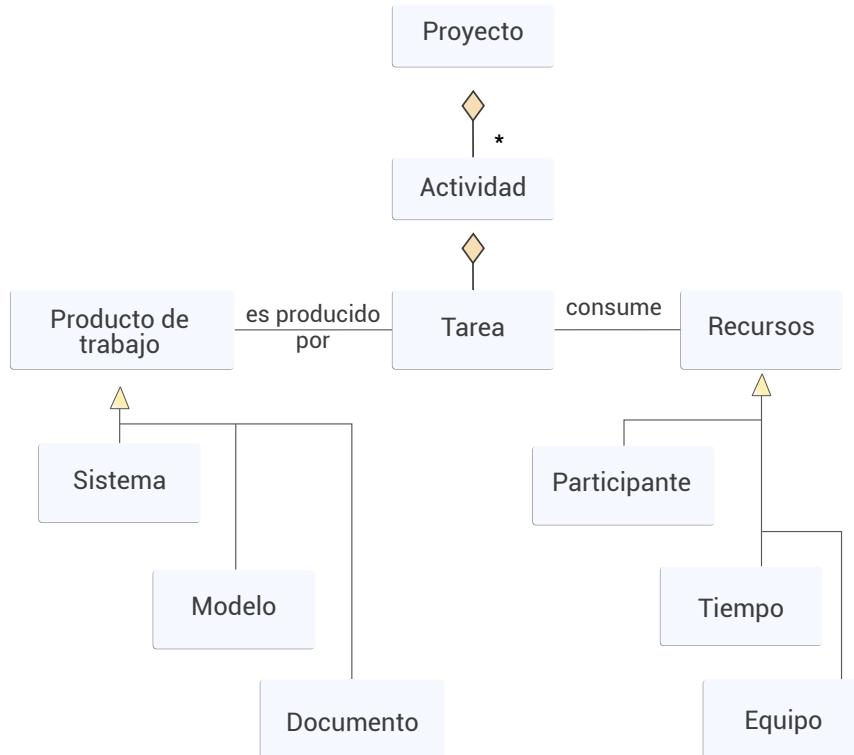
La IEEE define a la Ingeniería de Software (IS) como 1. La Aplicación de un enfoque sistemático, disciplinado y cuantificable para el desarrollo, operación y mantenimiento del software, esto es, la aplicación de ingeniería al software. 2. El estudio de enfoques como en 1”(IEEE, 1990). Como se puede apreciar, la Ingeniería de Software incorpora el estudio y aplicación de la ingeniería a las distintas actividades que se requiere para el desarrollo del software. Los Ingenieros de Software trabajan en estas actividades para:

- Realizar el modelado.
- Dar solución de problemas.
- La adquisición de conocimiento.
- Dirigida por una fundamentación.

Estas actividades requieren de cierto dominio y conocimiento de conceptos relacionados con la IS, y que directamente afectan a las actividades de la IR, ya sea en las actividades de desarrollo como las actividades de gestión. En la figura 1, se muestra la relación de los principales conceptos que definen a la IS (Bruegge & Dutoit, 2009).

Figura 1

Conceptos de Ingeniería de Software



Nota. Adaptado de *Object oriented software engineering using uml, patterns, and java* (p. 11), por Bruegge, B. y Dutoit, A., 2009, Pearson Education.

El diagrama de la figura 1, se realiza utilizando la notación del Lenguaje de Modelado Unificado (UML, por sus siglas en inglés). El diagrama describe los conceptos y la relación que están asociados a la IS. Se puede identificar un **proyecto**, cuyo objetivo es desarrollar un sistema software, y que, por lo tanto, está compuesto por varias **actividades**. Cada actividad a su vez está compuesta por varias **tareas**. Una tarea consume recursos y origina un **producto de trabajo**. Un producto de trabajo puede ser un **sistema**, un **modelo** o un **documento**. Los Recursos que se requiere para el desarrollo son **participantes**, **tiempo** o **equipo**.

Existe diferentes procesos para el desarrollo de software, pero todos deben incluir cuatro actividades que son fundamentales para la ingeniería de software, estas son:

1. **Especificación del software**: se define tanto la funcionalidad del software como las restricciones de su operación.
2. **Diseño e implementación del software**: se desarrolla el software para cumplir con las especificaciones.
3. **Validación del software**: validar el software para asegurarse de que cumple lo que el cliente requiere.
4. **Evolución del software**: el software tiene que evolucionar para satisfacer las necesidades cambiantes del cliente.

En los siguientes apartados nos dedicaremos al estudio de esta primera parte “Especificación del software”, desde la perspectiva de la Ingeniería de Software.

1.1.2. Requisitos de software

El concepto de requisito ha sido debatido por los profesionales del software durante mucho tiempo, ya que es un concepto fundamental para el desarrollo y gestión de proyectos de software. Cuando se habla de requisitos a menudo se dan problemas con su terminología, por ejemplo: requisitos de usuario, requisitos del sistema, requisitos del producto, requisitos de negocios, etc., lo que genera cierta confusión. A continuación se indican algunos conceptos relacionados con los Requisitos de Software.

“Los requisitos son una especificación de lo que podría ser implementado. Son descripciones de: cómo el sistema podría comportarse o de las propiedades y atributos del sistema” (Sommerly & Sawyer, 1997). En esta definición claramente se evidencia que al momento de definir los requisitos, estos se orientan a lo que será el software, las características que tendrá y que permitirá a los usuarios realizar de forma apropiada las actividades.

“Un requisito del software es una característica que debe exhibir para solucionar cierto problema del mundo real. Por lo tanto, un requisito del software es una característica que debe exhibir el software desarrollado o adaptado para solucionar un problema particular” (Bourque & Fairley, 2014). En esta definición se realza la importancia del requisito para solucionar problemas a los que afronta una organización mediante las características que debe poseer el sistema cuando sea implementado. También se

establece que estas características se pueden definir a partir de *software* ya desarrollado, pero que al momento está dando problemas y que debe ser actualizado.

"Los requisitos son descripciones de las necesidades y propiedades de un producto que satisface las necesidades del cliente" (Gottesdiener, 2005). En esta definición se establece la importancia de la participación del usuario, al momento de definir los requisitos y la forma en que estos aportan al *software* (producto).

La IEEE define a un requisito como:

1. Condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo.
2. Condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente.
3. Una representación en forma de documento de una condición o capacidad como las expresadas en (1) o en (2). (IEEE, 1990). Esta definición acapara algunos aspectos importantes en el desarrollo de los requisitos, y que precisamente son los que estudiaremos en los siguientes apartados.

Con base en las distintas definiciones que realizan los autores, se puede decir que los requisitos son declaraciones sobre lo que debe hacer un sistema, de cómo debe comportarse, de las propiedades que debe tener, de las características que debe poseer y las restricciones que debe cumplir el sistema y su desarrollo.

La Ingeniería de Requisitos cumple una importante tarea en el proceso de desarrollo de *software*, por lo tanto, para analizar detenidamente cada una de las actividades es necesario comprender los conceptos que se indican en la tabla 1.

Tabla 1*Conceptos relacionados con los requisitos*

Término	Definición
Requisito de negocio	Representan los objetivos de alto nivel de la organización o del cliente que lo requiere para desarrollar un producto <i>software</i> .
Regla de negocio	Son una política, directriz, estándar o regulación que define o restringe algún aspecto del negocio. No es un requisito de <i>software</i> en sí, sino el origen de varios tipos de requisitos de <i>software</i> .
Restricción	Es imposición a las posibilidades disponibles para el desarrollo, diseño y construcción de un producto.
Requisito de interfaz externa	Es una descripción de una conexión entre el sistema de <i>software</i> y un usuario, otro sistema de <i>software</i> o un dispositivo de <i>hardware</i> .
Característica	Capacidades del sistema relacionadas lógicamente que proporcionan valor al usuario y se describen mediante un conjunto de requisitos funcionales.
Requisito funcional	Una descripción de un comportamiento que exhibirá un sistema bajo condiciones específicas.
Requisito no funcional	Una descripción de una propiedad o característica que un sistema debe exhibir o una restricción que debe respetar.
Atributo de calidad	Es un tipo de requisito no funcional que describe un servicio o una característica de desempeño de un producto.
Requisito de sistema	Es un requisito de alto nivel para un producto que contiene varios subsistemas, que podría ser todo el <i>software</i> o <i>software y hardware</i> .
Requisito de usuario	Es una meta o tarea específica de usuarios que podría ejecutar con un sistema o un atributo de un producto deseado.

Nota: Sucunuta, M., 2023

Estos conceptos son sobre los que se definen las actividades de la IR, por lo tanto, es importante su interpretación y análisis, especialmente para determinar el resultado de cada actividad. Existen otros conceptos, pero en el desarrollo de la asignatura se abordarán.

1.1.3. Ingeniería de Requisitos

La IR es una rama de la Ingeniería de *Software* que se encarga del desarrollo de actividades relacionadas con la comprensión de las necesidades del usuario y definir los servicios que se espera del sistema. Varios autores han propuesto definiciones enfocadas a criterios que consideran relevantes. A continuación se indican los más relevantes.

- Según (Sommerville & Sawyer, 1997), es un “proceso sistemático de desarrollo de requisitos mediante un proceso iterativo y cooperativo de analizar el problema, documentar las observaciones resultantes

en varios formatos de representación y comprobar la precisión del conocimiento obtenido”.

- Según Gottesdiener (2005), es una “disciplina dentro de la ingeniería de sistemas y software que abarca todas las actividades y entregables asociados con la definición de los requisitos de un producto –es una de las mejores formas para desarrollar excelentes requisitos–. La ingeniería de Requisitos está compuesta por el desarrollo y gestión de requisitos”.
- Según Durán (2000), es “un proceso de descubrimiento y comunicación de las necesidades de clientes y usuarios y la gestión de los cambios en dichas necesidades”.
- Según Christel y Kang (1992), es “el proceso sistemático de desarrollar requisitos mediante un proceso iterativo y cooperativo de analizar el problema, documentar las observaciones resultantes en varios formatos de representación y comprobar la precisión del conocimiento obtenido”.
- Según Ian y Ljerka (2009), “todas las actividades relacionadas con: a) identificación y documentación de las necesidades de clientes y usuarios; b) creación de un documento que describe la conducta externa y las restricciones asociadas de un sistema que satisfará dichas necesidades; c) análisis y validación del documento de requisitos para asegurar consistencia, compleción y viabilidad; d) evolución de las necesidades”.
- Según Pressman (2010), es un “conjunto de procesos, tareas y técnicas que permiten la definición y gestión de los requisitos de un producto de un modo sistemático. En definitiva, facilita los mecanismos adecuados para comprender las necesidades del cliente, analiza sus necesidades, confirma su viabilidad, negocia una solución razonable, especifica la solución sin ambigüedad, valida la especificación y gestiona los requisitos para que se transformen en un sistema operacional”.

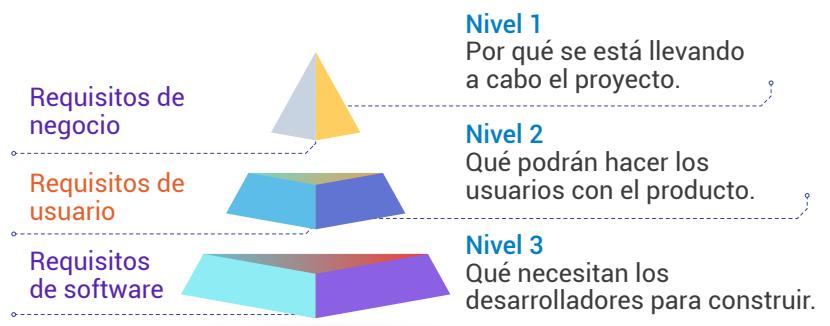
Como se puede apreciar, existen diversas definiciones acerca de la IR, pero todas las definiciones se enfocan a identificar las necesidades del cliente y mediante un proceso sistemático determinar cuáles serían las

características que deberá tener un sistema software. Esto nos lleva a considerar dos instancias:

- El “system-as-is”. Se determina el sistema cómo es, la situación actual. Se conoce como “sistema actual”.
- El “system-to-be”. Se especifica el sistema a realizar, cuándo se construirá y operará. Se denomina el sistema futuro.

Las actividades de la IR permiten establecer tres niveles de requisitos de software: requisitos de negocio, requisitos de usuario y requisitos funcionales, tal como se puede apreciar en la figura 2.

Figura 2
Niveles de los requisitos

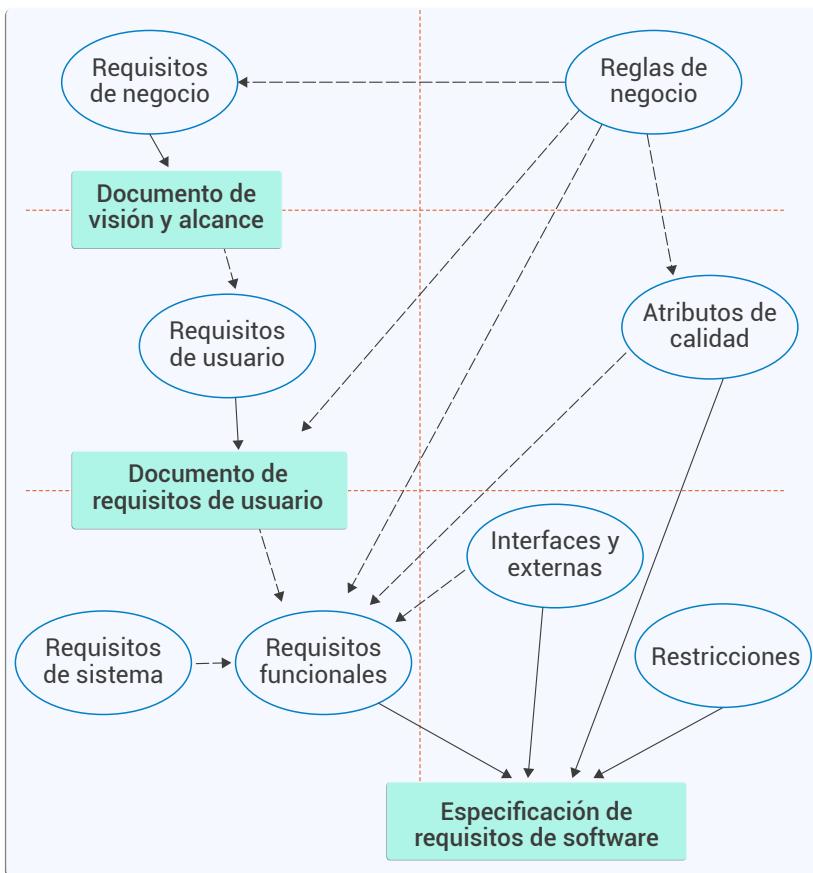


Nota. Adaptado de *The Software Requirements Memory Jogger* (p. 9), por Gottesdiener, E., 2005, Goal/Opc.

Estos niveles de requisitos se definen con base en la información que se obtenga y analice, mediante las estrategias de obtención y análisis, complementado con documentación, ya sea a través de diagramas o de forma textual.

Figura 3

Relación de los diferentes niveles de requisitos



Nota: Adaptado de *Software Requirements* (p. 8), por Wiegers, K. y Beatty, J., 2013. Microsoft.

En la figura 3, se muestran los tres niveles de requisitos con sus respectivos elementos que se generan y aquellos que aportan para su definición. Las flechas continuas significan “están almacenados en”; las flechas punteadas significan “son el origen de” o “influencia”, en este sentido se puede identificar que el resultado de los “requisitos de negocio”, se plasman en el “documento de visión y alcance”, de igual manera, las “reglas de negocio” aporta para esta definición. El objetivo principal es la “Especificación de requisitos de software”, para lo cual existen diferentes elementos

como es los “requisitos de sistema”, “requisitos funcionales”, “interfaces externas” y las “restricciones”, cuyo resultado se refleja en este documento (contenedor).

1.1.4. Requisitos de negocio

Los requisitos de negocio describen por qué la organización está implementando el sistema, los beneficios comerciales que la organización espera lograr. El interés está en los objetivos de negocio de la organización o del cliente que solicita el sistema. Suponga que una aerolínea quiere reducir los costos de personal de mostrador del aeropuerto en un 25 por ciento. Este objetivo podría llevar a la idea de construir un quiosco para que los pasajeros puedan usar para registrarse para sus vuelos en el aeropuerto. Los requisitos de negocio generalmente provienen del patrocinador que financia el proyecto, del cliente adquirente, del gerente, de los usuarios reales, del departamento de *marketing* o de un visionario del producto. Los requisitos de negocio se registran en el documento de visión y alcance. Otros documentos de orientación estratégica que a veces se usan para este propósito incluyen un acta de constitución del proyecto, un caso de negocios y un documento de requisitos de mercado (o *marketing*). En los siguientes apartados especificaremos con mayor detalle estos requisitos.

1.1.5. Requisitos de usuario

Describen objetivos o tareas que los usuarios deben poder realizar con el producto que proporcionará valor a alguien. El dominio de los requisitos del usuario también incluye descripciones de atributos o características del producto que son importantes para la satisfacción del usuario. Las formas de representar los requisitos de los usuarios incluyen casos de uso, historias de usuarios y tablas de respuesta a eventos. Idealmente, los representantes de los usuarios proporcionarán esta información. Los requisitos del usuario describen lo que el usuario podrá hacer con el sistema. Un ejemplo de un caso de uso es “registrarse para un vuelo” utilizando el sitio web de una aerolínea o un quiosco en el aeropuerto. Escrito como una historia de usuario, el mismo requisito de usuario podría decir: “Como pasajero, quiero registrarme para un vuelo para poder abordar mi avión”. Es importante recordar que la mayoría de los proyectos tienen múltiples clases de usuarios, así como otros interesados cuyas necesidades también se deben obtener.

1.1.6. Los requisitos funcionales

Describen lo que los desarrolladores deben implementar para permitir que los usuarios realicen sus tareas (requisitos del usuario), satisfaciendo así los requisitos de negocio. Esta alineación entre los tres niveles de requisitos es esencial para el éxito del proyecto. Los requisitos funcionales a menudo se escriben de la forma tradicional “deberá”, por ejemplo: “El pasajero podrá imprimir tarjetas de embarque para todos los segmentos de vuelo para los que se haya registrado” o “Si el perfil del pasajero no indica una preferencia de asiento, el sistema de reservas asignará un asiento.” El analista de negocio es el que documenta los requisitos en una Especificación de Requisitos de Software (ERS) que describe tan completamente como sea necesario el comportamiento esperado del sistema de software. El ERS se utiliza en el desarrollo, las pruebas, el aseguramiento de la calidad, la gestión de proyectos y funciones de proyectos relacionadas. Un ERS podría ser un informe generado a partir de información almacenada en una herramienta de gestión de requisitos.

1.1.7. Requisitos de sistema

Describen los requisitos para un producto que se compone de múltiples componentes o subsistemas. El sistema puede ser todo el software o puede incluir subsistemas de software y hardware. Las **personas** y los **procesos** también forman parte de un sistema, por lo que ciertas funciones del sistema pueden asignarse a los seres humanos. Un buen ejemplo de un “**sistema**” es la estación de trabajo del cajero en un supermercado. Hay un escáner de código de barras integrado con una báscula, así como un escáner de código de barras de mano. El cajero tiene un teclado, una pantalla y una caja registradora. Podrá ver un lector de tarjetas y un teclado PIN para su tarjeta de fidelización y tarjeta de crédito o débito, y quizás un dispensador de cambio. Es posible que vea hasta tres impresoras para su recibo de compra, recibo de tarjeta de crédito y cupones que no le interesan. Todos estos dispositivos de hardware interactúan bajo el control del software. Los requisitos para el sistema o producto como un todo, entonces, llevan al analista de negocios a derivar funcionalidades específicas que deben asignarse a uno u otro de esos subsistemas componentes, además de exigir una comprensión de las interfaces entre ellos.

1.1.8. Reglas de negocio

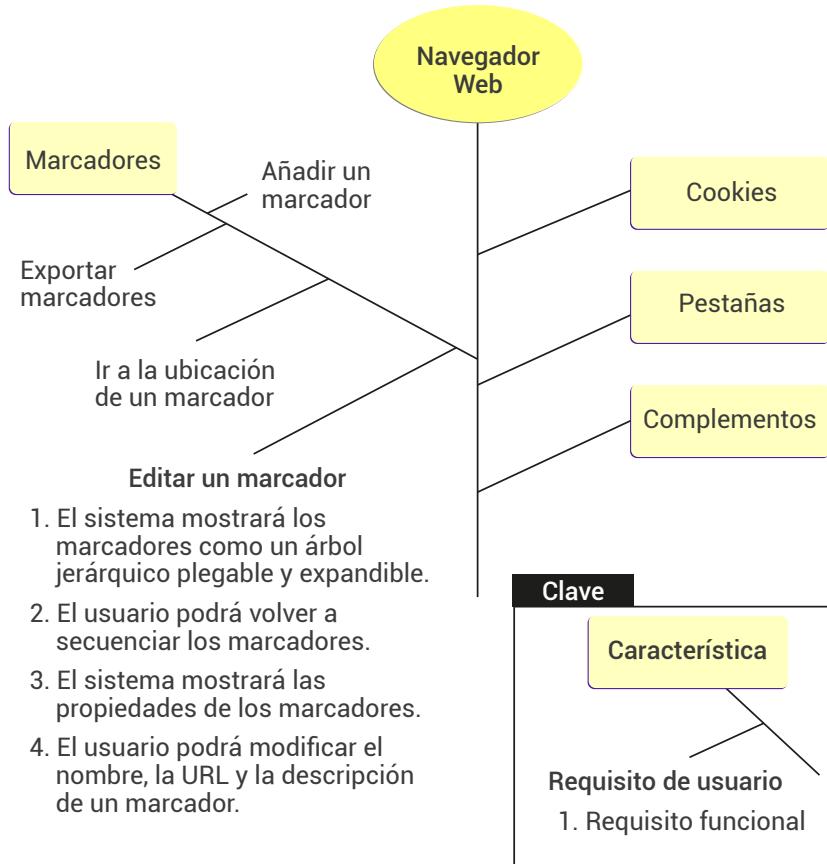
Incluyen políticas corporativas, regulaciones gubernamentales, estándares de la industria y algoritmos computacionales, las reglas de negocio no son en sí mismas requisitos de software porque tienen una existencia más allá de los límites de cualquier aplicación de software. Sin embargo, a menudo dictan la funcionalidad que debe tener el sistema para cumplir con las reglas pertinentes. A veces, como sucede con las políticas de seguridad corporativas, las reglas de negocio son el origen de atributos de calidad específicos que luego se implementan en la funcionalidad. Por lo tanto, puede rastrear la génesis de ciertos requisitos funcionales hasta una regla de negocio particular.

Una **característica** consta de una o más capacidades del sistema relacionadas lógicamente que proporcionan valor a un usuario y se describen mediante un conjunto de requisitos funcionales. La lista de características deseadas del producto de un cliente no es equivalente a una descripción de las necesidades relacionadas con la tarea del usuario. Los marcadores del navegador web, los correctores ortográficos, la capacidad de definir un programa de entrenamiento personalizado para un equipo de ejercicios y la actualización automática de firmas de virus en un producto antimalware son ejemplos de características. Una característica puede abarcar múltiples requisitos de usuario, cada uno de los cuales implica que se deben implementar ciertos requisitos funcionales para permitir que el usuario realice la tarea descrita por cada requisito de usuario.

La figura 4 muestra un ejemplo de árbol de características. Un modelo de análisis que muestra cómo una característica puede descomponerse jerárquicamente en un conjunto de características más pequeñas, que se relacionan con los requisitos específicos del usuario y conducen a la especificación de conjuntos de requisitos funcionales (Chen & Beatty, 2012).

Figura 4

Ejemplo de relación entre características, requisitos de usuario y requisitos funcionales

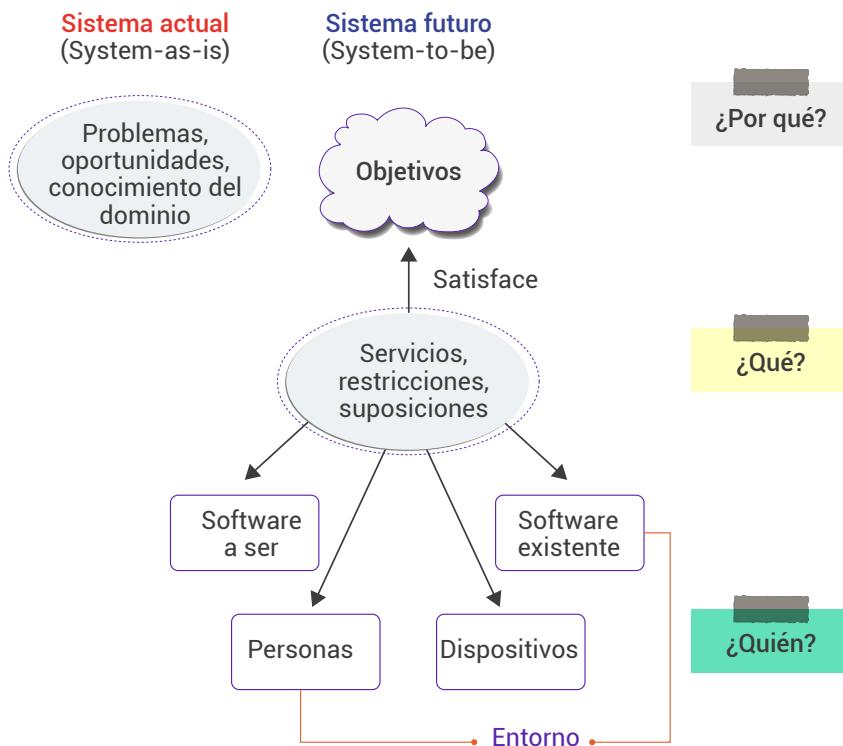


Nota. Adaptado de Relaciones entre las características, los requisitos del usuario y los requisitos funcionales. (p. 11), por K. Wiegers y J. Beatty, 2013, Microsoft Press.

El ejemplo de la figura 4, muestra las características: marcadores, cookies, pestañas y complementos. Luego para los marcadores los requisitos de usuario: añadir un marcador, exportar marcadores, ir a la ubicación de un marcador y editar un marcador. Esto permite definir los requisitos funcionales que se indican en el recuadro.

Para un perfecto desarrollo debe considerar siempre la existencia de un sistema o situación actual. Esto permite asegurar que un producto *software* resuelva satisfactoriamente un problema. Esto a simple vista puede parecer de sentido común, pero como se detalla más adelante se requiere descubrir, entender, formular, analizar y acordar sobre “**qué**” problema debe ser resuelto, “**por qué**” tal problema necesita ser resuelto y “**quién**” podría ser el responsable de solucionar el problema (Gottesdiener, 2005).

Figura 5
Relación entre los requisitos



Nota: Adaptado de *Requirements Happens* (p. 11), por Lawrence, B., 1997.

Los requisitos se definen de acuerdo con la información que aporte cada interesado y el nivel de especificación que se realice a cada uno de ellos, por tal motivo a partir de unos objetivos se puede llegar a especificar cada requisito, tal como se especifica en la figura 5. A continuación se describen las dimensiones que permiten aclarar la definición de estos requisitos.

1.1.9. Dimensión ¿Por qué?

La razón fundamental para que una nueva versión de un sistema deba hacerse, explícitamente, se realizará en términos de los objetivos a ser satisfechos por esta. Tales objetivos deben ser identificados con respecto a las limitaciones del sistema actual y de las oportunidades a ser explotadas. Esto requiere de un análisis cuidadoso.

1.1.10. La dimensión ¿Qué?

Esta dimensión de especificación de requisitos está preocupada por los “servicios funcionales” que el sistema a futuro podría proveer para satisfacer los objetivos identificados a través de la dimensión “por qué”. Tales servicios a menudo se basan en especificaciones supuestas del sistema para trabajar adecuadamente. Ellos necesitan reunir restricciones relacionadas con el funcionamiento, seguridad, usabilidad, interoperabilidad, costo, entre otros. Algunos de los servicios podrían ser implementados por el *software* futuro (*software-to-be*) en tanto que otros pueden ser realizados a través de procedimientos manuales u operaciones de dispositivo.

1.1.11. La dimensión ¿Quién?

Esta dimensión de Especificación de Requisitos dirige la asignación de responsabilidades para lograr el objetivo, servicios y restricciones a través de los componentes del sistema futuro – humanos, dispositivos o *software*. Las decisiones acerca de la asignación de responsabilidades son a menudo críticas; un importante objetivo, servicio o restricción podría no ser alcanzado si la responsabilidad del componente del sistema deja de comportarse como corresponde.

1.2. Actividades de la Ingeniería de Requisitos

Tal como se mencionó anteriormente, existe cierta confusión en el uso de la terminología de los requisitos que incluso afectan a esta disciplina. Algunos autores utilizan el término Ingeniería de Requisitos, mientras que otros autores suelen utilizar el término Gestión de Requisitos; otros autores se refieren a estas actividades como un subconjunto del amplio dominio del análisis de negocio. Resulta útil dividir la IR en desarrollo de requisitos y gestión de requisitos tal como se muestra en la figura 5, independientemente del ciclo de vida de desarrollo que se utilice,

las actividades de desarrollo de requisitos son: Obtención, análisis, especificación y validación. Podemos apreciar en el siguiente recurso las actividades de la Ingeniería de Requisitos:

Actividades de la IR

Estas son las actividades de desarrollo las que permitirán definir los requisitos de sistema que se verán plasmados en el ERS. El desarrollo de cada una de estas actividades requiere de incorporar un conjunto de estrategias y herramientas que permitan gestionar adecuadamente la información de las diferentes fuentes. A continuación se describe brevemente cada una de estas actividades.

1.2.1. Obtención:

También llamada Elicitación, comprende aquellas estrategias que se utilizan para descubrir los requisitos, como es: entrevistas, talleres, análisis de documentación, prototipado y otras más. Entre las actividades clave tenemos:

- Identificar los usuarios clave e interesados del producto.
- Comprender las tareas y objetivos del usuario y los objetivos de negocio con las que se alinean las tareas.
- Conocer el entorno o contexto en el que se utilizará el nuevo producto.
- Trabajar con personas que representan a cada clase de usuario para comprender sus necesidades de funcionalidad y sus expectativas de calidad.

1.2.2. Análisis:

Consiste en llegar a una comprensión efectiva de cada requisito para poder representarlo por categorías de múltiples maneras. Las actividades principales son:

- Analizar la información proveniente de los usuarios para distinguir los objetivos de trabajo de los requisitos funcionales, las expectativas de calidad, las reglas de negocio, las soluciones sugeridas y otra información.

- Descomponer los requisitos de alto nivel a un nivel de detalle apropiado.
- Derivación de requisitos funcionales desde otros requisitos de información.
- Comprensión de la importancia de los atributos de calidad.
- Asignación de requisitos a los componentes de software definidos en la arquitectura del sistema.
- Negociación de prioridades de implementación.
- Identificar lagunas en los requisitos o requisitos innecesarios en relación con el alcance definido.

1.2.3. Especificación:

Esta actividad implica representar y documentar los requisitos recopilados de manera persistente y bien organizada. La actividad principal es:

- Traducir el conjunto de necesidades de usuario, escribiendo requisitos y diagramas apropiados para comprender, revisar y utilizar por un público específico.

1.2.4. Validación:

Consiste en disponer del conjunto de requisitos de información correctos que permitirá a los desarrolladores crear una solución que satisfaga los objetivos del negocio. Las actividades principales son:

- Revisar los requisitos documentados para corregir algún problema antes que el grupo de desarrollo los acepte.
- Desarrollar pruebas y criterios de aceptación para confirmar que un producto basado en requisitos podría satisfacer las necesidades del cliente y lograr los objetivos del negocio.

La iteración es la clave para el éxito del desarrollo de requisitos. Planifique múltiples ciclos de **exploración de requisitos**, refinando progresivamente los requisitos de alto nivel con mayor precisión y detalle, y confirmando la corrección con los usuarios. Esto toma tiempo y puede ser frustrante. No

obstante, es un aspecto intrínseco de lidiar con la incertidumbre difusa de definir un nuevo sistema de software.

Estimado/a estudiante, con el fin de ampliar la información, le invito a realizar la siguiente actividad de aprendizaje.



Actividad de aprendizaje recomendada

1. Revise el video [Ingeniería de requerimientos ¿Qué son los requerimientos de software?](#) De la serie Ingeniería de software de élite.

Como podrá haberse dado cuenta, las definiciones que se indican de los requisitos están orientadas al desarrollo de software para apoyar las actividades que son parte de un proceso en las organizaciones. Se recalca sobre las actividades que son parte del desarrollo y lo importante de utilizar las estrategias de obtención para entender lo que realmente necesita el usuario.



Semana 2

1.3. Buenas prácticas de la Ingeniería de Requisitos

Para el desarrollo de las actividades a la hora de construir un software, cada profesional necesita de conocimiento, técnicas y estrategias que le permita cumplir con las actividades encomendadas. Los profesionales que carecen de estas habilidades se ven obligados a inventar un método basado en lo que parece razonable en ese momento, pero que rara vez produce buenos resultados. Seguir un guion que brinda una metodología no es suficiente, se requiere identificar y aplicar las mejores prácticas de la industria (Wieggers & Beatty, 2006).

El enfoque de mejores prácticas considera un kit de herramientas de software con una variedad de técnicas que se puede aplicar a diversos problemas. La noción de mejores prácticas es discutible: ¿quién decide qué es lo “mejor” y con base en qué? Un enfoque es convocar a un cuerpo de expertos de la industria para analizar proyectos de muchas organizaciones. Estos expertos buscan prácticas cuyo desempeño efectivo esté asociado

con proyectos exitosos y muy raramente en proyectos fallidos. A través de estos medios, los expertos llegan a un consenso sobre las actividades que consistentemente arrojan resultados superiores y las etiquetan como **mejores prácticas**.

A continuación, en la infografía se lista las buenas prácticas asociadas a la ingeniería de requisitos, las cuales se agrupan en siete grupos, cuatro relacionadas con el desarrollo de requisitos y tres relacionadas con la gestión de requisitos.

Buenas prácticas para la ingeniería de requisitos

Estas prácticas que se indican en la infografía, varias de ellas contribuyen a más de una categoría, pero cada práctica aparece solo una vez en la infografía. La mayoría de estas prácticas contribuyen a una comunicación más eficaz entre los Interesados del proyecto. Tenga en cuenta que esto se denomina “Buenas prácticas para la Ingeniería de Requisitos”, no “Mejores prácticas”.

1.4. El analista de negocio

El analista de negocios (BA, por sus siglas en inglés), es la persona que tiene la responsabilidad principal de obtener, analizar, documentar y validar las necesidades de los Interesados del proyecto. Actúa como el principal intérprete a través del cual fluyen los requisitos entre los clientes y el equipo de desarrollo de software, tal como se muestra en la figura 6, aunque también se utilizan otras vías de comunicación, por lo que el analista no es el único responsable del intercambio de información sobre el proyecto. El BA es el responsable en la recopilación y difusión de información del producto, mientras que el director del proyecto toma la iniciativa en la comunicación de la información del proyecto.

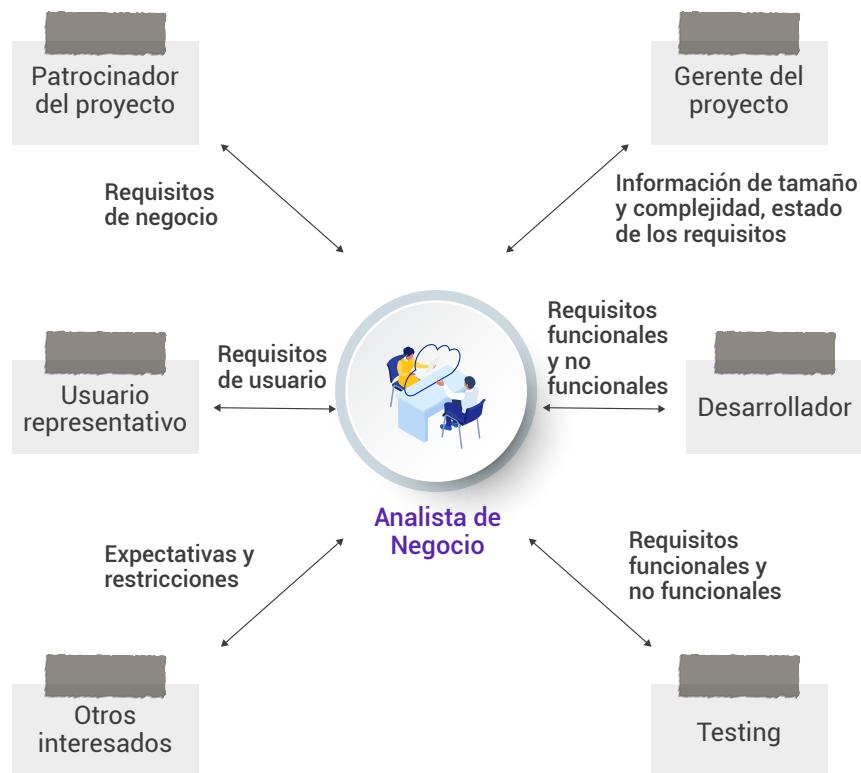
El analista de negocio es un rol del proyecto y no necesariamente un título de trabajo. Los sinónimos de analista de negocios incluyen analista de requisitos, analista de sistemas, ingeniero de requisitos, administrador de requisitos, analista de aplicaciones, analista de sistemas de negocios, analista de negocios de TI o simplemente analista. Estos títulos de trabajo se usan de manera inconsistente de una organización a otra.

Es importante tener en cuenta que cuando una persona que tiene otra función en el proyecto también se puede desempeñar como analista de negocio, en este caso está realizando dos trabajos distintos. Por ejemplo, considere un gerente del proyecto que también podría desempeñarse como el BA en el proyecto.

El analista de negocio se relaciona con diferentes interesados y miembros del equipo de desarrollo para realizar de manera eficiente cada una de sus funciones, tal como se indica en la figura 6.

Figura 6

El analista de negocio



Nota: Adaptado de *Software Requirements* (p. 62), por Wiegers, K. y Beatty, J., 2013. Microsoft Press.

Un gerente de proyecto necesita crear y administrar la planificación, incluidos los cronogramas y las necesidades de recursos, en función del trabajo que definen los BA. El director del proyecto debe ayudar a gestionar el alcance y hacer frente a los cambios en el cronograma a medida que evoluciona el alcance. Puede desempeñar la función de gestión de proyectos en un minuto y al siguiente minuto cambiar de sombrero para ejecutar las prácticas de analista. Pero estos son roles distintos, que requieren conjuntos de habilidades algo diferentes.

Un analista talentoso puede marcar la diferencia entre un proyecto que tiene éxito y uno que tiene dificultades. Una empresa descubrió que podía inspeccionar las especificaciones de requisitos escritas por analistas experimentados el doble de rápido que las escritas por novatos porque contenían menos defectos. En el popular modelo Cocomo II para la estimación de proyectos, la experiencia y la capacidad del analista tienen una gran influencia en el esfuerzo y el costo de un proyecto (Boehm et al., 2009). El uso de analistas altamente experimentados puede reducir el esfuerzo general del proyecto en un tercio en comparación con proyectos similares con analistas sin experiencia.

1.4.1. Tareas del analista de negocio

El analista primero debe comprender los objetivos de negocio del proyecto y luego definir los requisitos de usuario, funcionales y de calidad que permitan a los equipos estimar y planificar el proyecto y diseñar, construir y verificar el producto. El analista de negocio también es un líder y un comunicador, que convierte las vagas nociones de los clientes en especificaciones claras que guían el trabajo del equipo de software. A continuación, se describe algunas de las actividades típicas que podría realizar el BA.

1.4.1.1. Definir los requisitos de negocio

Su trabajo como BA, comienza cuando ayuda al patrocinador comercial o financiero, al gerente de producto o al gerente de *marketing* a definir los requisitos comerciales del proyecto. Puede sugerir una plantilla para un documento de visión y alcance y trabajar con quienes tienen la visión para ayudarlos a expresar con claridad.

1.4.1.2. Planificar el enfoque de requisitos

El analista debe desarrollar planes para obtener, analizar, documentar, validar y gestionar los requisitos a lo largo del proyecto. Trabaja en estrecha colaboración con el gerente del proyecto para garantizar que estos planes se alineen con los planes generales del proyecto y que contribuyan a lograr los objetivos del proyecto.

1.4.1.3. Identificar los Interesados del proyecto y las clases de usuarios

El BA debe trabajar directamente con los patrocinadores del proyecto para seleccionar representantes apropiados para cada clase de usuario, obtenga su participación y negocie sus responsabilidades.

1.4.1.4. Obtener requisitos

Un analista proactivo ayuda a los usuarios a articular las capacidades del sistema que necesitan para cumplir con sus objetivos de negocio mediante el uso de una variedad de técnicas de recopilación de información.

1.4.1.5. Analizar requisitos

El BA busca requisitos derivados que sean una consecuencia lógica de lo que solicitaron los clientes y requisitos implícitos que los clientes parecen esperar sin decirlo. Utiliza modelos de requisitos para reconocer patrones, identificar brechas en los requisitos, revelar requisitos conflictivos y confirmar que todos los requisitos especificados están dentro del alcance. Trabaje con los Interesados para determinar el nivel de detalle necesario para especificar los requisitos funcionales y de usuario.

1.4.1.6. Documentar requisitos

El analista es responsable de documentar los requisitos de una manera bien organizada y escrita que describa claramente la solución que abordará el problema del cliente. El uso de plantillas estándar acelera el desarrollo de requisitos al recordarle al BA los temas a discutir con los representantes de los usuarios.

1.4.1.7. Comunicar requisitos

Debe comunicar los requisitos de manera eficaz y eficiente a todas las partes. El BA debe determinar cuándo es útil representar los requisitos

mediante el uso de métodos distintos al texto, incluidos varios tipos de modelos de análisis visual, tablas, ecuaciones matemáticas y prototipos. La comunicación no es simplemente una cuestión de poner requisitos en papel y tirarlos por encima de una pared. Implica una colaboración continua con el equipo para garantizar que comprendan la información que está comunicando.

1.4.1.8. Validación de requisitos

El BA debe asegurarse de que las declaraciones de requisitos posean las características deseadas y que una solución basada en los requisitos satisfaga las necesidades de los interesados. Los analistas son los participantes centrales en las revisiones de requisitos. También debe revisar los diseños y las pruebas que se derivaron de los requisitos para asegurarse de que los requisitos se interpretaron correctamente. Si está creando pruebas de aceptación en lugar de requisitos detallados en un proyecto ágil, también se deben revisar.

1.4.1.9. Facilitar la priorización de requisitos

El analista negocia la colaboración y la negociación entre las diversas partes interesadas y los desarrolladores para garantizar que tomen decisiones prioritarias sensatas en consonancia con el logro de los objetivos comerciales.

1.4.1.10. Administrar requisitos

Un analista de negocios está involucrado durante todo el ciclo de vida del desarrollo de software, por lo que debe ayudar a crear, revisar y ejecutar el plan de gestión de requisitos del proyecto. Después de establecer una línea de base de requisitos para una versión de producto determinada o una iteración de desarrollo, el enfoque de BA cambia al seguimiento del estado de esos requisitos, verificando su satisfacción en el producto y gestionando los cambios en la línea de base de requisitos. Con aportes de varios colegas, el analista recopila información de trazabilidad que conecta los requisitos individuales con otros elementos del sistema.

1.4.2. Habilidades del analista

Sin la capacitación, tutoría y experiencia adecuadas, las personas no pueden realizar tareas de analista, no harán bien su trabajo y la experiencia será frustrante. Se requiere de habilidades que debe tener las personas

para utilizar una variedad de técnicas para obtener información y luego representar mediante modelos, diferentes al lenguaje natural que sea comprensible para los interesados. Los analistas de negocios combinan habilidades sólidas de comunicación, facilitación y trato con las personas con experiencia técnica y una personalidad apropiada para el área de negocio y el proyecto. La paciencia y un deseo genuino de trabajar con personas son importantes factores de éxito. Existen ciertas habilidades para analistas de requisitos de nivel bajo, medio y alto (Young, 2004). Ahora, le animo a revisar las habilidades más importantes que se describen a continuación:

1.4.2.1. Habilidades de escuchar

Para dominar la comunicación bidireccional, aprenda a escuchar con eficacia. La escucha activa implica eliminar las distracciones, mantener una postura atenta y el contacto visual, y reafirmar los puntos clave para confirmar su comprensión. Debe comprender lo que dice la gente y también leer entre líneas para detectar lo que podrían dudar en decir. Conozca cómo prefieren comunicarse sus colaboradores y evite imponer su filtro personal de comprensión sobre lo que escucha de los clientes. Esté atento a las suposiciones no declaradas que subyacen en lo que escucha de otros o en su propia interpretación.

1.4.2.2. Habilidades para entrevistar y hacer preguntas

La mayoría de los aportes de requisitos provienen de discusiones, por lo que el BA debe poder interactuar con diversos individuos y grupos sobre sus necesidades. Puede ser intimidante, trabajar con altos directivos y con personas muy testarudas o agresivas. Debe hacer las preguntas correctas para obtener información sobre los requisitos esenciales. Por ejemplo, los usuarios se enfocan naturalmente en los comportamientos esperados normales del sistema. Sin embargo, se escribe mucho código para manejar excepciones. Por lo tanto, también debe sondear para identificar las condiciones de error y determinar cómo debe responder el sistema. Con la experiencia, adquirirá destreza en el arte de hacer preguntas que revelen y aclaren incertidumbres, desacuerdos, suposiciones y expectativas no declaradas (Gause et al., 1989).

Pensando en los intereses

Los analistas de negocios siempre deben estar al tanto de la información existente y procesar nueva información en función de ella. Necesitan detectar contradicciones, incertidumbre, vaguedad y suposiciones para poder discutirlas en el momento si corresponde. Puede intentar escribir el conjunto perfecto de preguntas de la entrevista; sin embargo, siempre tendrás que preguntar algo que no podrías haber previsto. Debe redactar buenas preguntas, escuchar claramente las respuestas y pensar rápidamente en la siguiente cosa inteligente para decir o preguntar. A veces, no hará una pregunta, sino que dará un ejemplo apropiado en contexto para ayudar a su parte interesada a formular la siguiente respuesta.

Capacidad de análisis

Un analista de negocios eficaz puede pensar tanto en niveles altos como bajos de abstracción y sabe cuándo pasar de uno a otro. A veces, debe profundizar desde la información de alto nivel hasta los detalles. En otras situaciones, deberá generalizar a partir de una necesidad específica que un usuario describió a un conjunto de requisitos que satisfarán a múltiples partes interesadas. Los BA necesitan comprender información compleja proveniente de muchas fuentes y resolver problemas difíciles relacionados con esa información. Necesitan evaluar críticamente la información para reconciliar conflictos, separar los “deseos” del usuario de las verdaderas necesidades subyacentes y distinguir las ideas de solución de los requisitos.

1.4.2.3. Habilidades de pensamiento sistemático

Aunque un analista de negocios debe estar orientado a los detalles, también debe ver el panorama general. El BA debe comparar los requisitos con lo que sabe sobre toda la empresa, el entorno comercial y la aplicación para buscar inconsistencias e impactos. El BA necesita comprender las interacciones y relaciones entre las personas, los procesos y la tecnología relacionada con el sistema (IIBA, 2015). Si un cliente solicita un requisito para su área funcional, el BA debe juzgar si el requisito afecta a otras partes del sistema de manera no obvia.

1.4.2.4. Habilidades de aprendizaje

Los analistas deben aprender material nuevo rápidamente, ya sea sobre nuevos enfoques de requisitos o el dominio de la aplicación. Necesitan ser

capaces de traducir ese conocimiento a la práctica de manera eficiente. Los analistas deben ser lectores eficientes y críticos porque tienen que leer mucho material y captar la esencia rápidamente. No tienes que ser un experto en el dominio, así que no dudes en hacer preguntas aclaratorias. Sé honesto sobre lo que no sabes. Está bien no saberlo todo, pero no está bien ocultar tu ignorancia.

1.4.2.5. Habilidades de facilitación

La capacidad de facilitar las discusiones de requisitos y los talleres de elic平ación es una capacidad vital del analista. La facilitación es el acto de conducir a un grupo hacia el éxito. La facilitación es esencial cuando se definen requisitos de forma colaborativa, se priorizan necesidades y se resuelven conflictos. Un facilitador neutral que tenga fuertes habilidades de cuestionamiento, observación y facilitación puede ayudar a un grupo a generar confianza y mejorar la relación a veces tensa entre el personal de negocios y de TI.

1.4.2.6. Habilidades de liderazgo

Un analista fuerte puede influir en un grupo de partes interesadas para que se muevan en una dirección determinada para lograr un objetivo común. El liderazgo requiere comprender una variedad de técnicas para negociar acuerdos entre las partes interesadas del proyecto, resolver conflictos y tomar decisiones. El analista debe crear un entorno de colaboración, fomentando la confianza entre los diversos grupos de partes interesadas que pueden no entender las motivaciones, necesidades y limitaciones de los demás.

1.4.2.7. Habilidades de observación

Un analista observador detectará comentarios hechos de pasada que podrían resultar significativos. Al observar a un usuario realizar su trabajo o usar una aplicación actual, un buen observador puede detectar sutilezas que el usuario podría no mencionar. Las fuertes habilidades de observación a veces exponen nuevas áreas para discusiones de elic平ación, revelando así requisitos adicionales.

1.4.2.8. Habilidades de comunicación

El principal resultado del desarrollo de requisitos es un conjunto de requisitos escritos que comunica la información de manera eficaz entre los

clientes, el departamento de marketing, los gerentes y el personal técnico. El analista necesita un dominio sólido del idioma y la capacidad de expresar ideas complejas con claridad, tanto en forma escrita como verbal. Debe poder escribir para múltiples audiencias, incluidos los clientes que tienen que validar los requisitos y los desarrolladores que necesitan requisitos claros y precisos para la implementación. Un BA necesita hablar claramente, adaptándose a la terminología local ya las diferencias regionales en el dialecto. Además, un BA debe poder resumir y presentar información al nivel de detalle que necesita el público objetivo.

1.4.2.9. Habilidades organizativas

Los BA deben lidiar con una amplia gama de información confusa recopilada durante la obtención y el análisis. Hacer frente a información que cambia rápidamente y estructurar todas las partes en un todo coherente exige habilidades organizativas excepcionales y paciencia y tenacidad para encontrarle sentido a la ambigüedad y el desorden. Como analista, debe poder configurar una arquitectura de información para respaldar la información del proyecto a medida que crece a lo largo del proyecto (Chen & Beatty, 2012).

1.4.2.10. Habilidades de modelado

Los modelos que van desde el venerable diagrama de flujo hasta los modelos de análisis estructurado (diagrama de flujo de datos, diagrama de entidad-relación y diagramas similares) hasta las notaciones del lenguaje de modelado unificado (UML) deben ser parte del repertorio de cada analista (Chen & Beatty, 2012). Algunos serán útiles cuando se comuniquen con los usuarios, otros cuando se comuniquen con los desarrolladores y otros únicamente para el análisis para ayudar a la BA a mejorar los requisitos. El BA necesitará saber cuándo seleccionar modelos específicos en función de cómo agregan valor. Además, deberá educar a otras partes interesadas sobre el valor de usar estos modelos y cómo leerlos.

1.4.2.11. Habilidades interpersonales

Los analistas deben ser capaces de hacer que las personas con intereses contrapuestos trabajen juntas como un equipo. Un analista debe sentirse cómodo hablando con personas en diversas funciones laborales y en todos los niveles de la organización. Un BA debe hablar el idioma de la audiencia a la que se dirige, sin usar jerga técnica con las partes interesadas del

negocio. Es posible que necesite trabajar con equipos virtuales cuyos miembros estén separados por geografía, zonas horarias, culturas o idiomas nativos. Debe ser fácil comunicarse con un BA y ser claro y coherente al comunicarse con los miembros del equipo.

Creatividad

El BA no es simplemente un escriba que registra todo lo que dicen los clientes. Los mejores analistas inventan requisitos potenciales para que los clientes los consideren (Robertson & Robertson, 2013). Conciben capacidades de productos innovadores, imaginan nuevos mercados y oportunidades comerciales, y piensan en formas de sorprender y deleitar a sus clientes. Un BA realmente valioso encuentra formas creativas de satisfacer necesidades que los usuarios ni siquiera sabían que tenían. Los analistas pueden ofrecer nuevas ideas porque no están tan cerca como los usuarios del problema que se está resolviendo. Sin embargo, los analistas deben tener cuidado de evitar recubrir la solución; no agregue simplemente nuevos requisitos a la especificación sin la aprobación del cliente.

1.4.3. Conocimiento esencial del analista

Además de tener capacidades específicas y características personales, los analistas de negocios necesitan una amplitud de conocimientos, muchos de los cuales se obtienen a través de la experiencia. Necesitan comprender las prácticas de ingeniería de requisitos contemporánea y cómo aplicarlas en el contexto de varios ciclos de vida de desarrollo de software. Es posible que necesiten educar y persuadir a aquellos que no están familiarizados con las prácticas de requisitos establecidos. El analista efectivo tiene un rico conjunto de herramientas de técnicas disponibles y sabe cuándo, y cuándo no, usar cada una.

Los BA deben enhebrar las actividades de gestión y desarrollo de requisitos a lo largo de toda la vida útil del proyecto. Un analista con una sólida comprensión de la gestión de proyectos, los ciclos de vida del desarrollo, la gestión de riesgos y la ingeniería de calidad puede ayudar a evitar que los problemas de requisitos destruyan el proyecto. En un entorno de desarrollo comercial, el BA se beneficiará del conocimiento de los conceptos de gestión de productos. Los BA se benefician de un nivel básico de conocimiento sobre la arquitectura y el entorno operativo, para que puedan participar en conversaciones técnicas sobre prioridades y requisitos no funcionales.

El conocimiento del negocio, la industria y la organización son activos poderosos para un BA efectivo (IIBA, 2015). El analista experto en negocios puede minimizar los errores de comunicación con los usuarios. Los analistas que entienden la organización y los dominios comerciales a menudo detectan suposiciones no declaradas y requisitos implícitos. Pueden sugerir formas en que los usuarios podrían mejorar sus procesos comerciales o proponer funcionalidades valiosas que ninguna otra parte interesada pensó. Comprender el dominio de la industria puede ser particularmente útil en un entorno comercial para que los analistas puedan ofrecer un análisis de mercado y de productos competitivos.

1.4.4. El rol del analista en proyectos ágiles

En los proyectos que utilizan métodos de desarrollo ágiles, las funciones de analista de negocios aún deben realizarse, pero es posible que la persona que las realiza no se llame BA. Algunos enfoques ágiles tienen un miembro clave del equipo llamado propietario del producto. La persona en ese rol podría realizar algunas de las actividades tradicionales de análisis de negocios, además de proporcionar la visión del producto, comunicar las restricciones, priorizar la acumulación de trabajo pendiente del producto y tomar las decisiones finales sobre el producto (Cohn, 2010). Otros proyectos mantienen un rol de analista comercial separado del propietario del producto. Además, otros miembros del equipo, como los desarrolladores, realizan partes del rol de analista. El punto es que, independientemente del enfoque de desarrollo del proyecto, las tareas asociadas con el rol de BA aún deben realizarse. El equipo se beneficiará de tener miembros que posean las habilidades asociadas con los analistas de negocios.

A menudo, en una organización que avanza hacia un enfoque de desarrollo ágil, el BA no está seguro de cómo puede contribuir de manera más efectiva al proyecto. En el espíritu del desarrollo ágil, el analista debe estar dispuesto a salir de un rol preconcebido de “analista de negocios” y completar lo que sea necesario para ayudar a entregar un producto exitoso. Ellen Gottesdiener (2009), ofrece una lista detallada de cómo las actividades tradicionales de los analistas de negocios pueden adaptarse a un entorno ágil. Las siguientes son algunas sugerencias para que un BA aplique sus habilidades en un proyecto ágil:

- Defina un proceso de requisitos ligero y flexible y adáptelo según lo requiera el proyecto.

- Asegúrese de que la documentación de requisitos esté en el nivel correcto: ni muy poco ni demasiado. (Muchos BA tienden a documentar todo en especificaciones hasta el enésimo grado. Algunos puristas sugieren que los proyectos ágiles deberían tener poca o ninguna documentación de requisitos. Ninguno de los extremos es ideal).
- Ayude a determinar el mejor enfoque para documentar el atraso, incluso si las tarjetas de historias o herramientas más formales son las más apropiadas.
- Aplique habilidades de facilitación y liderazgo para garantizar que las partes interesadas hablen entre sí con frecuencia sobre requisitos, necesidades, preguntas e inquietudes.
- Ayude a validar que las necesidades del cliente estén representadas con precisión en la cartera de productos y facilite la priorización de la cartera de pedidos.
- Trabaje con los clientes cuando cambien de opinión acerca de los requisitos y las prioridades, y ayude a registrar esos cambios. Trabaje con el resto del equipo para determinar el impacto de los cambios en los contenidos de iteración y los planes de lanzamiento.

Tiene mucho valor tener un rol como el propietario del producto para representar a los usuarios a lo largo del desarrollo. Sin embargo, es posible que la persona que desempeña el rol de propietario del producto no tenga todas las habilidades de análisis comercial o el tiempo para realizar todas las actividades relacionadas. Un BA puede aportar esas capacidades críticas al equipo.

1.4.5. El analista de negocio en la creación de un equipo colaborativo

Los proyectos de software a veces experimentan relaciones tensas entre analistas, desarrolladores, usuarios, gerentes y *marketing*. Las partes no siempre confían en las motivaciones de la otra parte ni aprecian las necesidades y limitaciones de la otra parte. Sin embargo, en realidad, los productores y consumidores de un producto de software comparten objetivos comunes. Para el desarrollo de sistemas de información corporativos, todas las partes trabajan para la misma empresa, por lo que todos se benefician de las mejoras en los resultados corporativos. Para los

productos comerciales, los clientes satisfechos generan ingresos para el productor y satisfacción para los desarrolladores.

El analista comercial tiene la responsabilidad principal de forjar una relación de colaboración entre los representantes de los usuarios y otras partes interesadas del proyecto. Un analista eficaz aprecia los desafíos que enfrentan las partes interesadas tanto comerciales como técnicas y demuestra respeto por sus colaboradores en todo momento. El analista dirige a los participantes del proyecto hacia un acuerdo de requisitos que conduce a un resultado de ganar-ganar-ganar de las siguientes maneras:

- Los clientes están encantados con el producto.
- La organización en desarrollo está satisfecha con los resultados comerciales.
- Todos los miembros del equipo están orgullosos del buen trabajo que hicieron en un proyecto desafiante y gratificante.



Actividades de aprendizaje recomendadas

Estimado estudiante, finalizada la unidad 1 y para afianzar los conceptos de esta unidad, se recomienda el desarrollo de las siguientes actividades:

1. Identifique las actividades que sugiere el SWEBOK, para la definición de los requisitos. Realice un esquema para que cubra las actividades de gestión y desarrollo. Revise [Guide to the Software Engineering Body of Knowledge, Version 3.0 SWEBOK. IEEE Computer Society](#).
2. Busque en la web el artículo “¿Analista de negocio, de sistemas, o de procesos?” de Norberto Figuerola y analice detenidamente los conceptos relacionados con el analista de negocio desde distintos puntos de vista. Realice un mapa conceptual de las definiciones.

Nota. Conteste las actividades en un cuaderno de apuntes o en un documento Word.

3. Le invito a desarrollar la autoevaluación 1.



Autoevaluación 1

Lea detenidamente cada uno de los literales y elija la opción correcta:

1. El comportamiento externo del sistema, se define desde el punto de vista del:
 - a. Usuario.
 - b. Desarrollador.
 - c. Analista.

2. A la descripción de una relación (conexión) entre un sistema *software* y el usuario, se conoce como:
 - a. Requisitos de negocio.
 - b. Regla de negocio.
 - c. Requisito de interfaz externa.

3. Los requisitos que representan los objetivos de alto nivel de la organización se conocen como:
 - a. Usuario.
 - b. Funcional.
 - c. Negocio.

4. La persona que traduce las necesidades del cliente (en función de los objetivos estratégicos) y dan origen, entre otras cosas, a los proyectos de tecnología informática, se conoce como:
 - a. Tester.
 - b. Analista de negocio.
 - c. Desarrollador.

5. La persona que está más preocupada por el diseño e implementación del *software*, se le conoce como analista de:
 - a. Sistema.
 - b. Proyecto.
 - c. Negocio.

6. Una de las actividades comunes del analista de negocio es:
 - a. Identificar a los interesados del proyecto y las clases de usuarios.
 - b. Diseñar la arquitectura del software.
 - c. Asesora en el diseño de la base de datos.
7. El analista, al desarrollar un producto. ¿Qué es lo que tiene que considerar para que el usuario compre o acepte el producto de software?
 - a. Que no sea costoso.
 - b. Que la organización se adapte al software.
 - c. Que el producto sea valioso, útil y satisfactorio.
8. El analista de negocio con el patrocinador intercambia información relacionada con:
 - a. Requisitos funcionales.
 - b. Requisitos de negocio.
 - c. Plan de pruebas.
9. Identificar a los interesados del producto, es una de las actividades que se realiza en la fase de:
 - a. Obtención.
 - b. Análisis.
 - c. Especificación.
10. Descomponer los requisitos de alto nivel a descripciones más detalladas, es una de las actividades que se realiza en la fase de:
 - a. Obtención.
 - b. Análisis.
 - c. Especificación.

[Ir al solucionario](#)



Unidad 2. Definiendo los requisitos de negocio

Antes de que proceda a desarrollar los requisitos de software, debe realizar ciertas actividades que le permita establecer una comprensión compartida del producto y sus interesados, con el objeto de “preparar el escenario” para un desarrollo eficaz del software. Para establecer esta comprensión compartida y facilitar el proceso de desarrollo de requisitos, necesita definir una visión común para el producto entre los interesados y deberá establecer una estrategia para identificar y tratar cualquier riesgo relacionado con los requisitos que pueda encontrar (Gottesdiener, 2005).

Wiegers (1996), indica que sin una participación adecuada del cliente, el resultado al final del proyecto es una “brecha de expectativas”, un abismo entre lo que los clientes realmente necesitan y lo que los desarrolladores ofrecen según lo que escucharon al comienzo del proyecto. La brecha de expectativas es una sorpresa desagradable para todos los Interesados. Los requisitos también quedan obsoletos debido a los cambios que ocurren en el negocio, por lo que las interacciones continuas con los clientes son vitales.

La mejor manera de minimizar la brecha de expectativas es organizar puntos de contacto frecuentes con representantes de clientes adecuados. Estos puntos de contacto pueden tomar la forma de entrevistas, conversaciones, revisiones de requisitos, tutoriales de diseño de interfaz de usuario, evaluaciones de prototipos y, con un desarrollo ágil, comentarios de los usuarios sobre pequeños incrementos de software ejecutable. Cada punto de contacto brinda la oportunidad de cerrar la brecha de expectativas, lo que los desarrolladores construyan algo que está más alineado con lo que necesita el cliente. Antes de que podamos hablar sobre los clientes, debemos hablar sobre los interesados.

2.1. Interesados

Un interesado es una persona, grupo u organización que participa activamente en un proyecto, que se ve afectado por su proceso o resultado,

o que puede influir en su proceso o resultado. Tendrá que aprender a usar técnicas de mapeo de interesados para identificar quiénes son sus interesados clave y asegurarse de cumplir con sus requisitos. Los interesados pueden ser internos o externos al equipo del proyecto y a la organización en desarrollo. La figura 7 identifica varios de los potenciales interesados que corresponden a estas categorías. Por supuesto, no todos estos se aplicarán a todos los proyectos o situaciones.

Figura 7
Potenciales Interesados para un proyecto de software

Fuera de la organización en desarrollo

- | | | |
|---------------------------|---------------------------|---------------------------|
| • Usuario directo | • Director de negocio | • Consultor |
| • Usuario indirecto | • Oficial de contratación | • Auditor |
| • Adquirente | • Agencia del gobierno | • Certificador |
| • Personal de adquisición | • Experto en la materia | • Cuerpo regulador |
| • Personal legal | • Director del programa | • Proveedor de software |
| • Contratista | • Probador beta | • Proveedor de materiales |
| • Subcontratista | • Público en general | • Capitalista de riesgo |

Organización en desarrollo

- | | | |
|---------------------------------|-------------------------|--|
| • Director de desarrollo | • Personal de ventas | • Sponsor ejecutivo |
| • Marketing | • Instalador | • Jefe de la oficina de proyectos |
| • Personal de soporte operativo | • Mantenedor | • Fabricante |
| • Personal legal | • Director del programa | • Personal de entrenamiento |
| • Arquitecto de información | • Experto en usabilidad | • Arquitecto de portafolio |
| • Dueño de la compañía | • Experto en la materia | • Personal de soporte de infraestructura |

Equipo del Proyecto

- | | |
|------------------------------|--|
| • Director del proyecto | • Tester |
| • Analista de negocio | • Director del producto |
| • Arquitecto de aplicaciones | • Personal de control de calidad |
| • Diseñador | • Escritor de documentación |
| • Desarrollador | • Administrador de Base de Datos |
| • Propietario del producto | • Ingeniero de Hardware |
| • Modelador de datos | • Analista de Infraestructura |
| • Analista de procesos | • Arquitecto de soluciones empresariales |

Nota. Adaptado de *Software Requirements* (p. 28), por Wiegers, K. y Beatty, J., 2013. Microsoft Press.

Como puede observar en la figura 7, existen diferentes interesados que aportan desde diferente ámbito al desarrollo del proyecto. Tal como se indicó anteriormente, estos interesados formarán parte de las actividades que se desarrollarán para el desarrollo de la solución, y precisamente el alcance y tipo de proyecto permitirá elegir estos interesados. Un análisis minucioso requiere del “cliente”, ya que es quien se verá afectado por las actividades del sistema; a continuación, se describen los interesados más relevantes.

2.1.1. Cliente

Los clientes son un subconjunto de los interesados. Un cliente es un individuo u organización que obtiene un beneficio directo o indirecto de un producto. Los clientes de software pueden solicitar, pagar, seleccionar, especificar, usar o recibir la salida generada por un producto de software. Los clientes que se pueden apreciar en la figura 7 incluye al usuario directo, al usuario indirecto, al patrocinador ejecutivo, al personal de adquisiciones y al adquirente. Algunos Interesados no son clientes, como es: el personal legal, los auditores de cumplimiento, los proveedores, los contratistas y los capitalistas de riesgo.

Los requisitos de los usuarios deben provenir de las personas que directa o indirectamente utilizarán el producto. Estos usuarios (a menudo denominados usuarios finales) son un subconjunto de los clientes. Los usuarios directos realmente utilizarán el producto, mientras que los usuarios indirectos pueden recibir los resultados del sistema sin tocarlo ellos mismos, como por ejemplo el gerente de almacén que recibe un informe automático de las actividades diarias del almacén por correo electrónico. Los usuarios pueden describir las tareas que deben realizar con el producto, los resultados que necesitan y las características de calidad que esperan que presente el producto.

Los requisitos de negocio deben provenir de la persona que es responsable en última instancia del valor comercial del producto. Los requisitos del usuario deben provenir de personas que manejan el sistema o recibirán los resultados. Si existe una desconexión entre los clientes compradores que pagan por el proyecto y los usuarios finales, se garantizan problemas importantes.

La situación es diferente para el desarrollo de software comercial, donde el cliente y el usuario suelen ser la misma persona. Los sustitutos de

los clientes, como el personal de *marketing* o un gerente de producto, generalmente intentan determinar qué les parecería atractivo a los clientes. Sin embargo, incluso para el software comercial, debe esforzarse por involucrar a los usuarios finales en el proceso de desarrollo de los requisitos del usuario.

Pueden surgir conflictos entre los interesados del proyecto. Los requisitos de negocio a veces reflejan estrategias organizacionales o restricciones presupuestarias que no son evidentes para los usuarios. Los usuarios que están molestos porque la administración les impone un nuevo sistema de información pueden no querer trabajar con los desarrolladores de software, viéndolos como los presagios de un futuro no deseado. Estas personas a veces se denominan “grupos de perdedores” (Gause et al., 1989). Para manejar tales conflictos potenciales, pruebe estrategias de comunicación sobre los objetivos y limitaciones del proyecto que puedan generar aceptación y evitar debates y resentimientos.

Los excelentes requisitos son el resultado de una colaboración eficaz entre los desarrolladores y los clientes (usuarios reales). Un esfuerzo colaborativo puede funcionar solo cuando todos los Interesados saben lo que necesitan para tener éxito y cuando entienden y respetan lo que sus colaboradores necesitan para tener éxito. A medida que aumentan las presiones del proyecto, es fácil olvidar que todos los interesados comparten un objetivo común: crear un producto que proporcione valor comercial adecuado y recompensas a todos los interesados. El analista de negocio suele ser la persona clave que tiene que forjar esta asociación de colaboración.

Los clientes tienen derechos y responsabilidades con los analistas de negocio y los desarrolladores durante el desarrollo de las actividades de desarrollo de requisitos. En la tabla 2, se indican estas expectativas y responsabilidades.

Tabla 2*Derechos y responsabilidades de los clientes*

Derechos	Responsabilidades
<p>Tiene derecho a</p> <ol style="list-style-type: none"> 1. Los BA hablen su idioma. 2. Los BA aprendan sobre su negocio y sus objetivos. 3. Los BA registren los requisitos de forma adecuada. 4. Recibir explicaciones prácticas y entregables de requisitos. 5. Cambiar sus requisitos. 6. Esperar un ambiente de respeto mutuo. 7. Escuchar ideas y alternativas para sus requerimientos y para su solución. 8. Describir las características que harán que el producto sea fácil de usar. 9. Escuchar formas de ajustar los requisitos para acelerar el desarrollo a través de la reutilización. 10. Recibir un sistema que satisfaga sus necesidades funcionales y expectativas de calidad 	<p>El cliente es responsable de...</p> <ol style="list-style-type: none"> 1. Educar a los BA y desarrolladores sobre su negocio. 2. Dedicar el tiempo que sea necesario para brindar y aclarar requisitos. 3. Ser específico y preciso al proporcionar información sobre los requisitos. 4. Tomar decisiones oportunas sobre los requisitos cuando se le solicite. 5. Respetar la evaluación del desarrollador sobre el costo y la viabilidad de los requisitos. 6. Establecer prioridades de requisitos realistas en colaboración con los desarrolladores. 7. Revisar requisitos y evaluar prototipos. 8. Establecer criterios de aceptación. 9. Comunicar con prontitud los cambios en los requisitos. 10. Respetar el proceso de desarrollo de requisitos.

Nota. Sucunuta, M.. 2023

Estos derechos y responsabilidades se aplican a los clientes reales cuando el *software* se desarrolla para uso corporativo interno, bajo contrato o para un conjunto conocido de clientes importantes. Para el desarrollo de productos de mercado masivo, los derechos y responsabilidades son más aplicables a los sustitutos del cliente, como el gerente de producto. Como parte de la planificación del proyecto, el cliente clave y los Interesados en el desarrollo deben revisar estas dos listas y negociar para llegar a un acuerdo. Asegúrese de que los participantes en el desarrollo de requisitos entiendan y acepten sus responsabilidades. Esta comprensión puede reducir la fricción

más adelante, cuando una de las partes espere algo que la otra no está dispuesta o no puede proporcionar.



Analice detenidamente [Clientes de software: Declaración de derechos](#), donde se describe cada uno de estos derechos y responsabilidades de los clientes frente a los BA y los desarrolladores.

2.1.2. Patrocinador

El patrocinador o Sponsor en un proyecto de software es la persona o entidad que tiene el mayor interés en el éxito del proyecto y está dispuesto a asumir la responsabilidad de asegurarse de que el proyecto tenga éxito. Puede ser un miembro de la alta gerencia de una empresa, un funcionario gubernamental o una organización sin fines de lucro, entre otros.

Muchos proyectos son desarrollados para organizaciones que son de estricta propiedad de los accionistas. Naturalmente, no se puede hablar con todos los accionistas, ni es probable que se tenga acceso a la junta directiva. En este caso es necesario nombrar un patrocinador para el proyecto que represente a los intereses de la organización (Robertson & Robertson, 2013).

El patrocinador es el que paga por el producto, por lo que es quien tiene la última palabra en cuanto a ¿Qué hace el producto?, ¿cómo lo hace?, y ¿cómo elaborar?. En otras palabras, el patrocinador es quien es el árbitro final de que producto tiene un valor óptimo. No se puede proceder sin patrocinador, si nadie está representando el interés de la organización, entonces no tiene mucho sentido proceder con el proyecto. Al arrancar el proyecto, el patrocinador probablemente debe estar en la reunión de inicio.

2.1.3. Usuario

El concepto de usuario se utiliza para referirse al individuo que finalmente se convierte en el operador real del producto. Los usuarios también son conocidos como operadores convencionales, operadores de soporte operativo y operadores de mantenimiento. Para los productos internos de una organización, los usuarios suelen ser personas que trabajan para el patrocinador del proyecto. En el caso de productos individuales, el cliente y el propietario suelen ser la misma persona.

Descubrir los usuarios es el primer paso para comprender el trabajo que realizan, debido a que los productos estarán diseñados para mejorar esta tarea. Además, es necesario comprender qué tipo de personas son, para poder proporcionar la experiencia de usuario adecuado. Se tiene que desarrollar un producto que los usuarios puedan y sobre todo quieran utilizar para el desarrollo de sus actividades. Obviamente, cuanto mejor se comprende a los usuarios, mejor tendrá la oportunidad de especificar un producto adecuado para ellos.

Los usuarios hacen diferentes demandas del producto. Por ejemplo, con respecto al requisito de usabilidad, un piloto de línea aérea tiene requisitos muy diferentes, al de un viajero comprando un billete en un sistema de transporte terrestre.

Cuando se desarrollan productos de consumo masivo, *software* de mercado o sitios web, debe considerar el uso de la persona como el usuario. Esta persona es una persona virtual que es representativo de la mayoría de los usuarios. Al determinar las características de esta persona en un grado suficiente, el equipo de requisitos puede conocer los requisitos correctos para satisfacer a cada uno de los personajes.

Desde el punto de vista ágil considerar a los usuarios potenciales es vital. Muchos equipos operan con un solo usuario que se le pide que suministre los requisitos para un producto y poca o ninguna consideración a lo que ocurrirá cuando el producto se lance a un público más amplio. Es importante que considere siempre un espectro más amplio de usuarios y como mínimo, elija usuarios interesados de ambos extremos.

Para cada tipo de usuario, escriba una sección en su especificación para describir, tan completamente como el tiempo lo permita, los atributos de sus usuarios. Considere las siguientes posibilidades:

- Experiencia en el tema: ¿Cuánta ayuda necesitan?
- Experiencia tecnológica: ¿Pueden operar el producto? ¿Qué términos técnicos se deben utilizar?
- Habilidades intelectuales: ¿Deberían simplificarse las tareas? ¿O dividido en un nivel inferior?
- Actitud hacia el trabajo: ¿Cuáles son las aspiraciones de los usuarios?
- Educación: ¿Qué puede esperar que su usuario sepa?
- Habilidades lingüísticas: no todos los usuarios hablan o leen el lenguaje nativo.

- Y lo más importante, ¿qué es lo que más desean mejorar en su trabajo?

Además, para cada categoría de usuario, identifique los atributos particulares que el producto debe satisfacer:

- Personas con discapacidades: considere todas las discapacidades. Esto, en algunos casos, es un requisito legal.
- No leídos: considere las personas que no saben leer y las personas que no hablan el lenguaje nativo.
- Personas que necesitan lentes de lectura: esto es particularmente utilizado por la mayoría de las personas.
- Personas que no pueden resistirse a cambiar cosas como fuentes, estilos, etc.
- Personas que seguramente llevarán equipaje, paquetes grandes o un bebé.
- Personas que normalmente no utilizan una computadora.
- Personas que podrían estar enojadas, frustradas, presionadas o con prisa.

Escribir todo esto parece una tarea difícil y tediosa, sin embargo, se debe tomar el tiempo necesario para registrar, para que otras personas puedan leerlo. Los usuarios son tan importantes para su causa que usted debe entender qué tipo de personas son y qué capacidades tienen.

En esta etapa, los usuarios que identifique son usuarios potenciales. Es decir, aún no conoce con precisión el alcance del producto, se determinará posteriormente en el proceso de requisitos, por lo que está identificando a las personas que podrían utilizar, mantener y dar soporte al producto. Recuerde que las personas que no sean los usuarios podrían terminar teniendo contacto directo con el producto. Es mejor identificar a los usuarios superfluos que dejar de encontrarlos todos porque cada categoría de usuario diferente tendrá algunos requisitos diferentes.

2.1.4. Otros interesados

Al desarrollar las actividades existen otras personas con las que tiene que hablar para poder encontrar todos los requisitos, su contacto con estas personas puede ser fugaz, pero es necesario. Cualquiera de ellos puede tener potenciales requisitos para el producto.

El problema que a menudo se enfrentan en los proyectos de requisitos es que no se descubre a todos los Interesados, y, por lo tanto, no se pueden descubrir sus necesidades. Esta deficiencia puede resultar en una serie de peticiones de cambio cuando el producto sea ya liberado. Naturalmente, las personas que pasaron por alto no se sentirán felices. Además, debe tener en cuenta que cuando se instala un nuevo sistema, alguien gana y alguien pierde poder. Algunas personas encuentran que el producto les aporta nuevas capacidades y algunas personas no son capaces de hacer su trabajo de la forma en que lo hacían. A continuación, en la siguiente infografía se describen algunos Interesados.

Otros interesados

2.2. Análisis de interesados

El análisis de interesados es una tarea que se encarga de la identificación de los interesados que pueden verse afectado por una propuesta de *software* o que comparten una necesidad de negocio. El análisis de interesados se realiza tan pronto como una necesidad de negocio es identificada y habitualmente será una actividad continuada mientras dure el Análisis de negocio.

El análisis de los Interesados se inicia con la identificación de los interesados que pueden ser afectados por la necesidad de negocio o la solución nueva. Los interesados pueden ser agrupados en categorías que reflejen su nivel de involucramiento o interés en el proyecto. Deben ser claramente descritos los roles, responsabilidades y autoridad sobre los requerimientos para cada interesado o grupo de interesados. Este análisis también involucra entender la influencia y actitud que tienen sobre la iniciativa, y evaluar actitudes y comportamientos positivos y negativos que pueden afectar el resultado de la iniciativa y la aceptación de la solución (IIBA, 2015).

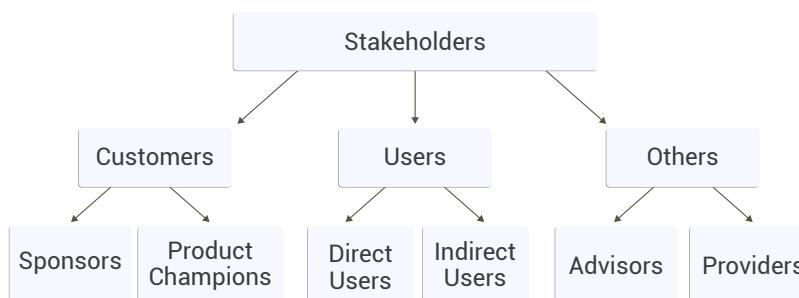
De manera general el Análisis de interesados puedes seguir los siguientes pasos:

- **Identificación:** identifica a todas las personas o grupos que pueden estar involucrados o afectados por el proyecto. Esto incluye tanto a aquellos que pueden tener un impacto positivo como negativo en el proyecto.

- **Clasificación:** clasifica a los Interesados con base en su nivel de influencia y el impacto que pueden tener en el proyecto. Por ejemplo, puedes clasificarlos como altamente influyentes, influyentes, moderadamente influyentes o poco influyentes.
- **Evaluación:** evalúa las necesidades y expectativas de cada interesado en relación con el proyecto. Considera cuáles son sus motivaciones y objetivos, así como sus puntos de vista y opiniones sobre el proyecto.
- **Priorización:** prioriza a los interesados en función de su influencia y el impacto que pueden tener en el proyecto. Esto te ayudará a determinar qué interesados deben ser objeto de más atención y recursos.
- **Comunicación:** planea cómo comunicarte con los interesados y cómo involucrarlos en el proyecto. Considera tanto la frecuencia como el contenido de la comunicación y cómo asegurarte de que los interesados estén al tanto de los avances y cambios en el proyecto.

Es importante actualizar y revisar periódicamente el análisis de interesados a medida que el proyecto evoluciona y los intereses y prioridades de los interesados cambian. Estos interesados deben ser categorizados, para lo cual existen. Tres categorías: clientes, usuarios y otros interesados. En la figura 8 se indica esta clasificación.

Figura 8
Categoría de los Interesados



Nota. Adaptado de *The Software Requirements Memory Jogger* (p. 50) por Gottesdiener, E., 2005, Goal/Opc.

Para realizar la Identificación es necesario realizar una categorización de los interesados que consiste en realizar una organización estructurada en grupos o de forma individual de quienes tienen interés o se ven influenciados en el producto que se está desarrollando. Para realizar esto, siga los siguientes pasos:

1. Identifique a los interesados como clientes, usuarios u otros interesados.

Se recomienda hacer una lista de Interesados como roles en cada una de las categorías, en lugar de personas específicas, obviamente depende de las exigencias del equipo de desarrollo (en algunos casos especialmente las estrategias ágiles si recomiendan indicar a las personas). Recuerde que algunos roles podrían aparecer en múltiples categorías.

Tome como referencia un listado de roles de interesados genéricos como punto de partida para categorizar sus interesados, luego traduzca a los roles específicos del proyecto, por ejemplo, el rol genérico “Experto financiero” para un proyecto de contabilidad podría ser “Asesor fiscal” (esto depende exclusivamente de lo que especifique en el contexto del negocio). Finalmente, debe categorizarlo al interesado.

2. Revise el listado de categorías de Interesados con los interesados del proyecto para asegurarse que la lista está completa y actualizada.
3. Revisar la lista según sea necesario y comparta con todo el equipo.

En la tabla 3, se indica la categoría de los interesados para el sistema “Syscargo”, para el área de *marketing*.

Tabla 3*Categoría de los Interesados*

Clientes		Usuarios		Otros Interesados	
Patrocinador	Product Champion	Directo	Indirecto	Asesores	Proveedores
<ul style="list-style-type: none"> • Gerente general 	<ul style="list-style-type: none"> • Gerente de marketing 	<ul style="list-style-type: none"> • Asistente de mercadeo • Asistente de marketing • Publicista • Gestor de contenido • Redactor • Especialista en TI 	<ul style="list-style-type: none"> • Gerente general • Gerente de marketing • Diseñador 	<ul style="list-style-type: none"> • Consultor externo 	<ul style="list-style-type: none"> • Gerente de proyecto • Administrador de la base de datos • Desarrolladores

Nota. Sucunuta, M., 2023

Otra de las actividades es definir el perfil de los Interesados, que consiste en una descripción que caracteriza a cada Interesado y explica su relación con el proyecto. Esto es necesario porque permite:

- Comprender los intereses, inquietudes y los criterios de éxito del producto.
- Descubrir potenciales fuentes de conflicto de requisitos entre los interesados.
- Resaltar los temas de requisitos que pueden necesitar tiempo y atención adicional.

El perfil de los Interesados puede revelar potenciales obstáculos para la implementación exitosa del producto y ayudan a definir cuánta participación debe tener cada interesado en la obtención de requisitos.

Para definir el perfil de los interesados es necesario seguir los siguientes pasos:

1. Escribir un perfil básico para cada interesado, con la siguiente información:
 - **Rol:** listar la categoría del interesado (por ejemplo, patrocinador, Product Champion, usuario directo, usuario indirecto, asesor o proveedor) al que pertenece.

- **Responsabilidades:** describa brevemente el rol de cada interesado en relación con el proyecto.
 - **Intereses:** liste las necesidades, los interesados, deseos y expectativas para el producto.
 - **Los criterios de éxito:** describir las características o capacidades que el producto debe tener para ser considerado como exitoso.
 - **Preocupaciones:** listar todos los obstáculos, limitaciones, o factores limitantes que puedan impedir o inhibir al interesado a aceptar el producto.
 - **Capacidad técnica:** describir al usuario directo el grado de familiaridad con la tecnología.
 - **Características del entorno de trabajo y limitaciones:** describir las condiciones de trabajo relevante que pueda afectar el uso del sistema (por ejemplo, un entorno de trabajo ruidoso o el uso del móvil o al aire libre).
2. Incluir los perfiles de los interesados en el documento de requisitos de usuario (si se utiliza) y en el documento de especificaciones de requisitos de software. Si los perfiles contienen gran cantidad de información, documente un perfil por cada interesado en una tabla o sección en el documento de requisitos correspondientes.

Tabla 4

Perfil de los Interesados

Interesado	Rol	Responsabilidad	Intereses	Criterios de éxito	Preocupaciones	Competencias técnicas
Gerente general	-Patrocinador -Usuario indirecto	-Aprobar el proyecto	-	-	-	-
	-	-	-	-	-	-
	-	-	-	-	-	-
	-	-	-	-	-	-
	-	-	-	-	-	-

Nota. Sucunuta, M., 2023



2.3. Modelado de negocio

El propósito del modelado de negocio no es solo desarrollar aplicaciones de software para una empresa, sino que es una actividad esencial para comprender la evolución de una empresa, con el desarrollo de actividades que van desde el diseño organizacional hasta el diseño de sistemas. Desde el punto de vista del desarrollo de sistemas nos centraremos en el conocimiento de los procesos de la organización para entender su funcionamiento.

Un sistema software en la empresa no opera de forma aislada, es necesario conocer el entorno donde funcionará, no solamente a nivel tecnológico, sino la organización completa donde se ubicará. El objetivo es definir el modelo de negocio o, al menos, el modelo del contexto. Desde el enfoque de desarrollo de software lo que se busca con el modelado de negocio es:

- Comprender la estructura y dinámica existente en la organización.
- Asegurarse de que todos los involucrados en el proyecto tienen la misma visión de la organización.
- Comprender cómo desarrollar el nuevo sistema para aumentar la productividad y cuáles de los sistemas ya existentes se verán afectados.

A continuación, se explican diversos enfoques para el modelado de negocio.

2.3.1. Modelado de dominio

La idea de este enfoque es poder comunicarse eficazmente con los clientes y usuarios, comprender su negocio, entender sus necesidades y proponer una solución adecuada. Para lograr todo esto es necesario conocer un conjunto de conceptos interrelacionados. Por ejemplo, para el ámbito académico, el conjunto de conceptos relacionados sería: aspirante, estudiante, profesor, admisión, becas, promoción, acreditación, plan académico, malla curricular, entre otros.

Los términos del dominio, incluidos en un glosario, se pueden gestionar a través de un sinnúmero de herramientas, que van desde una wiki a una ontología, o desde un lenguaje natural hasta un lenguaje de modelado.

2.3.2. Modelado de procesos de negocio

Describe cada proceso de negocio, y especifica sus datos, actividades, roles y reglas de negocio. Esta actividad, generalmente, ejecutan los directivos y gestores de la empresa con el objeto de mejorar la eficiencia de los procesos y la calidad. Las mejoras no siempre van acompañadas del desarrollo de un sistema, pero en los casos que sí es necesario, disponer de un modelo de negocio facilitará los procesos relacionados con el desarrollo de software y, en especial, con las tareas relacionadas con la ingeniería de requisitos. Actualmente, existen varios métodos y técnicas para construir modelos de negocio, entre los que encontramos el Lenguaje Unificado de Modelado y la notación Business Process Modeling. El [Object Management Group](#) (OMG), entre una de sus alternativas, tiene un lenguaje estándar de modelado de procesos de negocio denominado BPMN.

Para ampliar el conocimiento del modelado de procesos y tener una idea más concreta, lo invito a revisar el tema [Business Process Model y Notation \(BPMN\)](#), que se encuentra en el sitio de la OMG. Esta página presenta una breve descripción de lo que es el modelo y notación de los procesos de negocio, y luego el acceso a importantes recursos como es el estándar ISO/IEC 19510.

2.3.3. Modelo basado en objetivos

Los objetivos son metas que el sistema debe conseguir a través de la cooperación e interacción entre actores, sistemas tecnológicos y el entorno. Este enfoque utiliza una aproximación de descomposición funcional, partiendo la especificación en objetivos más pequeños y alcanzables. Existen varias notaciones para los modelos basados en objetivos, entre las más conocidas tenemos: diagrama de flujo, mapa de ecosistemas y árbol de características.

2.4. Requisitos de negocio

Los “requisitos de negocio” definen la ruta estratégica de un proyecto, describen una necesidad que conduce al desarrollo uno o más proyectos

que permitan establecer una solución y los resultados de negocio deseados. Las oportunidades de negocio, los objetivos de negocio, las métricas de éxito y la declaración de visión conforman en conjunto los requisitos de negocio.

Los problemas que estén asociados a los requisitos de negocio deben resolverse antes de la especificación final de los requisitos funcionales y no funcionales. Una declaración del alcance y las limitaciones del proyecto es de gran ayuda con las discusiones sobre las características propuestas y las versiones que se desarrollarán. Los requisitos de negocio proporcionan una referencia para tomar decisiones sobre los cambios y mejoras de los requisitos propuestos. Es recomendable mostrar los objetivos de negocio, la visión y los puntos destacados del alcance en cada sesión de obtención de requisitos para que el equipo pueda juzgar rápidamente si un requisito propuesto está dentro o fuera del alcance.

2.4.1. Identificar los beneficios de negocio deseados

Los requisitos de negocio establecen el contexto y permiten la medición de los beneficios que la empresa espera lograr al emprender un proyecto de software. Las organizaciones no deben iniciar ningún proyecto sin una comprensión clara del valor que agregará al negocio. Establezca objetivos medibles con objetivos de negocio y luego defina métricas de éxito que le permitan medir si está en camino de cumplir esos objetivos.

Los requisitos de negocio provienen comúnmente de patrocinadores de fondos, ejecutivos corporativos, gerentes de marketing o visionarios de productos. Sin embargo, puede ser un desafío identificar y comunicar los beneficios de negocio. Los miembros del equipo a veces no están exactamente seguros de lo que se pretende lograr con el proyecto. A veces, los patrocinadores no quieren establecer objetivos de manera medible y luego ser responsables de alcanzarlos. Podría haber múltiples interesados importantes que no estén de acuerdo sobre cuáles deberían ser los objetivos. El analista de negocio puede garantizar que los interesados correctos establezcan los requisitos de negocio y faciliten la obtención, la priorización y la resolución de conflictos. Karl Wiegers (2006) sugiere algunas preguntas que el BA puede hacer para ayudar a obtener los requisitos de negocio.

El beneficio de negocio tiene que representar un valor real para los patrocinadores del proyecto y para los clientes del producto. Por ejemplo, la

simple fusión de dos sistemas en uno no es un objetivo comercial razonable. A los clientes no les importa si están usando una aplicación que involucra 1, 5 o 10 sistemas. Lo que les preocupa son cuestiones como el aumento de los ingresos y la disminución de los costos. La fusión de dos sistemas puede ser parte de la solución, pero rara vez es el verdadero objetivo de negocio. Los proyectos de cumplimiento normativo y legal también tienen objetivos de negocio claros. A menudo, los objetivos se expresan como la evitación de riesgos, posiblemente para evitar ser demandado o quedar fuera del negocio.

2.4.2. Visión del producto y alcance del proyecto

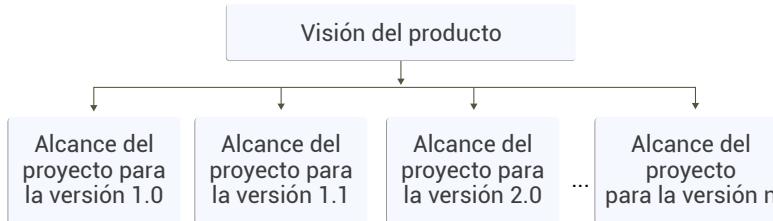
Dos elementos centrales de los requisitos de negocio son la visión y el alcance. La visión del producto describe sucintamente el producto final que logrará los objetivos de negocio. Este producto podría servir como la solución completa para los requisitos de negocio o solo como una parte de la solución.

La visión describe de qué se trata el producto y en qué podría convertirse finalmente. Proporciona el contexto para tomar decisiones a lo largo de la vida del producto y alinea a todos los interesados en una dirección común. El alcance del proyecto identifica qué parte de la visión final del producto abordará el proyecto actual o la iteración de desarrollo. La declaración de alcance traza el límite entre lo que está dentro y lo que está fuera de este proyecto.

La visión se aplica al producto como un todo. La visión cambia con relativa lentitud a medida que el posicionamiento estratégico de un producto o los objetivos de negocio de una empresa evolucionan con el tiempo. El alcance pertenece a un proyecto específico o iteración que implementará el próximo incremento de la funcionalidad del producto, como se muestra en la figura 9.

Figura 9

Visión del producto y alcance del proyecto.



Nota. Adaptado de *Software Requirements* (p. 79), por Wiegers, K. y Beatty, J., 2013. Microsoft Press.

Como se puede apreciar, el alcance es más dinámico que la visión porque los interesados ajustan el contenido de cada publicación dentro de sus limitaciones de programación, presupuesto, recursos y calidad. El alcance de la versión actual debe ser claro, pero el alcance de futuras versiones será más borroso cuanto más lejos se mire. El objetivo del equipo es administrar el alcance de un proyecto específico de desarrollo o mejora como un subconjunto definido de la visión estratégica del producto.

En Gottesdiener (2005) se indica una guía para definir el visionamiento, mediante los siguientes pasos:

1. Definir lo siguiente:

- **Clientes objetivo:** describa la persona que utilizará o comprará el software.
- **Declare la necesidad u oportunidad:** describa lo que hace el cliente objetivo y explique cómo este producto lo ayudará a hacerlo.
- **Nombre del producto:** indique el nombre del producto que desarrollará.
- **Categoría del producto:** describa el tipo de producto que construirá. La categoría del producto podría incluir una

aplicación de software de negocio interno, software embebido, software de juegos, dispositivo hardware o un sistema complejo.

- **Beneficio clave o razón convincente para comprar:** describa lo que el producto podrá hacer para el cliente objetivo o el justificativo para comprar el producto.
 - **Principal alternativa competitiva, sistema actual, o proceso manual actual:** describa la competencia clave del producto, sistema o proceso que el producto reemplazará.
 - **Declaración de la diferencia de los productos primarios:** explique las diferencias entre el producto que está desarrollando y la competencia.
2. Crear la declaración de la visión mediante la introducción de los términos definidos en la siguiente plantilla.
- Para** <Cliente objetivo> **quien** <declaración de la necesidad u oportunidad>, **el** <nombre del producto> **es un** < categoría del producto> **que** <principales beneficios o razones convincentes para comprar>.
- A diferencia de** <principal alternativa competitiva, sistema actual, o proceso manual actual>, **nuestro producto** <declaración de las diferencias del producto primario>.

Ejemplo:

 Para empresas de servicios y su personal **quienes** proporcionan servicios de limpieza de ventanas para el hogar y sitios comerciales, **el** sistema de limpieza **es una** aplicación software basado en web **que** estima y programa trabajos, asigna personal a los trabajos, promueve los servicios de la empresa, y retiene a los clientes actuales. **A diferencia del** producto actual que no permite a múltiples compañías colaborar en las ofertas u optimizar la asignación del personal a los trabajos, **nuestro producto** permitirá a múltiples compañías usar la aplicación, proporcionar todo el ciclo de vida de los servicios de negocio para toda la operación (incluye cuentas por pagar y cuentas por cobrar) y es fácil de usar.

3. Revisar la declaración de la visión y verificar que se alinea con las metas y objetivos de la organización.
 - Haga que el patrocinador se asegure de que la visión encaje con las metas y objetivos departamentales y organizacionales.
 - Haga que los miembros del equipo, idealmente en colaboración con el patrocinador del proyecto, revisen y modifiquen la declaración de visión según sea necesario.

La declaración del problema describe un problema actual que el negocio está experimentando y aclara cómo sería una solución exitosa. La declaración del problema es útil cuando la solución involucra mejoras al software existente o cuando la implementación del producto crea la necesidad de cambio en el proceso de negocio. Puede utilizar la siguiente plantilla para realizar una declaración del problema.

El problema de <Insertar el planteamiento del problema> **afecta** <nombre de las personas afectadas, organización, o grupo de clientes>. **El impacto de esto es** <nombre del impacto (por ejemplo, malas decisiones, los sobrecostos, información o procesos erróneos, tiempo de respuesta lento a los clientes, etc.)>. **Una solución exitosa sería** <describir la solución>.

Ejemplo:

El problema de cotización y programación de trabajos y el pago a los contratistas utilizando el proceso actual manual y automatizado **afecta** a clientes, contratistas, programadores y contadores. **El impacto de esto son** estimaciones incorrectas, doble reserva de contratistas, espacios vacíos en nuestro horario de trabajo y pago excesivo o insuficiente de contratos. **Una solución exitosa sería** que permita respuestas inmediatas a las solicitudes de cotización en el código postal del cliente, brinde la capacidad de programar y completar trabajos dentro de una semana a la solicitud realizada, permita la facturación rápida del cliente y emita pagos semanales al contratista.



2.4.3. Requisitos de negocio en conflicto

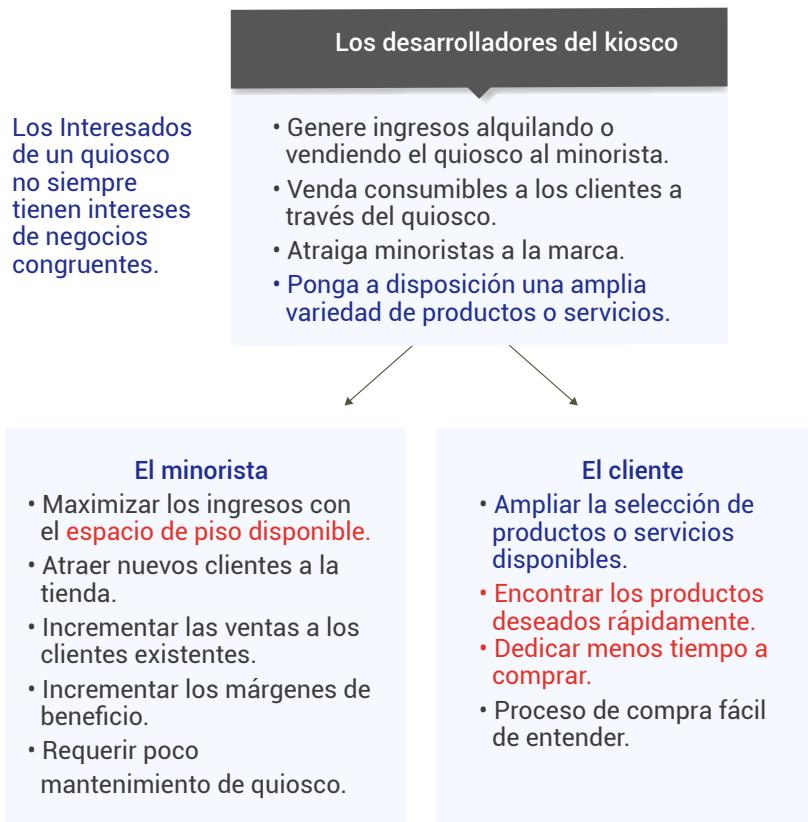
Los requisitos de negocio recopilados de varias fuentes pueden entrar en conflicto. Considere un quiosco que será utilizado por los clientes de una

tienda minorista. La figura 10 muestra los intereses de negocio probables del desarrollador, minorista y cliente del quiosco, ya que imaginamos cómo cada uno de estos interesados espera que el quiosco proporcione una ventaja sobre su forma actual de hacer negocios.

Los objetivos de los distintos interesados a veces están alineados. Por ejemplo, tanto los desarrolladores del quiosco como los clientes quieren tener una amplia variedad de productos o servicios disponibles a través del quiosco. Sin embargo, algunos objetivos de negocio podrían entrar en conflicto. El cliente quiere pasar menos tiempo comprando bienes y servicios, pero el minorista preferiría que los clientes permanezcan en la tienda y gasten más dinero. La tensión entre las partes interesadas con diferentes objetivos y limitaciones puede dar lugar a requisitos empresariales en conflicto. Los tomadores de decisiones del proyecto deben resolver estos conflictos antes de que el analista pueda detallar los requisitos del quiosco. El enfoque debe estar en entregar el máximo valor comercial a las principales partes interesadas. Es fácil distraerse con las características superficiales del producto que en realidad no abordan los objetivos de negocio.

Figura 10

Intereses de negocio de diferentes Interesados



Nota: Adaptado de *Software Requirements* (p. 80), por Wiegers, K. y Beatty, J., 2013. Microsoft Press.

Quienes toman las decisiones del proyecto no deben esperar que el equipo de software resuelva los conflictos entre los distintos interesados. A medida que se incorporen más grupos con intereses diversos, el alcance crecerá. El aumento del alcance descontrolado, en el que los interesados sobrecargan el nuevo sistema en un intento de satisfacer todos los intereses, puede hacer que el proyecto se derrumbe por su propio peso. Un BA puede ayudar sacando a la luz posibles áreas de conflicto y diferentes suposiciones, marcando objetivos de negocio en conflicto, señalando cuándo las características solicitadas no logran esos objetivos y facilitando

la resolución de conflictos. Resolver tales problemas es a menudo una lucha política y de poder.

Los proyectos de larga duración a menudo experimentan un cambio en los tomadores de decisiones a mitad de camino. Si esto le sucede, revise de inmediato los requisitos de negocio de referencia con los nuevos tomadores de decisiones. Deben conocer los requisitos de negocio existentes, que es posible que deseen modificar. Si es así, el gerente del proyecto tendrá que ajustar los presupuestos, los cronogramas y los recursos, mientras que el BA podría necesitar trabajar con los interesados para actualizar los requisitos funcionales y de los usuarios y restablecer sus prioridades.

2.5. Documento de visión y alcance

El documento de visión y alcance recopila los requisitos de negocio en un solo entregable que prepara el escenario para el trabajo de desarrollo posterior. Algunas organizaciones crean un acta de constitución del proyecto o un documento de caso de negocios que tiene un propósito similar. Las organizaciones que crean software comercial a menudo crean un documento de requisitos de mercado (o *marketing*) (MRD).

El propietario del documento de visión y alcance puede ser el patrocinador, alguna autoridad de financiamiento o alguien que desempeñe algún cargo similar. Un analista de negocio puede trabajar con estas personas para articular los requisitos de negocio y redacta el documento de visión y alcance. Los aportes a los requisitos de negocio deben provenir de personas que tengan una idea clara de por qué están llevando a cabo el proyecto. Estas personas pueden incluir la alta gerencia del cliente o de la organización de desarrollo, un visionario del producto, un gerente de producto, un experto en la materia o miembros del departamento de *marketing*.

Para tener una idea más clara de cómo desarrollar el documento de visionamiento para un proyecto de software, revise los siguientes recursos:

1. [Documento de visión de IBM](#).
2. [Documento de visión – Sistema para gestión de proyectos Scratch](#).
3. [Documento de visión y alcance – Proyecto Minjusticia](#).

En el primer caso, se explica la plantilla para documentar el visionamiento, se indica lo que en cada literal deberá describir. Es importante que revise detenidamente cada uno de los literales y determine lo que debe considerar para determinado proyecto. Es importante aclarar que esta actividad no se trata solamente de llenar una plantilla, sino que, como analista, debe planificar y desarrollar un conjunto de actividades, especialmente, de obtención que permitan disponer de la información adecuada. La plantilla es muy útil para documentar lo descubierto y evidentemente un artefacto que forma parte del proceso de especificación de requisitos.

Luego el segundo y tercer recurso se muestran dos ejemplos con distintos niveles de descripción, debido a la naturaleza y necesidades del proyecto. Es importante que analice detenidamente cada uno de estos casos, ya que le permitirá conocer la necesidad de elaborar un documento con un nivel adecuado de detalle.

Figura 11

Plantilla para documentar el visionamiento del producto

- 1. Requisitos de negocio**
 - 1.1. Antecedentes
 - 1.2. Oportunidad de negocio
 - 1.3. Objetivos de negocio
 - 1.4. Métricas de satisfacción
 - 1.5. Declaración de la visión
 - 1.6. Riesgos de negocio
 - 1.7. Supuestos y dependencias de negocio
- 2. Alcance y limitaciones**
 - 2.1. Características principales
 - 2.2. Alcance de la versión inicial
 - 2.3. Alcance de lanzamientos posteriores
 - 2.4. Limitaciones y exclusiones
- 3. Contexto del negocio**
 - 3.1. Perfil de los Interesados
 - 3.2. Prioridades del proyecto
 - 3.3. Consideraciones de despliegue

Nota. Adaptado de *Software Requirements* (p. 82), por Wiegers, K. y Beatty, J., 2013. Microsoft Press.

La figura 11 sugiere una plantilla para el documento de visión y alcance, como con cualquier plantilla, es necesario adaptar para satisfacer las necesidades específicas de sus propios proyectos. Si ya registró parte de esta información en otro lugar, no la duplique en el documento de visión y alcance. Algunos elementos del documento de visión y alcance pueden ser reutilizables de un proyecto a otro, como los objetivos de negocio, los riesgos de negocio y los perfiles de los interesados. A continuación se describe cada una de estas partes.

1. Requisitos de negocio

Los proyectos son lanzados con la convicción de que se creará o cambiará un producto proporcionando beneficios valiosos para alguien y un retorno de la inversión adecuado. Los requisitos de negocio describen los principales beneficios que el nuevo sistema proporcionará a sus patrocinadores, compradores y usuarios. Los requisitos de negocio influyen directamente en qué requisitos de usuario implementar y en qué secuencia.

1.1. Antecedentes

Resuma la razón fundamental y el contexto del nuevo producto o los cambios que se realizarán en uno existente. Describa la historia o situación que llevó a la decisión de construir este producto. El enfoque realice con base en el problema que identifique y sobre el que se fundamenta el proyecto.

1.2. Oportunidad de negocio

Para un sistema de información corporativo, describa el problema de negocio que se está resolviendo o el proceso que se está mejorando, así como el entorno en el que se utilizará el sistema. Para un producto comercial, describa la oportunidad comercial que existe y el mercado en el que competirá el producto. Esta sección podría incluir una evaluación comparativa de productos existentes, indicando por qué el producto propuesto es atractivo y las ventajas que ofrece. Describa los problemas que actualmente no se pueden resolver sin la solución prevista. Muestre cómo se alinea con las tendencias del mercado, la evolución de la tecnología o las direcciones estratégicas corporativas. Enumere cualquier otra tecnología, proceso o recurso necesarios para proporcionar una solución completa para el cliente.

Describa las necesidades de los clientes típicos o del mercado objetivo. Presente los problemas del cliente que abordará el nuevo producto. Proporcione ejemplos de cómo los clientes usarían el producto. Defina cualquier interfaz crítica conocida o requisitos de calidad, pero omita los detalles de diseño o implementación.

1.3. Objetivos de negocio

Resuma los beneficios de negocio importantes que proporcionará el producto de forma cuantitativa y medible. Los tópicos (“ser reconocido como un <lo que sea> de clase mundial”) y las mejoras vagamente declaradas (“proporcionar una experiencia más gratificante al cliente”) no son útiles ni verificables.

Las organizaciones generalmente emprenden un proyecto para resolver un problema o aprovechar una oportunidad. El objetivo de un modelo de negocio es mostrar una jerarquía de problemas de negocio relacionados y objetivos de negocio medibles (Chen & Beatty, 2012). Los problemas describen qué es lo que impide que la empresa cumpla sus metas en la actualidad, mientras que los objetivos definen formas de medir el logro de esas metas. Los problemas y los objetivos están entrelazados: comprender uno puede revelar el otro.

Dado un conjunto de objetivos de negocio, pregunte: “¿Qué nos impide alcanzar la meta?”, para identificar un problema de negocio más detallado. O trabaje hacia atrás preguntando: “¿Por qué nos preocupa ese objetivo?”, para comprender mejor el problema o la oportunidad de negocio de alto nivel. Ante un problema de negocio, pregunte: “¿Cómo evaluaremos si el problema está resuelto?”, para identificar el objetivo medible. El proceso es iterativo, recorriendo la jerarquía de problemas y objetivos hasta que vea que surge una lista de características que ayudarían a resolver los problemas y alcanzar los objetivos.

1.4. Métricas de éxito

Especifique los indicadores que los Interesados utilizaran para definir y medir el éxito de este proyecto. Identifique los factores que tienen el mayor impacto en el logro de ese éxito, incluidos los factores tanto dentro como fuera del control de la organización.

A veces, los objetivos de negocio no se pueden medir hasta mucho después de que se completa un proyecto. En otros casos, el logro de los objetivos de negocio puede depender de proyectos más allá del actual. Sin embargo, sigue siendo importante evaluar el éxito de un proyecto individual. Las métricas de éxito indican si un proyecto está bien encaminado para alcanzar sus objetivos de negocio. Las métricas se pueden rastrear durante las pruebas o poco después del lanzamiento del producto.

1.5. Declaración de visión

Escriba una declaración de visión concisa que resuma el propósito y la intención a largo plazo del producto. La declaración de visión debe reflejar una visión equilibrada que satisfaga las expectativas de los diversos interesados. Puede ser algo idealista, pero debe basarse en las realidades de los mercados existentes o anticipados, las arquitecturas empresariales, las direcciones estratégicas corporativas y las limitaciones de recursos. Refiérase al literal 2.4.2 Visión del producto y alcance del sistema, para documentar con base en la plantilla que se indica.

1.6. Riesgos de negocio

Resuma los principales riesgos de negocio asociados con el desarrollo o no desarrollo de este producto. Las categorías de riesgo incluyen competencia en el mercado, problemas de tiempo, aceptación del usuario, problemas de implementación y posibles impactos negativos en el negocio. Los riesgos comerciales no son los mismos que los riesgos del proyecto, que a menudo incluyen problemas de disponibilidad de recursos y factores tecnológicos. Estime la pérdida potencial de cada riesgo, la probabilidad de que ocurra y cualquier acción de mitigación potencial.

1.7. Supuestos y dependencias de negocio

Una suposición es una declaración que se cree que es verdadera en ausencia de prueba o conocimiento definitivo. Los supuestos de negocio están específicamente relacionados con los requisitos de negocio. Las suposiciones incorrectas pueden impedirle alcanzar sus objetivos de negocio. Por ejemplo, un patrocinador ejecutivo puede establecer como objetivo de negocio que un nuevo sitio web aumente

los ingresos en \$ 100,000 por mes. Para establecer este objetivo de ingresos, el patrocinador hizo algunas suposiciones, tal vez que el nuevo sitioatraiga a 200 visitantes únicos adicionales por día y que cada visitante gastará un promedio de \$ 17. Si el nuevo sitio no atrae suficientes visitantes con un promedio de venta por visitante lo suficientemente alto, es posible que el proyecto no logre su objetivo de negocio. Si se entera de que ciertas suposiciones son incorrectas, es posible que deba cambiar el alcance, ajustar el cronograma o lanzar otros proyectos para lograr los objetivos.

Registre las suposiciones que hicieron los interesados al concebir el proyecto y redactar su documento de visión y alcance. A menudo, las suposiciones de una de las partes no son compartidas por otras. Si los anota y los revisa, puede evitar posibles confusiones y agravamientos en el futuro.

Registre cualquier dependencia importante que tenga el proyecto de factores externos. Algunos ejemplos son los estándares de la industria pendientes o las regulaciones gubernamentales, los entregables de otros proyectos, los proveedores externos o los socios de desarrollo. Algunas suposiciones y dependencias comerciales pueden convertirse en riesgos que el gerente de proyecto debe monitorear regularmente. Las dependencias rotas son una fuente común de retrasos en los proyectos. Tenga en cuenta el impacto de una suposición que no es cierta, o el impacto de una dependencia rota, para ayudar a las partes interesadas a comprender por qué es fundamental.

2. Alcance y limitaciones

Muchos proyectos sufren un aumento del alcance: un crecimiento desenfrenado a medida que se incorporan más y más funciones al producto. El primer paso para controlar el avance del alcance es definir el alcance del proyecto. El alcance describe el concepto y el ámbito de la solución propuesta. Las limitaciones detallan ciertas capacidades que el producto no incluirá y que algunas personas podrían asumir que estarán allí. El alcance y las limitaciones ayudan a establecer expectativas realistas por parte de los interesados, ya que los clientes a veces solicitan características que son demasiado caras o que se encuentran fuera del alcance previsto del proyecto.

El alcance se puede representar de muchas formas. En el nivel más alto, el alcance se define cuando el cliente decide a qué objetivos de negocio apuntar. En un nivel inferior, el alcance se define a nivel de características, historias de usuario, casos de uso o eventos y respuestas a incluir. En última instancia, el alcance se define a través del conjunto de requisitos funcionales planificados para su implementación en una versión o iteración específica. En cada nivel, el alcance debe permanecer dentro de los límites del nivel superior. Por ejemplo, los requisitos del usuario dentro del alcance deben correlacionarse con los objetivos de negocio, y los requisitos funcionales deben correlacionarse con los requisitos del usuario que están dentro del alcance.

2.1. Características principales

Enumere las principales características o capacidades del usuario del producto, enfatizando aquellas que lo distinguen de productos anteriores o de la competencia. Piense en cómo los usuarios utilizarán las funciones para asegurarse de que la lista esté completa y de que no incluya funciones innecesarias que suenen interesantes, pero que no aporten valor al cliente. Asigne a cada característica una etiqueta única y persistente para permitir su seguimiento a otros elementos del sistema. Puede incluir un diagrama de árbol de características, como se describe más adelante en este capítulo.

2.2. Alcance de la versión inicial

Resuma las capacidades que se planea incluir en el lanzamiento inicial del producto. El alcance a menudo se define en términos de características, pero también puede definir el alcance en términos de historias de usuario, casos de uso, flujos de casos de uso o eventos externos. También describa las características de calidad que permitirán que el producto proporcione los beneficios previstos a sus diversas clases de usuarios. Para enfocar el esfuerzo de desarrollo y mantener un cronograma de proyecto razonable, evite la tentación de incluir todas las características que cualquier cliente potencial eventualmente desee en la versión 1.0. Concéntrese en las funciones que proporcionarán el mayor valor, al costo más aceptable, a la comunidad más amplia, en el período de tiempo más apropiado.

2.3. Alcance de lanzamientos posteriores

Si imagina una evolución por etapas del producto, o si está siguiendo un ciclo de vida iterativo e incremental, elabore una hoja de ruta de lanzamiento que indique qué fragmentos de funcionalidad se aplazarán y el momento deseado de lanzamientos posteriores.

Las versiones posteriores le permiten implementar casos de uso y características adicionales, así como también enriquecer las capacidades de las iniciales. Cuanto más lejos mire, más confusas serán estas declaraciones de alcance futuras y más cambiarán con el tiempo. Espere cambiar la funcionalidad de una versión planificada a otra y agregar capacidades imprevistas. Los ciclos de publicación cortos brindan oportunidades frecuentes de aprendizaje en función de la retroalimentación del cliente.

2.4. Limitaciones y exclusiones

Enumere las capacidades o características del producto que el interesado podría esperar, pero que no está planificada para su inclusión en el producto o en una versión específica. Enumere los elementos que se eliminaron del alcance, para no olvidar la decisión del alcance. Tal vez un usuario solicitó poder acceder al sistema desde su teléfono mientras estaba lejos de su escritorio, pero se consideró que esto estaba fuera de alcance. Indique explícitamente en esta sección: “El nuevo sistema no proporcionará soporte para plataformas móviles”.

3. Contexto de negocio

Esta sección presenta los perfiles de las principales categorías de los interesados, las prioridades de la administración para el proyecto y un resumen de algunos factores a considerar al planificar la implementación de la solución.

3.1. Perfil de los interesados

Los perfiles de los Interesados describen diferentes categorías de clientes y otros interesados clave para el proyecto. No es necesario describir todos los grupos de interesados, como el personal legal que debe verificar el cumplimiento de las leyes pertinentes en un proyecto de desarrollo de sitios web. Concéntrese en diferentes tipos

de clientes, segmentos de mercado objetivo y las diversas clases de usuarios dentro de esos segmentos. Cada perfil de los Interesados debe incluir información que se indica en el literal 2.2 Análisis de interesados.

3.2. Prioridades del proyecto

Para permitir una toma de decisiones efectiva, los Interesados deben estar de acuerdo con las prioridades del proyecto. Una forma de abordar esto es considerar las cinco dimensiones: características, calidad, cronograma, costo y personal. Cada dimensión encaja en una de las siguientes tres categorías en cualquier proyecto:

- Restricción: un factor limitante dentro del cual el gerente de proyecto debe operar.
- Conductor: un objetivo de éxito significativo con una flexibilidad de ajuste limitada.
- Grado de libertad: un cierto factor de libertad que el director del proyecto tiene para ajustarse y equilibrarse con otras dimensiones.

El desafío del gerente de proyecto es ajustar los grados de libertad para lograr que los impulsores del éxito del proyecto actúen dentro de los límites impuestos por las restricciones. Suponga que el *marketing* exige de repente que se lance el producto un mes antes de lo programado. ¿Cómo respondes?

- ¿Aplazar ciertos requisitos para una versión posterior?
- ¿Acortar el ciclo de prueba del sistema planificado?
- ¿Exigir horas extra a su personal o contratar contratistas para acelerar el desarrollo?
- ¿Desplazar recursos de otros proyectos para que ayuden?

Las prioridades del proyecto guían las acciones que se debe considerar cuando surgen tales eventualidades. De manera realista, cuando se produce un cambio, es necesario conversar con los interesados clave para determinar las acciones más adecuadas a tomar en función del cambio solicitado. Por ejemplo, es posible que el *marketing* desee agregar funciones o acortar una línea de tiempo, pero tal vez estén dispuestos a diferir ciertas funciones a cambio.

Considere que no todas las cinco dimensiones pueden ser restricciones y no todas pueden ser factores impulsores. El director de proyecto necesita algunos grados de libertad para poder responder de forma adecuada cuando cambian los requisitos o las realidades del proyecto.

3.3. Consideraciones de despliegue

Resuma la información y las actividades necesarias para garantizar una implementación eficaz de la solución en su entorno operativo. Describa el acceso que los usuarios necesitarán para usar el sistema, por ejemplo, si los usuarios están distribuidos en múltiples zonas horarias o ubicados cerca unos de otros. Indique cuándo los usuarios en varias ubicaciones necesitan acceder al sistema. Si los cambios en la infraestructura son necesarios para apoyar las necesidades de software para capacidad, acceso a la red, almacenamiento de datos o migración de datos, describa esos cambios. Registre cualquier información que necesitarán las personas que prepararán la capacitación o modificarán los procesos de negocio junto con la implementación de la nueva solución.

Esta plantilla (figura 11), es otra de las a recomendaciones que se pueden utilizar para documentar el visionamiento y alcance. Es precisamente en el equipo de desarrollo donde se debe ajustar estas plantillas para realizar una documentación acorde al proyecto y sobre todo modo de trabajo del equipo de desarrollo.

En el [anexo 2 Documento de visión y Alcance](#), se establece el ejemplo del documento de visión y alcance con esta plantilla (figura 11), para el caso de Sistema de encomiendas.

2.6. Técnicas para representar el alcance

Existe diferentes modelos que se pueden utilizar para representar el alcance del proyecto de varias maneras. Considere cuáles son las que proporcionan la visión más útil para cada proyecto. Los modelos pueden incluirse en el documento de visión y alcance o almacenarse en otro lugar y referenciarse según sea necesario.

El propósito de herramientas como el diagrama de contexto, el mapa de ecosistema, el árbol de características y la lista de eventos es fomentar una comunicación clara y precisa entre los interesados del proyecto. Esa claridad es más importante que adherirse dogmáticamente a las reglas para un diagrama “correcto”. Sin embargo, se recomienda que adopte las normas ilustradas en los siguientes ejemplos como normas para dibujar los diagramas. Por ejemplo, en un diagrama de contexto, supongamos que utiliza un triángulo para representar el sistema en lugar de un círculo, y óvalos en lugar de rectángulos para entidades externas. Sus colegas tendrían dificultad para leer un diagrama que sigue sus preferencias personales en lugar de un estándar de equipo.

Los diagramas de contexto, mapas de ecosistemas, árboles de características y listas de eventos son las formas más comunes de representar visualmente el ámbito. Sin embargo, también se utilizan otras técnicas. A continuación, se detallan estas técnicas.

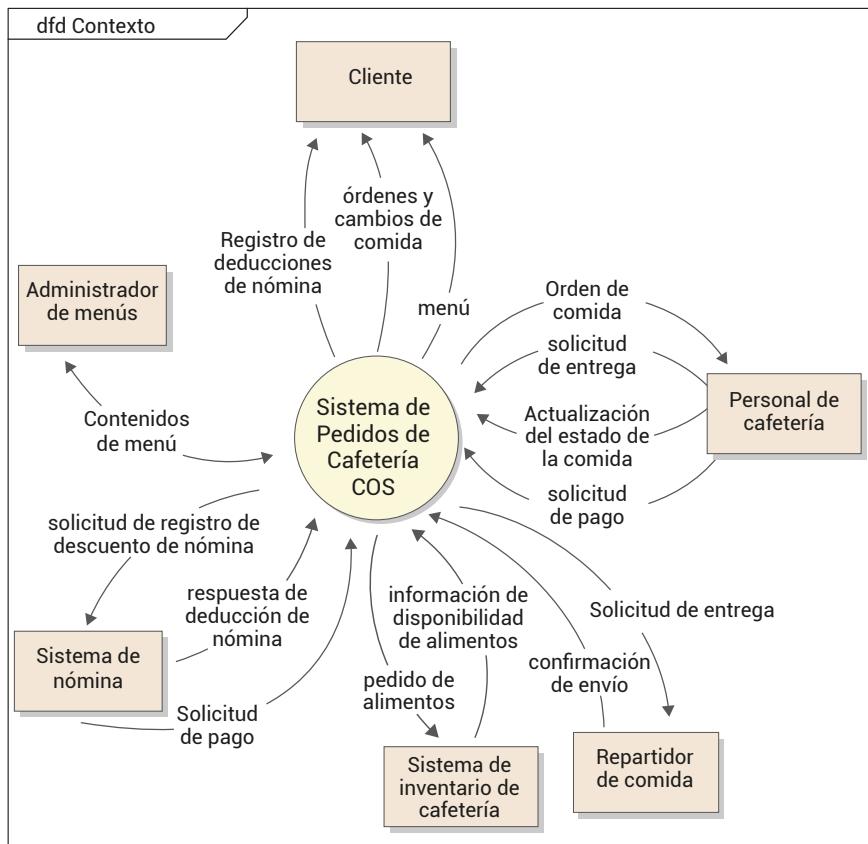
2.6.1. Diagrama de contexto

En la descripción del alcance se establecen los límites y las conexiones entre el sistema que está desarrollando y todo lo demás en el universo. El diagrama de contexto ilustra visualmente este límite. Identifica entidades externas fuera del sistema que interactúan con él de alguna manera, así como datos, control y flujos de materiales entre las entidades y el sistema. Un diagrama de contexto muestra al sistema en su entorno, con las entidades externas (es decir, personas y sistemas) que proporcionan y reciben información o material desde y hacia el sistema. El diagrama de contexto es el nivel superior en un diagrama de datos desarrollado de acuerdo con los principios del análisis estructurado, pero es un modelo útil para todos los proyectos.

El diagrama de contexto se lo utiliza para que los interesados ayuden de forma rápida y simple a definir el alcance del proyecto, y centrarse en los insumos que necesita el sistema, así como las salidas que ofrece. El diagrama de contexto ayuda al equipo de desarrollo a obtener modelos de requisitos (por ejemplo, los actores, casos de uso, y la información de los datos del modelo) y pueden surgir posibles problemas de alcance como nuevas entidades externas.

Figura 12

Diagrama de contexto – Sistema de Pedidos de Cafetería



Nota. Adaptado de *Software Requirements* (p. 585), por Wiegers, K. y Beatty, J., 2013. Microsoft Press.

Como puede apreciar en la figura 12, se muestra el diagrama de contexto para el sistema de pedidos de una cafetería, dónde claramente se establecen las entidades externas que interactuarán con el sistema (COS). Es importante analizar cada uno de los flujos de información que se establecen entre el sistema y cada una de las entidades, ya que esto permitirá definir los límites del sistema. Las entidades que no se consideren en esta representación quedarán fuera del alcance del proyecto.

Desarrollado el diagrama de contexto, con respecto a otros modelos, permite (Gottesdiener, 2005):

- Los flujos de entrada equivalen a los eventos de negocio en la tabla evento-respuesta.
- Los flujos de salida equivalen a las respuestas en la tabla evento-respuesta.
- Las entidades humanas externas pueden convertirse en actores.
- Los flujos de entrada pueden estar asociados con uno o más casos de uso.
- Los nombres de las etiquetas de los flujos pueden convertirse en entidades de datos o atributos en el modelo de datos.
- Los nombres en los flujos de entrada pueden convertirse en nombres generalizados que son detallados en el diccionario de datos.

Para proyectos grandes o aquellos en que el alcance no es suficientemente claro, se recomienda desarrollar un diagrama de contexto independiente que represente a toda la lista de entidades externas y flujos de información que los usuarios podrían gustarles ver incluidos en el proyecto. Seguidamente, revise los objetivos del proyecto y la declaración de la visión y filtre aquellas entidades externas y los flujos de información que considere no permiten obtener logros. Actualice el diagrama de contexto como el sistema “va a ser”.

Es importante considerar que un cambio en el diagrama de contexto implica un cambio en el alcance, que puede afectar al plan del proyecto, cumplimiento de fechas, y recursos del proyecto. Tiene que estar seguro de que el patrocinador, especialista y gerentes están de acuerdo de cualquier cambio y que los procesos de control de cambio de requisitos manejen esta situación.

2.6.2. Mapa de ecosistema

El mapa de ecosistema permite mostrar todos los sistemas que se relacionan con el sistema de interés y que podrían necesitar ser modificados para adaptarse a su nuevo sistema. Los mapas de ecosistemas difieren de los diagramas de contexto en que muestran otros sistemas que tienen una relación con el sistema en el que está trabajando, incluidos aquellos sin interfaces directas (Chen & Beatty, 2012). Puede identificar los sistemas afectados determinando cuáles consumen datos de su sistema. Cuando llegue al punto en que su proyecto no afecte ningún dato adicional, habrá

identificado el límite del alcance de los sistemas que participan en la solución.

Figura 13
Mapa de ecosistema



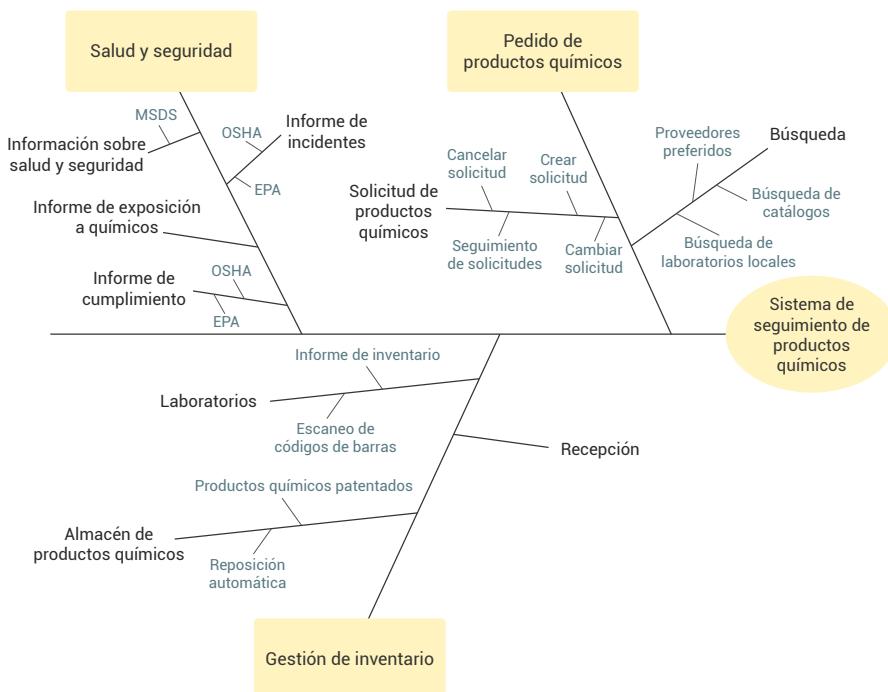
Nota. Sucunuta, M., 2023

La figura 13 es un mapa parcial del ecosistema para el Sistema de Seguimiento Químico. Todos los sistemas se muestran en recuadros (como el Sistema de compras o el Sistema de recepción). En este ejemplo, el sistema principal en el que estamos trabajando se muestra en un recuadro en negrita (Sistema de seguimiento químico), pero si todos los sistemas tienen el mismo estado en su solución, puede usar el mismo estilo de recuadro para todos ellos. Las líneas muestran interfaces entre sistemas (por ejemplo, las interfaces del sistema de compras con el sistema de seguimiento de productos químicos). Las líneas con flechas y etiquetas muestran que la mayor parte de los datos fluyen de un sistema a otro (por ejemplo, los “registros de capacitación” se transfieren de la base de datos de capacitación corporativa al sistema de seguimiento de productos químicos). Algunos de estos mismos flujos también pueden aparecer en el diagrama de contexto.

2.6.3. Árbol de características

Un árbol de características es una representación visual de las características del producto organizadas en grupos lógicos, subdividiendo jerárquicamente cada característica en otros niveles de detalle. El árbol de características proporciona una vista concisa de todas las características planificadas para un proyecto, lo que lo convierte en un modelo ideal para mostrar a los ejecutivos que quieren un rápido vistazo al alcance del proyecto. Un árbol de características puede mostrar hasta tres niveles de funciones, comúnmente llamadas nivel 1 (L1), nivel 2 (L2) y nivel 3 (L3). Las características L2 son subfunciones de las características L1 y las características L3 son subfunciones de las características L2.

Figura 14
Árbol de características



Nota. Adaptado de *Software Requirements* (p. 95), por Wiegers, K. y Beatty, J., 2013. Microsoft Press.

Al planificar un lanzamiento o una iteración, puede definir su alcance seleccionando un conjunto específico de funciones y subfunciones para implementar (Nejmeh y Thomas 2002; Wiegers 2006). Puede implementar una función en su totalidad en una versión específica, o puede implementar solo una parte de ella eligiendo solo ciertas subfunciones L2 y L3. Las versiones futuras podrían enriquecer estas implementaciones rudimentarias agregando más subfunciones L2 y L3 hasta que cada función esté completamente implementada en el producto final. Por lo tanto, el alcance de una versión en particular consiste en un conjunto definido de funciones L1, L2 y/o L3 elegidas del árbol de funciones. Puede marcar un diagrama de árbol de funciones para ilustrar estas asignaciones de funciones entre versiones mediante el uso de colores o variaciones de fuentes. Como alternativa, puede crear una tabla de hoja de ruta de funciones que enumere las subfunciones planificadas para cada versión (Wiegers 2006).

2.6.4. Lista de eventos

Una lista de eventos identifica eventos externos que podrían desencadenar un comportamiento en el sistema. La lista de eventos representa el límite del alcance del sistema, nombrando posibles eventos comerciales activados por usuarios, eventos activados por tiempo (temporales) o eventos de señal recibidos de componentes externos, como dispositivos de hardware. La lista de eventos solo nombra los eventos; los requisitos funcionales que describen cómo responde el sistema a los eventos se detallarían en el SRS mediante el uso de tablas de respuesta a eventos.

En el diagrama cada elemento de la lista indica que desencadena el evento (“El químico” hace algo o llega el “Tiempo para” hacer algo), así como también identifica la acción del evento. Una lista de eventos es una herramienta de alcance útil porque puede asignar ciertos eventos para que se implementen en versiones de productos específicos o iteraciones de desarrollo.

2.7. Mantener el alcance en foco

Una definición de alcance es una estructura, no una camisa de fuerza. Los requisitos comerciales y la comprensión de cómo los clientes usarán el producto brindan herramientas valiosas para lidiar con el cambio de alcance. El cambio de alcance no es algo malo si lo ayuda a dirigir el proyecto hacia la satisfacción de las necesidades cambiantes de los clientes. La

información en el documento de visión y alcance le permite evaluar si los requisitos propuestos son apropiados para su inclusión en el proyecto. Puede modificar el alcance para una iteración futura o para un proyecto completo si lo hace de manera consciente, por las personas adecuadas, por las razones comerciales correctas y con comprensión y aceptación de las compensaciones.

Recuerde, cada vez que alguien solicita un nuevo requisito, el analista debe preguntar: “¿Está esto dentro del alcance?” Una respuesta podría ser que el requisito propuesto está claramente fuera del alcance. Tal vez sea interesante, pero debería abordarse en una versión futura o en otro proyecto. Otra posibilidad es que la solicitud se encuentre obviamente dentro del alcance del proyecto definido. Puede incorporar nuevos requisitos dentro del alcance en el proyecto actual si son de alta prioridad en relación con los otros requisitos que ya estaban comprometidos. Incluir nuevos requisitos a menudo implica tomar la decisión de posponer o cancelar otros requisitos planificados, a menos que esté dispuesto a extender la duración del proyecto.

La tercera posibilidad es que el nuevo requisito propuesto esté fuera del alcance, pero es una buena idea ampliar el alcance para acomodarlo, con los cambios correspondientes en el presupuesto, cronograma y/o personal. Es decir, existe un ciclo de retroalimentación entre los requisitos del usuario y los requisitos comerciales. Esto requerirá que actualice el documento de visión y alcance, que debería haberse colocado bajo el control de cambios en el momento en que se estableció la línea de base. Mantener un registro de por qué se rechazaron los requisitos; tienen una manera de reaparecer.

2.7.1. Uso de los objetivos de negocio para tomar decisiones del alcance

Los objetivos de negocio son el factor más importante a considerar cuando se toman decisiones del alcance. Determinar qué características propuestas o requisitos del usuario agregan el mayor valor con respecto a los objetivos comerciales, programe esos para los primeros lanzamientos. Cuando una parte interesada quiera agregar funcionalidad, considere cómo los cambios sugeridos contribuirán a lograr los objetivos comerciales. Por ejemplo, un objetivo de negocio para generar el máximo de ingresos de un quiosco implica la implementación temprana de funciones que venden más productos o servicios al cliente. Las características deslumbrantes que atraen solo a unos pocos clientes hambrientos de tecnología y que

no contribuyen al objetivo comercial principal no deberían tener una alta prioridad.

Si es posible, cuantifique la contribución que hace la característica hacia los objetivos comerciales, de modo que las personas puedan tomar decisiones de alcance sobre la base de hechos en lugar de emociones (Chen & Beatty, 2012).



¿Una característica específica contribuirá aproximadamente con \$ 1,000, \$ 100,000 o \$ 1,000,000 hacia un objetivo de negocio?

Cuando un ejecutivo solicita una nueva función en la que pensó durante el fin de semana, puede usar el análisis cuantitativo para ayudar a determinar si agregarla es la decisión correcta.

2.7.2. Evaluación del impacto de los cambios en el alcance

Cuando el alcance del proyecto aumenta, el gerente del proyecto generalmente tendrá que renegociar el presupuesto, los recursos, el cronograma y/o el personal planificados. Idealmente, el cronograma y los recursos originales se adaptarán a una cierta cantidad de cambios debido a los amortiguadores de contingencia cuidadosamente incluidos. De lo contrario, deberá volver a planificar una vez que se aprueben los cambios en los requisitos.

Una consecuencia común del cambio de alcance es que las actividades completadas deben volver a trabajarse en respuesta a los cambios. La calidad a menudo sufre si los recursos asignados o el tiempo no aumentan cuando se agrega una nueva funcionalidad. Los requisitos empresariales documentados facilitan la gestión del crecimiento del alcance legítimo a medida que cambian las necesidades comerciales o del mercado. También ayudan a un administrador de proyectos acosado a justificar decir “no”, o al menos “todavía no”, cuando las personas influyentes intentan incluir más funciones en un proyecto demasiado limitado.

2.8. Visión y alcance en proyectos ágiles

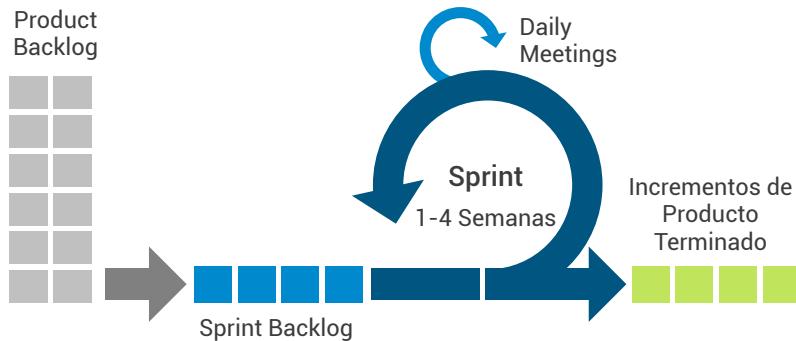
Administrar el alcance en un proyecto ágil, en el que el desarrollo se realiza en una serie de iteraciones de tiempo fijo, tiene un enfoque diferente. El alcance de cada iteración consta de historias de usuarios seleccionadas del

backlog del producto, en función de su prioridad y la capacidad de entrega estimada del equipo para cada período de tiempo. El equipo prioriza los nuevos requisitos frente a los ítems existentes en el *backlog* y los asigna a iteraciones futuras. El número de iteraciones depende de la cantidad total de funcionalidades que se implementará, pero el alcance de cada iteración se controla para garantizar que se complete a tiempo. Alternativamente, algunos proyectos ágiles fijan la duración general del proyecto, pero están dispuestos a modificar el alcance. El número de iteraciones puede seguir siendo el mismo, pero el alcance abordado en las iteraciones restantes cambia según las prioridades relativas de las historias de usuario existentes y recién definidas.

El equipo puede definir una hoja de ruta de iteraciones de alto nivel al comienzo del proyecto, pero la asignación de la historia de usuario para una iteración se realizará al comienzo de cada iteración. Hacer referencia a los requisitos de negocio a medida que el equipo establece el alcance de cada iteración ayuda a garantizar que el proyecto entregue un producto que cumpla con los objetivos de negocio. La misma estrategia se puede utilizar en cualquier proyecto que siga un proceso de desarrollo con límite de tiempo.

Aunque es posible que los proyectos ágiles no creen un documento formal de visión y alcance, los contenidos de la plantilla, que se indican en la figura 11, son relevantes y esenciales para entregar un producto exitoso. Muchos proyectos ágiles llevan a cabo una iteración de planificación inicial (iteración cero) para definir la visión general del producto y otros requisitos de negocio para el proyecto.

Figura 15
Desarrollo ágil



Nota. Adaptado de Cómo funciona la Metodología Scrum: Qué es y cómo utilizarla [Fotografía], por IEBS, sf, [enlace web](#), CC.

Los requisitos de negocio deben definirse para todos los proyectos de software, independientemente de su enfoque de desarrollo. Los objetivos de negocio describen el valor esperado que surge del proyecto y, en un proyecto ágil, se utilizan para ayudar a priorizar el trabajo pendiente para ofrecer el mayor valor comercial en las primeras iteraciones. Las métricas de éxito deben definirse de modo que, a medida que se activan los lanzamientos iterativos, se pueda medir el éxito y el resto de la cartera de pedidos se ajuste en consecuencia. Una declaración de visión describe el plan a largo plazo de lo que será el producto después de que se completen todas las iteraciones.

2.9. Uso de los objetivos de negocio para determinar la finalización

¿Cómo sabe cuándo puede dejar de implementar la funcionalidad? Tradicionalmente, un director de proyecto gestiona el proyecto hasta su finalización. Sin embargo, un BA está íntimamente familiarizado con los objetivos de negocio y puede ayudar a determinar cuándo se ha entregado el valor deseado, lo que implica que el trabajo está hecho.

Si comienza con una visión clara de la solución, y si cada versión o iteración está diseñada para ofrecer solo una parte de la funcionalidad total, habrá terminado cuando complete las iteraciones planificadas previamente. Las

iteraciones completadas deberían haber llevado a una visión del producto completamente realizada que cumpla con los objetivos de negocio.

Sin embargo, particularmente en los enfoques de desarrollo iterativo, el punto final puede ser vago. Dentro de cada iteración, se define el alcance para esa iteración. A medida que avanza el proyecto, disminuye la acumulación de trabajo incompleto. No siempre es necesario implementar todo el conjunto de funciones restantes. Es fundamental tener objetivos de negocio claros para que pueda avanzar hacia la satisfacción de esos objetivos de forma incremental a medida que la información esté disponible. El proyecto está completo cuando las métricas de éxito indican que tiene buenas posibilidades de alcanzar los objetivos de negocio. Los objetivos de negocio imprecisos garantizarán un proyecto abierto sin forma de saber cuándo ha terminado. A los patrocinadores de fondos no les gusta porque no saben cómo presupuestar, programar o planificar tales proyectos. A los clientes no les gusta porque pueden recibir una solución que se entrega a tiempo y dentro del presupuesto, pero que no proporciona el valor que necesitan. Pero ese podría ser el riesgo de trabajar en productos que no se pueden definir claramente desde el principio, a menos que refine los objetivos de negocio a mitad del proyecto.



Concéntrese en definir requisitos de negocio claros para todos sus proyectos. De lo contrario, solo estará deambulando sin rumbo con la esperanza de lograr algo útil sin ninguna forma de saber si está llegando a su destino.

¿Qué le parecieron los temas estudiados? Espero que les hayan resultado interesantes. Ahora, le invito a que realice las siguientes actividades de aprendizaje.



Actividades de aprendizaje recomendadas

Estimado estudiante, finalizada la unidad 2, recomiendo reforzar el estudio de los temas cubiertos en esta unidad, mediante el desarrollo de las siguientes actividades.

1. Busque la plantilla para realizar el análisis de interesados de Volere y liste los roles que utiliza esta metodología.

- Revise [Stakeholder Analysis and Engagement](#), y determine la importancia de los interesados desde el punto de vista organizacional.
- Dada la siguiente tabla, complete la categoría de los interesados.

Tabla a completar

DESARROLLO DE REQUERIMIENTOS				GESTIÓN DE REQUERIMIENTOS
ROL	Define los requisitos del negocio	Desarrolla requisitos de usuario	Especifica requisitos de software	
Auditor				
Comprador				
Administrador de la base de datos				
Analista de documentación				
Experto financiero				
Invitado				
Especialista en Help desk				
Experto legal				
Consultor				
Instalador del producto				
Especialista en ventas				
Programador				

- Analice detenidamente el documento de visión y alcance del caso, que se indica en el [anexo 2 Documento de visión y alcance](#). Observe detenidamente cada una de las partes y complete aquellas que considere que hace falta las definiciones.
- Estimado estudiante, realice la autoevaluación 2.



Autoevaluación 2

Lea detenidamente cada una de las preguntas y elija la opción correcta:

1. Un experto en seguridad, para el desarrollo de un sistema para una empresa, puede cumplir el rol de:
 - a. Interesado.
 - b. Analista.
 - c. Desarrollador.

2. El interesado más obvio en cualquier proyecto de desarrollo de *software* es el:
 - a. Consultor.
 - b. Propietario.
 - c. Contador.

3. Los interesados con respecto al equipo del proyecto o la organización pueden ser:
 - a. Internos.
 - b. Externos.
 - c. Interno o externo.

4. A las personas u organizaciones que tienen influencia directa, indirecta o se ven influenciados por un proceso de software, se los conoce como:
 - a. Tester.
 - b. Desarrollador.
 - c. Interesados.

5. Para un sistema de nómina, un Interesado podría ser:
 - a. Empleado.
 - b. Estudiante.
 - c. Sistema financiero.

6. ¿Cuál de los siguientes Interesados, no es considerado como miembro del “Equipo del proyecto”?
 - a. Analista de negocio.
 - b. Tester.
 - c. Experto en usabilidad.
7. Cuando se realiza el visionamiento y alcance, las organizaciones que construyen software comercial a menudo crean un documento de:
 - a. Requisitos de mercado.
 - b. Especificación de requisitos de software.
 - c. Casos de uso.
8. Un documento de requisitos de mercado contiene:
 - a. Los componentes arquitectónicos del software.
 - b. La definición del modelo de datos físico.
 - c. El detalle, los segmentos del mercado objetivo y los temas que se refieren al éxito comercial.
9. En un diagrama de contexto, el círculo representa:
 - a. Entidades externas.
 - b. Sistema.
 - c. Flujos de información.
10. El visionamiento se aplica como un todo al:
 - a. Interesado.
 - b. Producto.
 - c. Organización.

[Ir al solucionario](#)

Resultado de aprendizaje 2

- Escoge las mejores estrategias para recolectar información de los interesados.

Por medio de este resultado de aprendizaje conocerá y aplicará las técnicas que le permitan obtener información a partir de diferentes fuentes de información. Podrá decidir qué estrategia es la que mejor se adapte a la naturaleza del proyecto y al equipo de desarrollo, para ello deberá estudiar al detalle cada una de estas estrategias y aplicarlas mediante un proceso debidamente planificado y organizado.

Contenidos, recursos y actividades de aprendizaje



Semana 5

Unidad 3. Obtención de requisitos

3.1. Fuente de los requisitos

Para realizar la obtención de información, es preciso determinar dónde se encuentra la información; esto permitirá establecer la estrategia y herramientas de obtención a utilizar. Por este motivo es fundamental identificar la fuente de los requisitos (*¿dónde está la información?*), esto requiere disponer de una lista de fuentes de requisitos. Esta lista es un inventario de personas, documentos específicos y fuentes externas de información desde dónde se podría obtener los requisitos. Para esto se recomienda lo siguiente:

1. Identificar a los Interesados relevantes de los que se podría obtener requisitos.
 - Asegúrese de considerar a todos los interesados del proyecto. Incluya a los clientes que patrocinan y defienden el desarrollo del software, los usuarios que interactuarán directa o indirectamente con el software y otros que tienen conocimiento o interés en el producto.

- Desarrollar un plan de obtención por cada interesado. Tenga en cuenta que los interesados suelen estar ocupados y necesitan un aviso previo para participar en la obtención de requisitos.
2. Identificar cualquier documentación que pueda utilizar como fuente de información de requisitos.

Incluya documentación física que pueda usar como fuente de información de requisitos, como:

- Documentación de sistemas existentes e interconectados.
- Solicitudes de cambio, lista de defectos de *software*, registros de quejas de clientes y listas de problemas.
- Guías de usuario, materiales de capacitación y lineamientos de procedimientos de trabajo.
- Documentación de la mesa de servicio.
- Guías de políticas y procedimientos.
- Código en sistemas existentes.

3. Identificar fuentes externas de información.

Incluya:

- Departamentos o compañías de servicios que proveen datos de encuestas de mercado y análisis de la industria.
- Descripciones y reseñas de productos de *software* competitivos y materiales de los productos.
- Ventas, *marketing* y materiales de comunicación.
- Regulaciones, lineamientos y leyes de las agencias de gobierno y organismos reguladores.

Para realizar las actividades de obtención, considere estas recomendaciones que le ayudarán a focalizar de mejor manera cada técnica.

3.2. Obtención

Los clientes y usuarios a menudo no entienden cómo funciona el diseño y desarrollo de *software*, y no pueden especificar sus propios requisitos de *software* de una manera que funcione para los desarrolladores. Por su parte, los desarrolladores de *software* a menudo no entienden los problemas y

necesidades de los clientes y usuarios lo suficientemente bien como para especificar los requisitos en su nombre (Gottesdiener, 2005).

La obtención (o elicitación), es el proceso de identificación de las necesidades y limitaciones de los distintos Interesados para un sistema de software; obtención no es lo mismo que “recolectar los requisitos”. Tampoco es una simple cuestión de transcribir exactamente lo que dicen los usuarios. La obtención es un proceso colaborativo y analítico que incluye actividades para recolectar, descubrir, extraer y definir requisitos. La obtención se utiliza para descubrir los requisitos de negocio, de usuario, funcionales y no funcionales, junto con otro tipo de información. La obtención de requisitos es quizá el aspecto más desafiante, crítico, propenso a errores y de comunicación intensa del desarrollo de software.

Involucrar a los usuarios en el proceso de obtención es una forma de obtener soporte y apoyo para el proyecto. Si usted es el BA, trate de entender los procesos detrás de los requisitos de los usuarios. Recorra los procesos que los usuarios siguen para tomar decisiones sobre su trabajo, y extraer la lógica subyacente. Asegúrese de que todo el mundo entiende por qué el sistema debe realizar ciertas funciones. Busque los requisitos propuestos que afectan a procesos o reglas de negocio obsoletos o ineficaces que no deben ser incorporados en un nuevo sistema.

Los analistas de negocio deben crear un entorno propicio para una exploración minuciosa del producto que se especifique. Para facilitar una comunicación clara, utilice el vocabulario del dominio de negocio en lugar de forzar a los clientes a entender la jerga técnica, es recomendable registrar términos significativos de dominio de aplicación en un glosario, en lugar de asumir que todos los participantes comparten las mismas definiciones. Los clientes deben entender que una discusión acerca de la posible funcionalidad no es un compromiso para incluirla en el producto. La lluvia de ideas y la imaginación de las posibilidades es un asunto aparte del análisis de prioridades, la viabilidad y las realidades restrictivas.

El resultado del desarrollo de los requisitos es una comprensión común de las necesidades por los interesados del proyecto. Cuando los desarrolladores entienden esas necesidades, pueden explorar soluciones alternativas para abordarlas. Los participantes en la obtención deben resistir la tentación de diseñar el sistema hasta que entiendan el problema. Enfatizar las tareas del usuario en lugar de las interfaces de usuario y centrarse en las necesidades reales más que en los deseos expresados,

ayudan a evitar que el equipo se desvíe al especificar prematuramente los detalles del diseño (Bron, 2020).

La naturaleza del desarrollo de requisitos es cíclica. Realizará algunas actividades de obtención, estudiará lo que aprendió, escribirá algunos requisitos, quizás determinará que está faltando cierta información, entonces deberá realizar nuevamente la obtención y así sucesivamente. No espere solo hacer un par de talleres de obtención y luego declarar la victoria y seguir adelante.

Existen algunas dificultades que son importantes abordarlas a la hora de realizar las actividades de obtención, entre las que se incluye:

- Necesidades diferentes y a veces conflictivas entre los diferentes tipos de usuarios.
- Requisitos implícitos o supuestos por parte de los Interesados.
- Obtener acceso a Interesados con conocimiento.
- Incapacidad para imaginar nuevos o diferentes formas de uso del software.
- Tener un alto número de requisitos interrelacionados.
- Tener un tiempo limitado para obtener requerimientos desde Interesados ocupados.
- Superar la resistencia al cambio.

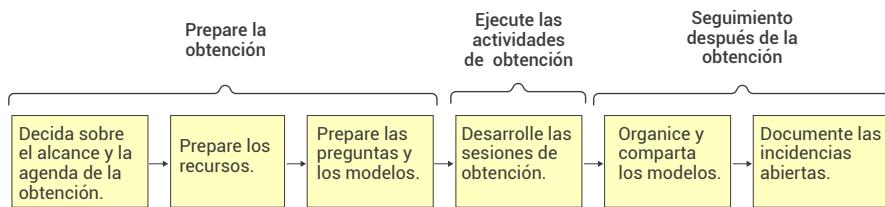
Para empezar conociendo las técnicas de obtención, revise el recurso educativo métodos de obtención, del documento [Ingeniería de requisitos](#). Incluye otros temas que puede utilizar para reforzar los apartados que se cubren en la asignatura.

Como puede apreciar existen diferentes técnicas, cada una de ellas orientadas a necesidades puntuales de obtención de información, que deben ser utilizadas de acuerdo con una planificación debidamente coordinada.

La figura 16 se muestra las actividades para una única sesión de elicitation de requisitos, pero para poder desarrollar estas actividades es necesario conocer las técnicas de obtención que se pueden utilizar, bajo este esquema.

Figura 16

Actividades para una simple sesión de obtención



Nota. Adaptado de *Actividades para una simple sesión de obtención* (p. 130), por K. Wiegers y J. Beatty, 2013, Microsoft Press

3.3. Técnicas para obtener requisitos

Se pueden emplear numerosas técnicas de obtención para los proyectos de software. El equipo de desarrollo del proyecto debe considerar no solamente el uso de una sola técnica de obtención, ya que siempre habrá información por descubrir, además que los Interesados preferirán diferentes enfoques. Un usuario puede articular claramente cómo usa el sistema, mientras que es posible que deba observar a otro realizar su trabajo para alcanzar el mismo nivel de comprensión.

Las técnicas de obtención incluyen actividades, en las que interactúa con los interesados para obtener requisitos y actividades independientes, en las que trabaja por su cuenta para descubrir información. Las actividades se centran en descubrir los requisitos del negocio y del usuario. Es necesario trabajar directamente con los usuarios porque los requisitos de los usuarios abarcan las tareas que los usuarios deben realizar con el sistema. Para obtener los requisitos de negocio, deberá trabajar con personas como el patrocinador del proyecto. La mayoría de los proyectos utilizarán una combinación de actividades de obtención facilitadas e independientes. Cada técnica ofrece una exploración diferente de los requisitos o incluso puede revelar requisitos completamente diferentes. A continuación se describen algunas de las técnicas comúnmente utilizadas para obtener requisitos.

3.3.1. Entrevistas

La forma más obvia de averiguar qué necesitan los usuarios de un sistema de software es preguntarles. Las entrevistas son una fuente tradicional de entrada de requisitos tanto para productos comerciales como para sistemas de información, en todos los enfoques de desarrollo de *software*. La mayoría de los BA facilitarán alguna forma de entrevistas individuales o en grupos pequeños para obtener los requisitos de sus proyectos. Los proyectos ágiles hacen un uso extensivo de las entrevistas como mecanismo para lograr la participación directa del usuario. Las entrevistas son más fáciles de programar y dirigir que las actividades para grupos grandes, como lo son los talleres.

Si es nuevo en un dominio de aplicación, las entrevistas con expertos pueden ayudarlo a ponerse al día rápidamente. Esto le permitirá preparar borradores de requisitos y modelos para usar en otras entrevistas o en talleres. Si puede establecer una buena relación con los entrevistados, se sentirán más seguros cuando comparten sus pensamientos uno a uno o en un grupo pequeño que en un taller más grande, particularmente sobre temas delicados. También es más fácil lograr que los usuarios acepten participar en el proyecto o revisar los requisitos existentes durante una entrevista individual o en un grupo pequeño que en un entorno de grupo grande.

Las entrevistas son conversaciones cara a cara en la que un entrevistador hace preguntas para obtener información del entrevistado. De acuerdo con Gottesdiener (2005) las entrevistas pueden ser de dos tipos: estructuradas (con preguntas preparadas con anterioridad), o no estructuradas (sin preguntas predefinidas).

- **Entrevistas estructuradas:** se dan cuando el entrevistador ha elaborado un conjunto de preguntas acorde a un propósito asociado con el entrevistado. Algunas de las preguntas pueden ser abiertas o preguntas en donde la respuesta tenga un conjunto de posibilidades conocidas.
- **Entrevistas no estructuradas:** donde el entrevistado y el entrevistador sin ninguna pregunta predefinida discuten de temas de interés abiertamente.

Para que una entrevista sea exitosa depende de los siguientes factores:

- Nivel de comprensión del dominio por parte del entrevistador.
- La experiencia del entrevistador.
- Habilidad del entrevistador en la documentación de los debates.
- Disposición del entrevistado para proporcionar la información pertinente.
- Grado de claridad del entrevistado acerca de lo que la empresa requiere del sistema.
- Armonía entre el entrevistador y el entrevistado.

A continuación, se presentan algunas sugerencias para realizar entrevistas. Estos son también consejos útiles para realizar talleres de elicitation.

- **Establecer una buena relación.** Para comenzar una entrevista, preséntese si los asistentes aún no lo conocen, revise la agenda, recuerde a los asistentes los objetivos de la sesión y aborde cualquier pregunta preliminar o inquietud que tengan los asistentes.
- **Manténgase en el ámbito.** Al igual que con cualquier sesión de obtención, mantenga la discusión enfocada en su objetivo. Incluso cuando está hablando con una sola persona o un grupo pequeño, existe la posibilidad de que la entrevista se salga del tema.
- **Prepare preguntas y modelos con anticipación.** Prepárese para las entrevistas redactando cualquier material que pueda de antemano, como una lista de preguntas para guiar la conversación. Los borradores de materiales brindarán a sus usuarios un punto de partida para pensar. Las personas a menudo pueden criticar el contenido más fácilmente de lo que pueden crearlo.
- **Sugiera ideas.** En lugar de simplemente transcribir lo que dicen los clientes, un BA creativo propone ideas y alternativas durante la obtención. A veces, los usuarios no se dan cuenta de las capacidades que pueden proporcionar los desarrolladores; es posible que se entusiasmen cuando sugiera una funcionalidad que hará que el sistema sea especialmente valioso. Cuando los usuarios realmente no pueden expresar lo que necesitan, quizás pueda verlos trabajar y sugerir formas de automatizar partes del trabajo. Los BA pueden pensar fuera de la caja mental que limita a las personas que están demasiado cerca del dominio del problema.

- **Escuchar activamente.** Practique las técnicas de escucha activa (inclinarse hacia adelante, mostrar paciencia, dar retroalimentación verbal y preguntar cuando algo no está claro) y parafrasear (reiterar la idea principal del mensaje de un orador para mostrar su comprensión de ese mensaje).

3.3.2. Talleres

Los talleres fomentan la colaboración de los interesados en la definición de los requisitos. Gottesdiener (2005) define un taller de requisitos como “una reunión estructurada” en la que un grupo cuidadosamente seleccionado de interesados y expertos en el tema trabajan juntos para definir, crear, refinar y alcanzar los entregables de cierre (como modelos y documentos). Los talleres son sesiones facilitadas con múltiples actores y roles formales, como es el facilitador y el escritor. Los talleres a menudo incluyen varios tipos de interesados, desde usuarios hasta desarrolladores y probadores. Se utilizan para obtener los requisitos de múltiples Interesados simultáneamente. Trabajar en un grupo es más eficaz para resolver los desacuerdos que hablar con las personas individualmente. Además, los talleres son útiles cuando se necesita un cambio urgente de obtención debido a restricciones de horario.

El facilitador juega un papel crítico en la planificación del taller, seleccionando a los participantes y guiándolos hacia un resultado exitoso. Los analistas de negocios frecuentemente facilitan talleres de obtención. Cuando un equipo está comenzando con nuevos enfoques para la obtención de requisitos, considere la posibilidad de tener un facilitador externo o un segundo analista de negocio para los talleres iniciales.

De esta manera, el analista de negocio puede dedicar toda su atención a la discusión. Si el único analista de negocio también actúa como facilitador, debe tener en cuenta cuándo habla como facilitador y cuando participa en la discusión. Un escritor ayuda al facilitador capturando los puntos que surgen durante la discusión. Es extremadamente difícil facilitar, escribir, y participar simultáneamente y hacer un buen trabajo de los tres.

Los talleres pueden ser intensivos en recursos, a veces se requiere de numerosos participantes durante varios días a la vez. Deben estar bien planificados para evitar perder tiempo. Minimizar el tiempo perdido al entrar en un taller con borradores de materiales preparados con antelación. Por ejemplo, puede redactar casos de uso que pueden revisarse como un grupo

en lugar de que el grupo completo los prepare juntos. Rara vez tiene sentido iniciar un taller con una pizarra completamente en blanco. Utilizar otras técnicas de obtención antes de los talleres, y luego reunir a los interesados para trabajar solo en las áreas necesarias.

Los siguientes son algunos consejos para realizar talleres de obtención efectivos, muchos de los cuales también se aplican a las entrevistas.

- **Establecer y hacer cumplir las reglas básicas.** Los participantes del taller deben ponerse de acuerdo sobre algunos principios operativos básicos. Por ejemplo, comenzar y terminar a tiempo; regresar de los descansos puntualmente; silenciar dispositivos electrónicos; mantener una conversación a la vez; esperar que todos contribuyan; y centrar los comentarios y las críticas en temas más que en individuos. Una vez establecidas las reglas, asegúrese de que los participantes las sigan.
- **Contar con todos los roles del equipo.** Un facilitador debe asegurarse de que las siguientes tareas estén cubiertas por personas en el taller: tomar notas, controlar el tiempo, administrar el alcance, administrar las reglas básicas y asegurarse de que todos sean escuchados. Un secretario(a) puede registrar lo que está pasando, mientras que otra persona controlaría el tiempo.
- **Planifica una agenda.** Cada taller necesita un plan claro. Cree el plan y la agenda del taller con anticipación y comuníquese a los participantes para que sepan los objetivos y qué esperar y en consecuencia puedan prepararse.
- **Mantenga el alcance.** Consulte los requisitos de negocio para confirmar si los requisitos de usuario propuestos se encuentran dentro del alcance del proyecto actual. Mantenga cada taller enfocado en el nivel correcto de abstracción para los objetivos de esa sesión. Los grupos se sumergen fácilmente en detalles que distraen durante las discusiones de requisitos. Esas discusiones consumen tiempo que el grupo que debería dedicar a desarrollar una comprensión de mayor nivel de los requisitos del usuario.
- **Timebox.** Considere asignar un período fijo de tiempo a cada tema de discusión. Es posible que la discusión deba completarse más tarde, pero el *timeboxing* ayuda a evitar la trampa de dedicar mucho más tiempo del previsto al primer tema y descuidar por completo otros

temas importantes. Al cerrar una discusión con límite de tiempo, resuma el estado y los próximos pasos antes de abandonar el tema.

- **Mantenga un equipo pequeño, pero incluya a los Interesados adecuados.** Los grupos pequeños pueden trabajar mucho más rápido que los equipos grandes. Los talleres de elicitation con más de cinco o seis participantes activos pueden verse envueltos en viajes secundarios, conversaciones simultáneas y disputas. Considere la posibilidad de realizar varios talleres en paralelo para explorar los requisitos de las diferentes clases de usuarios. Los participantes del taller podrían incluir al Product Champion y otros representantes de los usuarios, tal vez un experto en la materia, un BA, un desarrollador y un evaluador. El conocimiento, la experiencia y la autoridad para tomar decisiones son requisitos para participar en los talleres de obtención.

3.3.3. Grupos de enfoque

Un grupo de enfoque es un conjunto representativo de usuarios que se reúnen en una actividad facilitadora para generar información e ideas sobre los requisitos funcionales y de calidad de un producto. Las sesiones de grupo de enfoque deben ser interactivas, permitiendo a todos los usuarios la oportunidad de expresar sus pensamientos. Los grupos focales son útiles para explorar las actitudes, impresiones, preferencias y necesidades de los usuarios. Son especialmente valiosos si está desarrollando productos comerciales y no tiene acceso fácil a los usuarios finales dentro de su empresa.

A menudo, tendrás una base de usuarios amplia y diversa para escoger, así que seleccione cuidadosamente a los miembros del grupo de enfoque. Incluya a los usuarios que hayan utilizado versiones anteriores o productos similares a los que está implementando. Seleccione un grupo de usuarios del mismo tipo (y mantenga varios grupos de enfoque para las diferentes clases de usuario) o seleccione un grupo que represente el espectro completo de clases de usuario para que todos estén igualmente representados.

Los grupos focales deben ser facilitados. Usted tendrá que mantenerlos en el tema, pero sin influir en las opiniones que se expresan. Es posible que desee grabar la sesión para que pueda volver atrás y escuchar atentamente los comentarios. No esperen análisis cuantitativos de los grupos focales, sino más bien mucha retroalimentación subjetiva que puede ser evaluada

y priorizada a medida que se desarrollan los requisitos. Las sesiones de sensibilización con grupos focales se benefician de muchos de los mismos consejos descritos anteriormente para los talleres. Normalmente, los participantes en los grupos focales no tienen autoridad para tomar decisiones sobre los requisitos.

3.3.4. Observación

Permite realizar una evaluación del ambiente de trabajo de los interesados. Esta técnica es apropiada cuando se documentan los detalles sobre un proceso actual o si el proyecto está destinado a mejorar o cambiar un proceso existente (IIBA, 2015). Es común que al pedir a los usuarios que describan cómo hacen su trabajo, es probable que les resulte difícil ser precisos: es posible que falten detalles o que sean incorrectos. A menudo, esto se debe a que las tareas son complejas y es difícil recordar cada detalle. En otros casos, se debe a que los usuarios están tan familiarizados con la ejecución de una tarea que no pueden articular todo lo que hacen. Quizás la tarea es tan habitual que ni siquiera piensan en ella. A veces se puede aprender mucho observando exactamente cómo los usuarios realizan sus tareas (Wiegert & Beatty, 2013).

Las observaciones consumen mucho tiempo, por lo que no son adecuadas para todos los usuarios ni para todas las tareas. Para evitar interrumpir las actividades laborales asignadas regularmente a los usuarios, limite cada tiempo de observación a dos horas o menos. Seleccione tareas importantes o de alto riesgo y varias clases de usuarios para las observaciones. Si usa observaciones en proyectos ágiles, haga que el usuario demuestre solo las tareas específicas relacionadas con la próxima iteración.

Observar el flujo de trabajo de un usuario en el entorno de tareas le permite al BA validar la información recopilada de otras fuentes, identificar nuevos temas para entrevistas, ver problemas con el sistema actual e identificar formas en que el nuevo sistema puede respaldar mejor el flujo de trabajo. El BA debe abstraer y generalizar más allá de las actividades del usuario observado para garantizar que los requisitos capturados se apliquen a la clase de usuario en su conjunto, no solo a ese individuo. Un BA hábil también puede sugerir ideas para mejorar los procesos de negocio actuales del usuario.

Hay dos enfoques básicos para la técnica de observación:

- Pasivo o invisible (silenciosa): en este enfoque, el BA observa al usuario trabajando en su rutina de trabajo, pero no hace preguntas. El BA solamente registra lo que ha observado, y queda al margen de lo que pasa. El BA espera hasta que el proceso entero se termine antes de hacer cualquier pregunta. El BA deberá observar el proceso de negocio varias veces para asegurarse de que entiende cómo el proceso funciona hoy y por qué funciona del modo en que lo hace.
- Activo o visible: en este enfoque, mientras el BA observa el proceso actual y toma notas, puede dialogar con el usuario. Permiten que el BA interrumpa al usuario en medio de una tarea y le haga una pregunta. Esto es útil para comprender inmediatamente por qué un usuario hizo una elección o para preguntarle en qué estaba pensando cuando realizó alguna acción.

Documente lo que observa para su posterior análisis después de la sesión. También puede considerar grabar la sesión en video, si las políticas lo permiten, para que pueda actualizar su memoria más tarde.

3.3.5. Cuestionarios

Los cuestionarios son una forma de encuestar a grandes grupos de usuarios para comprender sus necesidades. Son económicos, lo que los convierte en una opción lógica para obtener información de grandes poblaciones de usuarios, y se pueden administrar fácilmente a través de fronteras geográficas. Los resultados analizados de los cuestionarios se pueden utilizar como entrada para otras técnicas de obtención. Por ejemplo, puede usar un cuestionario para identificar los puntos débiles más grandes de los usuarios con un sistema existente y luego usar los resultados para discutir la priorización con los tomadores de decisiones en un taller. También puede usar cuestionarios para encuestar a los usuarios de productos comerciales para obtener comentarios.

A continuación algunas sugerencias para elaborar los cuestionarios:

- Proporcione opciones de respuesta que cubran el conjunto completo de posibles respuestas.
- Haga que las opciones de respuesta sean mutuamente excluyentes (sin superposiciones en rangos numéricos) y exhaustivas (enumere

todas las opciones posibles y / o tenga un lugar de escritura para una opción que no pensó).

- No formule una pregunta de manera que implique una respuesta "correcta".
- Si utiliza escalas, utilícelas de manera consistente en todo el cuestionario.
- Utilice preguntas cerradas con dos o más opciones específicas si desea utilizar los resultados del cuestionario para el análisis estadístico. Las preguntas abiertas permiten a los usuarios responder de la manera que quieran, por lo que es difícil buscar puntos en común en los resultados.
- Considere consultar con un experto en diseño y administración de cuestionarios para asegurarse de hacer las preguntas correctas a las personas adecuadas.
- Siempre pruebe un cuestionario antes de distribuirlo. Es frustrante descubrir demasiado tarde que una pregunta fue redactada de forma ambigua o darse cuenta de que se omitió una pregunta importante.
- No haga demasiadas preguntas o la gente no responderá.

Son varias las ventajas que nos ofrecen los cuestionarios, ya que nos permite obtener información subjetiva de forma rápida, a un bajo costo, de forma remota a un gran número de personas. La forma de aplicar el cuestionario también es diversa, desde una manera tradicional mediante un papel impreso hasta utilizando herramientas tecnológicas con diseños acordes al grupo que se desea aplicar.

Contrariamente, una de las desventajas de los cuestionarios es que la información obtenida sea sesgada por diversos motivos, por ejemplo, la muestra de personas que se escogieron para aplicar el cuestionario, las personas que estaban dispuestos a responder, no hay una relación directa con los encuestados por lo que no se puede contextualizar las preguntas, preguntas ambiguas, etc.

3.3.6. Análisis de interfaz de sistema

El análisis de interfaz es una técnica de obtención independiente que implica examinar los sistemas a los que se conectará su sistema. El análisis de la interfaz del sistema revela requisitos funcionales relacionados con el intercambio de datos y servicios entre sistemas. Los diagramas de contexto y los mapas de ecosistemas son una opción obvia para comenzar a encontrar interfaces para estudios posteriores. De hecho, si encuentra una interfaz que tiene requisitos asociados y que no está representada en uno de estos diagramas, los diagramas están incompletos (IIBA, 2015).

Para cada sistema que interactúe el nuevo sistema, identifique la funcionalidad en el otro sistema que podría generar requisitos para su sistema. Estos requisitos podrían describir qué datos pasar al otro sistema, qué datos se reciben de él y las reglas sobre esos datos, como los criterios de validación. También puede descubrir la funcionalidad existente que no necesita implementar en su sistema. A través del análisis de la interfaz del sistema, puede aprender cómo otros sistemas pasan las órdenes al sistema de gestión de pedidos, cómo se realiza la validación, y por qué no es necesario crear esta función.

3.3.7. Análisis de interfaz de usuario

El análisis de la Interfaz de Usuario (UI) es una técnica de extracción independiente en la que se estudian los sistemas existentes para descubrir los requisitos de usuario y funcionales. Lo mejor es interactuar con los sistemas existentes directamente, pero si es necesario puede utilizar capturas de pantalla. Los manuales de usuario para las implementaciones de software empaquetado compradas a menudo contienen capturas de pantalla que funcionarán como punto de partida. Si no existe un sistema, es posible que pueda ver interfaces de usuario de productos similares.

Al trabajar con soluciones empaquetadas o con un sistema existente, el análisis de la interfaz de usuario puede ayudarle a identificar una lista completa de pantallas que le permitan descubrir características potenciales. Al navegar por la interfaz de usuario existente, puede conocer los pasos comunes que los usuarios toman en el sistema y los casos de uso preliminar para revisarlos con los usuarios. Análisis de la interfaz de usuario puede revelar piezas de datos que los usuarios necesitan ver. Es una gran manera de ponerse al día sobre cómo funciona un sistema existente (a menos que necesite mucho entrenamiento para hacerlo). En lugar de preguntar a los

usuarios cómo interactúan con el sistema y qué pasos toman, tal vez usted puede llegar a un entendimiento inicial usted mismo.

No asuma que ciertas funcionalidades son necesarias en el nuevo sistema solo porque usted lo encontró en un sistema existente. Por otra parte, no asuma que debido a que la interfaz de usuario es de esa manera en el sistema actual debe ser implementado de esa misma manera en el futuro sistema.

3.3.8. Análisis de documentos

El análisis de documentos implica examinar cualquier documentación existente sobre los posibles requisitos de software. La documentación más útil incluye especificaciones de requisitos, procesos empresariales, colecciones aprendidas por experiencia y manuales de usuario para aplicaciones existentes o similares. Los documentos pueden describir los estándares corporativos o de la industria que deben ser seguidos o las regulaciones con las cuales el producto debe cumplir. Al reemplazar un sistema existente, la documentación pasada puede revelar la funcionalidad que podría necesitar ser retenida, así como la funcionalidad obsoleta.

Para la implementación de soluciones empaquetadas, la documentación del proveedor menciona la funcionalidad que sus usuarios pueden necesitar, pero es posible que tenga que explorar más a fondo cómo implementarla en el entorno de destino. Las revisiones comparativas señalan deficiencias en otros productos que usted podría dirigir para ganar una ventaja competitiva. Los informes de problemas y las solicitudes de mejoras recopiladas de los usuarios por el personal de asistencia y el personal de soporte técnico pueden ofrecer ideas para mejorar el sistema en futuras versiones.

El análisis de documentos es una manera de ponerse al día en un sistema existente o en un nuevo dominio. Hacer algunas investigaciones y redactar algunos requisitos de antemano reduce el tiempo necesario para la reunión. El análisis de documentos puede revelar información que la gente no le dice, ya sea porque no piensa en ella o porque no la conoce. Por ejemplo, si está creando una nueva aplicación de centro de llamadas, puede encontrar alguna lógica empresarial complicada descrita en el manual del usuario para una aplicación existente. Tal vez los usuarios ni siquiera saben acerca de esta lógica. Usted puede utilizar los resultados de este análisis como entrada a las entrevistas del usuario.

Un riesgo con esta técnica es que los documentos disponibles pueden no estar actualizados. Es posible que los requisitos hayan cambiado sin que se actualicen las especificaciones o se pueda documentar la funcionalidad que no se necesita en un nuevo sistema.

Todas estas técnicas de obtención requieren de una planificación, ejecución y resultados, tal como se indica en la figura 16. A continuación se analizan cada una de estas actividades.



Semana 6

3.4. Planificar la obtención

Al principio de un proyecto, el BA planifica el enfoque del proyecto para la obtención de requisitos. Incluso un simple plan de acción aumenta las posibilidades de éxito y establece expectativas realistas para los Interesados. Solo al obtener un compromiso explícito sobre los recursos de obtención, el cronograma y los entregables, puede evitar que los participantes se aparten para hacer otro trabajo. Un plan de obtención incluye las técnicas que usará, cuándo planea usarlas y con qué propósito. Al igual que con cualquier plan, utilícelo como guía y recordatorio a lo largo del proyecto, pero tenga en cuenta que es posible que deba cambiar el plan a lo largo del proyecto. Su plan debe abordar los siguientes elementos:

- Objetivos de elicitation.
- Estrategia de elicitation y técnicas planificadas.
- Estimaciones de cronograma y recursos.
- Documentos y sistemas necesarios para la obtención.
- Productos esperados de los esfuerzos de elicitation.
- Riesgos en la obtención.

Revise la siguiente infografía donde se detalla cada elemento:

Plan de obtención

Muchos BA tienen su técnica de “excusarse”, comúnmente en entrevistas y talleres, y no piensan en usar otras técnicas que podrían reducir las necesidades de recursos o aumentar la calidad de la información descubierta. Rara vez un BA obtiene los mejores resultados mediante el uso

de una sola técnica de obtención en un proyecto. Las técnicas de obtención se aplican a todo el espectro de enfoques de desarrollo. La selección de las técnicas de obtención debe basarse en las características del proyecto. En la figura 17 se sugiere las técnicas de obtención que son más probables de ser útiles para varios tipos de proyectos.

Figura 17

Técnicas de obtención sugeridas por tipo de proyecto

	Entrevistas	Talleres	Grupos focales	Observación	Cuestionarios	Ánalisis de interfaz del sistema	Ánalisis de interfaz de usuario	Ánalisis de documentos
Software para mercado masivo	X		X		X			
Software corporativo interno	X	X	X	X		X		X
Reemplazo de un sistema existente	X	X		X		X	X	X
Mejoramiento de una sistema existente	X	X				X	X	X
Aplicaciones nuevas	X	X				X		
Implementación de software empaquetado	X	X		X		X		X
Sistemas embebidos	X	X				X		X
Interesados geográficamente distribuidos	X	X			X			

Nota. Adaptado de *Software Requirements* (p. 130), por Wiegers, K. y Beatty, J., 2013. Microsoft Press.

Si está desarrollando un proyecto en el cual se desea construir una nueva aplicación, es probable que obtenga los mejores resultados con una combinación de entrevistas a interesados, talleres y análisis de la interfaz de sistemas existentes.

Para complementar sobre las herramientas de obtención de información para definir los requisitos, revise [Ingeniería de Requerimientos – Recolección de requerimientos](#).

Como puede ver, la función del BA es la de utilizar la herramienta apropiada para obtener información que permita ir conociendo la situación actual, para de esta manera entender que es lo que necesita el usuario. De ninguna manera el usuario va a definir las características del sistema, es función del analista.



Semana 7

3.5. Preparación para la obtención

Las actividades de obtención requieren de preparación, para hacer mejor uso del tiempo de cada uno de los interesados. Cuanto mayor es el grupo que participa en la sesión, más importante es la preparación. Prepare cada sesión decidiendo el alcance, comunicando la agenda, preparando preguntas y elaborando materiales que podrían ser útiles durante la sesión. Como se indica en la figura 16 se observan las actividades que se deben considerar para la preparación. Los siguientes consejos le ayudarán a prepararse de manera apropiada.

- **Planifique el alcance y la agenda de la sesión:** decidir sobre el alcance de la sesión de obtención, teniendo en cuenta cuánto tiempo está disponible. Puede definir el ámbito de la sesión mediante un conjunto de temas o preguntas, o puede enumerar un conjunto específico de procesos o casos de uso a explorar. Alinee el alcance de la sesión con el alcance general del proyecto definido en los requisitos del negocio para poder mantener la conversación sobre el tema. El orden del día debe detallar qué temas serán cubiertos, el tiempo disponible para cada tema y los objetivos específicos. Comparta la agenda de la sesión con los interesados con antelación.
- **Preparar recursos:** programe los recursos físicos necesarios, como salas, proyectores, números de teleconferencias y equipos de videoconferencia. Además, programe con los participantes, siendo sensibles a las diferencias de zona horaria si no están todos en la

misma ubicación. Para grupos geográficamente dispersos, cambie el horario cada vez que se reúna para que las sesiones no siempre molesten a las mismas personas en una parte particular del mundo. Recopilar la documentación de varias fuentes. Tener acceso a los sistemas según sea necesario. Tome capacitación en línea para aprender sobre los sistemas existentes.

- **Conozca las partes interesadas:** identifique a los interesados relevantes para la sesión. Conozca las preferencias culturales y regionales de los interesados para las reuniones. Si algunos de los participantes no son hablantes nativos del idioma en el que se llevará a cabo la sesión, considere proporcionarles documentación de apoyo, como diapositivas, antes de tiempo para que puedan leer adelante o seguir adelante. Las diapositivas pueden enumerar preguntas específicas que usted estará preguntando o simplemente proporcionar contexto para la sesión que también podría explicar verbalmente. Evite crear una tensión entre “nosotros” y “ellos”.
- **Preparar las preguntas:** vaya a cada sesión de obtención facilitada con un conjunto de preguntas preparadas. Utilice las áreas de incertidumbre en los modelos de hombre de paja (descritos en la siguiente sección) como fuente de preguntas. Si se está preparando para una entrevista o taller, use los resultados de otras técnicas de obtención para identificar preguntas sin resolver.

Como analista, debe investigar con base en los requisitos qué presentan los clientes para comprender sus verdaderas necesidades. Si pregunta a los usuarios, “¿Qué quieres?” genera una masa de información aleatoria que deja al analista desconcertado. “¿Qué necesitas hacer?” Es una pregunta mejor. Preguntar “por qué” varias veces puede mover la discusión de una solución presentada a una comprensión sólida del problema que necesita ser resuelto. Haga preguntas abiertas para ayudarle a comprender los procesos de negocio actuales de los usuarios y ver cómo el nuevo sistema podría mejorar su rendimiento.

Imagínese aprendiendo el trabajo del usuario, o realizando el trabajo bajo la dirección del usuario. ¿Qué tareas realizaría usted? ¿Qué preguntas haría usted? Otro enfoque consiste en desempeñar el papel de un aprendiz aprendiendo de un usuario maestro. El usuario que está

entrevistando a continuación, guía la discusión y describe lo que él considera como los temas importantes para la discusión.

Explore alrededor de las excepciones: ¿Qué podría impedir al usuario completar satisfactoriamente una tarea? ¿Cómo debe el sistema responder a diversas condiciones de error?

Haga preguntas que comiencen con “¿Qué más podría...,” “Que pasa cuando...,” “¿Alguna vez necesitarás...,” “Donde consigues...,” “¿Por qué usted (o no)...,” “Y” “¿Alguna vez alguien...”. Documente el origen de cada requisito para que pueda obtener más información si es necesario y rastrear las actividades de desarrollo de nuevo a los orígenes específicos del cliente.

Como con cualquier actividad de mejora, la insatisfacción con la situación actual proporciona excelentes argumentos para el nuevo y mejorado estado futuro. Cuando esté trabajando en un proyecto de reemplazo para un sistema heredado, pregunte a los usuarios: “¿Cuáles son las tres cosas que más le molestan en el sistema existente?” Esta pregunta determina las expectativas que los usuarios tienen para el sistema subsiguiente.

Las preguntas preparadas son para ayudarte si te quedas atascado. Las preguntas deben parecer naturales y cómodas, como una conversación, no un interrogatorio. A cinco minutos de una sesión, quizás sepa que se perdió un área importante para la discusión. Esté listo para abandonar sus preguntas si es necesario. Al final de su sesión, pregunte “¿Hay algo más que esperaba que yo le preguntara?” Para tratar de plantear problemas que usted simplemente no pensaba.

- **Preparar modelos de *Straw man*:** los modelos de análisis se pueden utilizar durante las sesiones de inducción para ayudar a los usuarios a proporcionar mejores requisitos. Algunos de los modelos más útiles son los casos de uso y los procesos, porque se alinean estrechamente con la forma en que la gente piensa en hacer su trabajo. Un hombre de paja sirve como punto de partida que le ayuda a aprender sobre el tema e inspira a sus usuarios a pensar en ideas. Es más fácil revisar un modelo de borrador que crear uno desde cero.

Si es nuevo en el dominio del proyecto, puede ser difícil crear un modelo de borrador por su cuenta. Utilice otras técnicas de

obtención para recoger suficiente conocimiento para trabajar. Lea los documentos existentes, examine los sistemas existentes para los modelos que puede reutilizar como punto de partida o realice una entrevista individual con un experto en la materia para aprender lo suficiente para empezar. Luego dígale al grupo con el que está trabajando: "Este modelo probablemente estará equivocado. Por favor decirme cómo debería ser".

3.6. Realizar las actividades de obtención

Consiste en realizar las sesiones de obtención. Se recomienda considerar lo siguiente:

- **Educar a los interesados:** enseñe a los interesados acerca de su enfoque de obtención y por qué lo eligió. Explique las técnicas de exploración que utilizará, como casos de uso o procesos, y cómo pueden ayudar los interesados a proporcionar mejores requisitos. También describa cómo va a capturar su información y enviarles materiales para su revisión después de la sesión.
- **Tomar notas apropiadas:** asigne a alguien que no está participando activamente en la discusión para que sea el escritor responsable de tomar notas. Las notas de la sesión deben contener una lista de asistentes, invitados que no asistieron, decisiones tomadas, acciones a tomar y quién es responsable de cada uno, asuntos pendientes y los puntos altos de discusiones clave. Desafortunadamente, los analistas de negocio a veces mantienen sesiones de obtención facilitadas sin un escritor dedicado y tienen que llenar el papel ellos mismos.

Si se encuentra en esta situación, debe estar preparado para escribir taquigrafía, escribir rápidamente o usar un dispositivo de grabación (si los participantes están de acuerdo). También puede usar pizarras y papel en las paredes y fotografiarlos.

Prepare preguntas con anticipación para eliminar algunos de los pensamientos in situ necesarios para mantener la conversación. Llegue con una notación abreviada para capturar una pregunta que viene a la mente mientras alguien está hablando, así que rápidamente puede volver a ella cuando tenga una oportunidad. No trate de capturar

diagramas complicados mediante un software, basta con dibujar diagramas rápidamente con la mano.

- **Aprovechar el espacio físico:** la mayoría de las habitaciones tienen cuatro paredes, así que úselas durante la facilitación para dibujar diagramas o crear listas. Si no hay pizarras disponibles, coloque grandes hojas de papel en las paredes. Tenga notas adhesivas y marcadores disponibles. Invite a otros participantes a levantarse y contribuir también; Moverse ayuda a mantener a la gente comprometida. En Gottesdiener (2005) se refiere a esta técnica como el patrón de colaboración “Wall of Wonder”. Si existen artefactos a considerar (como modelos de hombre de paja, requisitos o sistemas existentes) proyectelos en la pared.

Facilitar sesiones de colaboración con los participantes en múltiples ubicaciones requiere más creatividad. Puede utilizar herramientas de conferencia en línea para compartir diapositivas y permitir interacciones. Si hay varios participantes en la misma sala, utilice herramientas de videoconferencia para mostrar a los participantes remotos lo que hay en las paredes y pizarras.

3.7. Seguimiento luego de la obtención

Después de completar cada actividad de obtención, aún queda mucho por hacer. Debe organizar y compartir sus notas, documentar problemas abiertos y clasificar la información recién recopilada. Es necesario considerar las siguientes actividades.

- **Organizar y compartir las notas:** si dirigió una entrevista o un taller, la organización de sus notas probablemente requiere más esfuerzo que si organizó la información como la encontró durante una actividad de obtención. Consolide sus aportaciones desde múltiples fuentes. Revise y actualice sus notas poco después de que la sesión esté completa, mientras que el contenido todavía está fresco en su mente. Poco después de cada entrevista o taller, comparta las notas consolidadas con los participantes y pídale que las revisen para asegurarse de que representan con precisión la sesión. La revisión temprana es esencial para el desarrollo exitoso de los requisitos, porque solo las personas que suministraron los requisitos pueden juzgar si fueron capturados correctamente. Mantenga discusiones

adicionales para resolver cualquier inconsistencia y para llenar cualquier espacio en blanco. Considere la posibilidad de compartir las notas consolidadas con otros interesados del proyecto que no estuvieron presentes en la sesión, para que conozcan el progreso. Esto les da la oportunidad de resolver cualquier problema o inquietud inmediatamente.

- **Documentación de problemas abiertos:** durante las actividades de obtención, es posible que haya encontrado elementos que necesitan ser explorados más adelante en una fecha posterior o las brechas de conocimiento que necesita cerrar. O puede que haya identificado nuevas preguntas al revisar sus notas. Examine todos los problemas que todavía están abiertos y registrarlos en una herramienta de seguimiento de problemas. Para cada emisión, registre todas las notas relevantes relacionadas con la resolución de los problemas, el progreso ya realizado, el propietario y la fecha de vencimiento. Considere la posibilidad de utilizar la misma herramienta de seguimiento de problemas que utilizan los equipos de desarrollo y pruebas.



Actividades de aprendizaje recomendadas

Estimado estudiante finalizada la unidad 3, recomiendo reforzar el estudio de los temas correspondientes a esta unidad, mediante el desarrollo de las siguientes actividades.

1. Con base al equipo de desarrollo, desea desarrollar una aplicación que permita a todas las empresas o personas que se dedican a la venta de productos en línea, realizar el seguimiento de entrega de los productos vendidos para que cada cliente conozca en qué momento será entregado el producto. Para ello lo han designado a usted para que realice un cuestionario que se desea aplicar a cada posible cliente del producto. Desarrolle:
 - a. Listado de posibles clientes del producto.
 - b. Objetivo del cuestionario.
 - c. Banco de preguntas.

d. Estrategia de aplicación.

Nota: conteste la actividad en un cuaderno de apuntes o en un documento Word

2. Realice la autoevaluación 3, con el fin de afianzar sus conocimientos:



Autoevaluación 3

1. La obtención incluye actividades para:
 - a. Documentar lo que dicen los usuarios.
 - b. Definir los requisitos funcionales y no funcionales.
 - c. Recolectar, descubrir, extraer y definir requisitos.
2. Preparar la obtención requiere de:
 - a. Realizar las sesiones de obtención.
 - b. Organizar y compartir las notas, y los documentos de los problemas pendientes.
 - c. Determinar el alcance y agenda de la obtención, preparar los recursos, y preparar las preguntas y los modelos.
3. Para que una entrevista sea productiva, se recomienda que debe durar entre:
 - a. 10 a 15 minutos.
 - b. 45 a 60 minutos.
 - c. Mínimo 2 horas.
4. A la reunión de interesados cuidadosamente seleccionados que trabajan bajo la guía de un experto neutral que produce y documenta modelos de requisitos, se conoce como:
 - a. Taller.
 - b. Entrevista.
 - c. Prototipos.
5. Para obtener información general sobre las necesidades de los interesados, es conveniente realizar:
 - a. Entrevista.
 - b. Prototipos.
 - c. Mapa de relación.

6. ¿Qué actividad debería realizarse primeramente en una entrevista?
 - a. Preparar las preguntas de la entrevista.
 - b. Identificar las personas que se entrevistará.
 - c. Realizar la entrevista.
7. En el enfoque pasivo de la observación, el ingeniero de requisitos:
 - a. Se involucra en las tareas, pasando a formar parte del equipo de trabajo.
 - b. Realiza el análisis con base en notas, videos, etc.
 - c. Realiza el análisis con base en documentos.
8. En un taller, la primera actividad que se debe realizar es:
 - a. Identificar las reglas.
 - b. Conducir la reunión.
 - c. Determinar el propósito y los participantes.
9. En las entrevistas estructuradas:
 - a. Se elaboran preguntas acordes a un propósito asociado al entrevistado.
 - b. No se elaboran preguntas.
 - c. Se discuten de temas abiertos.
10. Las disposiciones que emite el SRI para la facturación en línea, en la etapa de obtención se puede catalogar como:
 - a. Requisitos de usuario.
 - b. Fuente externa de Información.
 - c. Fuente interna.

[Ir al solucionario](#)



Semana 8



Actividades finales del bimestre

Han transcurrido 8 semanas de autoestudio y aprendizaje continuo, con el cual hemos cubierto tres unidades planificadas para este primer bimestre con lo cual hemos estudiado temas relacionados con el fundamento e importancia de la ingeniería de requisitos, requisitos de negocio y técnicas de obtención, pero la pregunta es ¿por qué es importante realizar las actividades de la ingeniería de requisitos?, evidentemente que ya conoce todos los conceptos relacionados con la definición de los requisitos y que el desarrollo de cada actividad le permitirá especificar los requisitos con un alto nivel de comprensión, con lo cual su definición será la más adecuada.

Lo relevante del presente curso es de que pueda ir relacionando los conceptos teóricos que se cubren en cada apartado con la práctica a través de los ejemplos y con el desarrollo del caso de estudio, para que pueda adquirir las habilidades necesarias que se persigue con cada resultado de aprendizaje. Es importante analizar cada ejemplo y recurso para desarrollar inicialmente el Análisis de Interesados y luego el visionamiento y alcance del producto, todo esto enmarcado en las actividades que recomienda la IR.

El desarrollo de estos primeros componentes requiere de información, por lo tanto, es fundamental establecer ¿dónde se encuentra la información?, a estos recursos se los conoce como fuentes de información. Estas fuentes, como se indicó y analizó, pueden ser tres: los Interesados, documentos internos y aquellas fuentes externas, y es precisamente con estas fuentes que debe planificar la estrategia de obtención. La correcta planificación y ejecución de la estrategia le permitirá ir conociendo al detalle cada actividad, por ende, definiendo los requisitos.

Espero que haya analizado los ejemplos propuestos y desarrollado las actividades recomendadas, ya que esto contribuirá a su preparación, además del desarrollo de las actividades calificadas, estas son muy importantes porque aportan a la práctica. Desarrolle las autoevaluaciones tantas veces sean necesario, con el objeto de comprender los temas del curso.

Finalmente, reitero su participación en las actividades planificadas y que son de interacción con el docente a través de las tutorías semanales, donde se abordarán temas para reforzar el conocimiento y como desarrollar las actividades calificadas.

Con estas sugerencias e indicaciones finalizamos el primer bimestre, y prepárese para la evaluación del primer bimestre estudiando todos los temas, revise los recursos adicionales que se indican en cada tema y sobre todo las actividades que ha desarrollado.

¡Buena suerte!



Actividades de aprendizaje recomendadas

Revise los ejemplos que se cargarán en la plataforma educativa para reforzar el estudio previo a su evaluación bimestral.



Segundo bimestre

Resultado de aprendizaje 3

- Compara y contrasta las diferentes técnicas de modelado de requisitos.

El resultado de aprendizaje está orientado conocer las estrategias del modelado que permitan, por un lado, tener un conocimiento de la situación actual del negocio, así como determinar los modelos que mejor se adapten a la realidad de la empresa para determinar los requisitos de usuario y sistema.

Contenidos, recursos y actividades de aprendizaje



Semana 9

Unidad 4. Análisis de requisitos

4.1. El análisis de requisitos

Para analizar los requisitos de manera efectiva, debe comprender y definir los requisitos de manera suficiente para que los interesados puedan priorizar sus necesidades y asignar requisitos al *software*. El resultado de este análisis es el modelo de requisitos o también conocido como modelo de análisis. En este modelo los requisitos de usuario se representan mediante diagramas, texto estructurado o una combinación de estos.

Para desarrollar un modelo de requisitos es necesario comprender lo que los usuarios pretenden realizar con el sistema, para ello es necesario adoptar un enfoque centrado en el usuario y en el uso para la obtención de requisitos. Centrarse en los usuarios y su uso anticipado ayuda a revelar la funcionalidad que se requiere, evita la implementación de funciones que nadie usará y ayuda con la priorización. (Wiegers & Beatty, 2013).

El análisis da lugar a un modelo del sistema que pretende ser correcto, completo, coherente e inequívoco. Los desarrolladores formalizan la especificación de los requisitos producida durante la obtención de los requisitos y examinan con más detalle las condiciones de los límites y los casos de excepción. Los desarrolladores validan, corrigen y aclaran la especificación de requisitos si se encuentran errores o ambigüedades. El cliente y el usuario normalmente participan en esta actividad cuando se debe cambiar la especificación de requisitos y cuando se debe recopilar información adicional (Bruegge & H Dutoit, 2012).

Los requisitos obtenidos desde los interesados y articulados usando modelos de análisis necesitan ser lo suficientemente claros y completos para posteriormente validar el proceso de requerimientos de software. El análisis de los requisitos es principalmente responsabilidad del analista, pero puede involucrar a los actores clave, tales como: usuarios, clientes y personal técnico, quienes son necesarios para entender las necesidades del usuario.

- **Facilitar la comunicación entre personal técnico y empresarios:** los modelos permiten que el equipo vea diferentes aspectos de las necesidades del usuario desde diferentes perspectivas.
- **Descubrir requisitos faltantes, erróneos, ambiguos y contradictorios:** los modelos de requisitos se enlazan, lo que permite al equipo dar a conocer los requisitos relacionados e inconsistentes entre modelos. Descubrir y corregir estos errores resulta en requisitos de alta calidad.
- **Hacer que el proceso de desarrollo de requisitos sea más interesante y atractivo para los interesados:** el uso de modelos textuales y visuales ofrece y permite a los interesados entender los requisitos desde diferentes ángulos.
- **Utilice los diferentes modos de pensamiento humano:** algunas personas piensan con más precisión con las palabras, mientras que otras son más capaces de entender los conceptos con los diagramas. Utilice los dos tipos de representación aprovechando los diferentes modelos de pensamiento.

4.2. Modelado de requisitos

Los sistemas de *software* satisfacen funciones y propósitos de negocio específicos. Estos propósitos son articulados por los usuarios, los usuarios o sus representantes son, por lo tanto, una parte integral del proceso de desarrollo. Tal como se ha indicado anteriormente, los usuarios expresan sus necesidades, articulan los escenarios de negocio con lo que se determina la visión del producto. El desafío consiste en capturar, modelar y traducir estas necesidades y requisitos en una solución aceptable. Manejar este desafío de desarrollar un *software* usable que sea aceptable para los usuarios es el núcleo del éxito de un proyecto de *software* (Wiegers & Beatty, 2013).

Idealmente, todos los proyectos de ingeniería de *software* comienzan con una comprensión de los objetivos de negocio, luego emprenden el modelado y análisis de requisitos. Estas actividades ayudan a dar luz y modelar las necesidades y deseos de los usuarios. Al análisis le sigue el diseño de la solución de *software* y la codificación eventual que se basa en el entorno técnico disponible y las capacidades de la organización. Las actividades dentro del análisis y el diseño se llevan a cabo de manera iterativa e incremental.

El diseño proporciona el puente entre el análisis y la codificación. Un buen diseño es un conducto fluido que transforma los requisitos en implementación. Los buenos diseñadores de *software* conocen las actividades de análisis y el modelo de requisitos resultante y están familiarizados con las tecnologías y las limitaciones impuestas por la arquitectura de la organización.

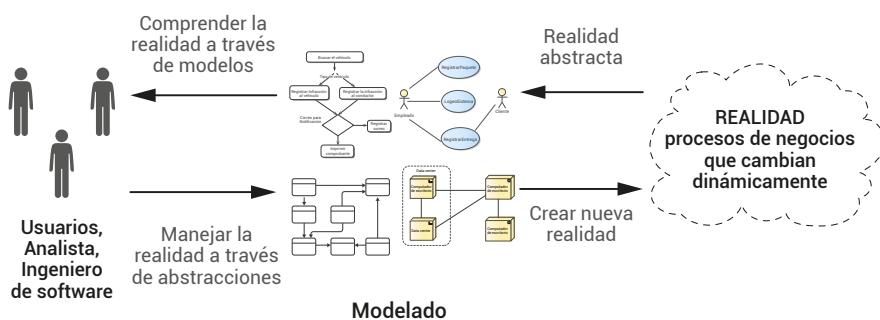
El diseño cierra la brecha entre el análisis y la codificación. Un buen diseño es una canalización fluida que traduce los requisitos en implementación. Los buenos desarrolladores de *software* están familiarizados con las actividades de análisis y el modelo de requisitos resultantes, así como con los métodos y las restricciones impuestas por la arquitectura de la organización.

Dichos modelos mejoran significativamente las comunicaciones dentro y entre los equipos de desarrollo. Esto es así porque estos modelos representan requisitos en el espacio del problema, diseños en el espacio de la solución y restricciones en el espacio arquitectónico. En general, los modelos brindan al equipo del proyecto grandes oportunidades para

identificar brechas, errores de comprensión, desajuste tecnológico y expectativas cambiantes de los usuarios. Los modelos permiten que los equipos hagan todo lo que necesitan hacer antes de comenzar a “codificar”. La codificación, o programación, se convierte así en casi la última y quizás la actividad menos agotadora de todas en un proyecto de ingeniería de software bien organizado y modelado. Por lo tanto, el modelado y las comunicaciones subsiguientes se ven cada vez más como una de las actividades más importantes en la producción de calidad y valor en soluciones de software.

Como lo señaló Alan Davis, ninguna visión de los requisitos proporciona una comprensión completa. Se necesita una combinación de representaciones de requisitos textuales y visuales en diferentes niveles de abstracción para tener una imagen completa del sistema deseado. Las vistas de requerimientos pueden incluir listas de requisitos funcionales, tablas, modelos de análisis visual, prototipos de interfaz de usuario, pruebas de aceptación, árboles de decisión, tablas de decisión, fotografías, videos y expresiones matemáticas (Wiegert & Beatty, 2006).

Figura 18
Importancia del modelado en la Ingeniería de software



Nota. Adaptado de *Software engineering with uml* (p. 4), por Unhelkar, B., 2018, Auerbach Publications.

Los requisitos obtenidos a partir de los interesados y articulados usando modelos de análisis necesitan ser lo suficientemente claros y completos para posteriormente validar el proceso de requerimientos de software. Los modelos de requisitos le ayudarán a:

- **Facilitar la comunicación entre personal técnico y empresarios.**
Los modelos permiten que el equipo vea diferentes aspectos de las necesidades del usuario desde diferentes perspectivas.
- **Descubrir requisitos faltantes, erróneos, ambiguos y contradictorios.**
Los modelos de requisitos se enlazan, lo que permite al equipo dar a conocer los requisitos relacionados e inconsistentes entre modelos. Descubrir y corregir estos errores resulta en requisitos de alta calidad.
- **Hacer que el proceso de desarrollo de requisitos sea más interesante y atractivo para los interesados.** El uso de modelos textuales y visuales ofrece y permite a los interesados entender los requisitos desde diferentes ángulos.
- **Utilice los diferentes modos de pensamiento humano.** Algunas personas piensan con más precisión con las palabras, mientras que otras son más capaces de entender los conceptos con los diagramas. Utilice los dos tipos de representación aprovechando los diferentes modelos de pensamiento.

El análisis de los requisitos es principalmente responsabilidad del BA, pero puede involucrar a los actores clave, tales como: usuarios, clientes y personal técnico quienes son necesarios para entender las necesidades del usuario. Los BA pueden esperar encontrar una técnica que reúna todo en una descripción holística de los requisitos de un sistema. Desafortunadamente, no existe un diagrama que lo abarque todo. De hecho, si pudiera modelar todo el sistema en un solo diagrama, ese diagrama sería tan inutilizable como una larga lista de requisitos por sí solo.

Los modelos de requisitos visuales pueden ayudar a identificar requisitos faltantes, extraños e inconsistentes. Dadas las limitaciones de la memoria humana a corto plazo, es casi imposible analizar una lista de mil requisitos en busca de incoherencias, duplicaciones y requisitos superfluos. Para cuando alcances el decimoquinto requisito, es probable que hayas olvidado los primeros que leíste. Es poco probable que encuentre todos los errores simplemente revisando los requisitos textuales. Entre los modelos de requisitos visuales tenemos:

- Diagramas de Flujo de Datos (DFD).
- Diagramas de flujo de procesos, como diagramas de carriles.
- Diagramas de Transición de Estado (STD) y tablas de estado.

- Mapas de diálogo.
- Tablas de decisión y árboles de decisión.
- Tablas de respuesta a eventos.
- Árboles de características.
- Diagramas de casos de uso.
- Diagramas de actividades.
- Diagramas Entidad-Relación (ERD)

Estos modelos son útiles para elaborar y explorar los requisitos, así como para diseñar soluciones de software. Ya sea que los esté utilizando para el análisis o para el diseño, depende del momento y la intención del modelado. Al utilizarlos para el análisis de requisitos, estos diagramas le permiten modelar el dominio del problema o para crear representaciones conceptuales del nuevo sistema. Representan los aspectos lógicos de los componentes de datos, transacciones y transformaciones, objetos del mundo real y cambios en el estado del sistema del dominio del problema.

Puede basar los modelos en los requisitos textuales para representarlos desde diferentes perspectivas, o puede derivar requisitos funcionales de modelos de alto nivel que se basan en la entrada (aportes) del usuario. Durante el diseño, los modelos representan cómo pretende implementar el sistema: la base de datos real que se creará, las clases de objetos que se instanciarán y los módulos de código que se desarrollarán. Debido a que los diagramas de análisis y diseño usan las mismas notaciones, identifique claramente cada uno que dibuje como un modelo de análisis (los conceptos) o un modelo de diseño (lo que pretende construir).

Las técnicas de modelado de análisis son compatibles con una variedad de herramientas de modelado de negocio, herramientas de gestión de requisitos y herramientas de dibujo como Microsoft Visio. Las herramientas de modelado especializadas brindan varios beneficios sobre las herramientas de dibujo de uso general. En primer lugar, facilitan la mejora de los diagramas a través de la iteración. Casi nunca obtendrá un modelo correcto la primera vez, por lo que la iteración es la clave para el éxito del modelado. Las herramientas también pueden aplicar las reglas para cada método de modelado que admitan. Pueden identificar errores de sintaxis e inconsistencias que las personas que revisan los diagramas pueden no ver. Las herramientas de gestión de requisitos que admiten el modelado le permiten rastrear los requisitos en los modelos. Algunas herramientas vinculan varios diagramas entre sí y con sus requisitos funcionales y de

datos relacionados. El uso de una herramienta con símbolos estándar puede ayudarlo a mantener la coherencia entre los modelos.

Escuchamos argumentos en contra del uso de modelos de requisitos que van desde “Nuestro sistema es demasiado complejo para modelarlo” hasta “Tenemos un cronograma de proyecto ajustado; no hay tiempo para modelar los requisitos”. Un modelo es más simple que el sistema que está modelando. Si no puede manejar la complejidad del modelo, ¿cómo podrá manejar la complejidad del sistema? La creación de la mayoría de los modelos no requiere mucho más tiempo del que dedicaría a escribir las declaraciones de requisitos y analizarlas en busca de problemas. Cualquier tiempo adicional dedicado al uso de modelos de análisis de requisitos debería compensarse con creces detectando errores de requisitos antes de construir el sistema. Los modelos, o partes de modelos, a veces se pueden reutilizar de un proyecto a otro, o al menos servir como punto de partida para la obtención de requisitos en un proyecto posterior.

Existen diferentes puntos de vista para utilizar el modelo adecuado. Gottesdiener (2005) indica que se puede usar una variedad de modelos de requisitos de usuario para analizar los requisitos. Estos modelos se representan para responder a las preguntas de las “4W's + H” (Who? What? When? Why? + How?) (¿Quién? ¿Qué? ¿Cuándo? ¿Por qué?, y ¿Cómo?). En la tabla 5 se define este enfoque.

Tabla 5

Modelos de requisitos basados en el enfoque del usuario

Enfoque de la pregunta	Ejemplo de la pregunta	Modelos de requisitos de usuario para este enfoque
¿Quién?	¿Quiénes son los interesados del proyecto? ¿Quiénes interactúan directamente con el sistema?	Categoría de interesados.
	¿Quién supervisará la interactividad con el sistema?	Mapa de diálogo.

Enfoque de la pregunta	Ejemplo de la pregunta	Modelos de requisitos de usuario para este enfoque
¿Qué?	¿Qué significa los términos de negocio importantes?	Glosario.
	¿Qué funciones de la organización interactúan para compartir información?	Mapa de relaciones (o mapa de negocio).
	¿Qué información o activos entran o salen del sistema?	Diagrama de contexto.
	¿Qué son los elementos de datos estáticos que se deben almacenar y cómo se relacionan?	Modelo de datos.
¿Cuándo?	¿Cuándo el sistema necesita responder o actuar?	
	¿Cuándo se realizan las tareas y cuándo cambia la información?	Diagrama de estado.
¿Por qué?	¿Por qué estamos motivados para hacer cumplir las normas, políticas, reglamentos y legislación?	Políticas de negocio.
	¿Por qué son las decisiones las que influyen en el comportamiento y hacen valer la estructura empresarial?	Reglas de negocio.
	¿Cómo hacer operar los procesos en el negocio para alcanzar los objetivos de negocio?	Mapa de procesos.
¿Cómo?	¿Cómo son las tareas realizadas y en qué secuencia?	Utilice el modelo de casos de uso (complementado o sustituido por escenarios, historias, diagramas de actividad de casos de uso, o diagrama de flujo de datos).

Nota. Adaptado de *¿Qué modelos puedo crear?* (p. 112), por E. Gottesdiener, 2006, GOAL/QPC.

Complementando la tabla 5, una quinta “W – Where” ante todo provee información acerca de los requisitos no funcionales, especialmente aquellos relacionados con el futuro operativo y ambiente de despliegue.

Ciertos modelos son más apropiados para determinar los requisitos de ciertos dominios de negocio. Se escoge el modelo de acuerdo con la pregunta de enfoque que se indica en la tabla 5. (¿Quién? ¿Qué? ¿Cuándo? ¿Por qué? Y ¿Cómo?), que proporcione una potencial comprensión entre los requisitos y el desarrollo del modelo.

Tenga presente que estas son solo directrices. Cada dominio es diferente, por lo que debe determinar qué modelos son los más útiles en el desarrollo de un subconjunto de modelos en forma preliminar y validación de ellos,

y luego ajustar sus selecciones. No es necesario usar todos los modelos, puede escoger un subconjunto que sea adecuado al dominio del problema. Ahorre tiempo a los interesados elaborando unos modelos de alto nivel, y luego consulte si son útiles.

Figura 19
Procesos “As-is” y “To-be”



Nota. Sucunuta, M., 2023

Sea claro si cada modelo representa la situación “as-is” o “to-be”, tal como se indica en la figura 19. Cuando el proceso actual, datos o sistema no es bien entendido, primero cree uno o más modelos “as-is”. Evite paralizar el análisis elaborando la situación actual como un ámbito de alto nivel, sea solo lo suficiente claro como para comprender el entorno actual, al mismo tiempo que se garantice que los requisitos importantes satisfechos por el sistema actual también sean incluidos en el nuevo sistema. El BA al escuchar atentamente a los clientes, puede seleccionar palabras clave que se traduzcan en elementos específicos del modelo.

Tabla 6*Relacionar la voz del cliente con los componentes del modelo de análisis*

Palabra	Ejemplo	Componentes de modelo de análisis
Sustantivo	Personas, organizaciones, sistemas de <i>software</i> , datos u objetos.	<ul style="list-style-type: none"> ▪ Entidades externas, almacenes de datos, o flujo de datos (Diagrama de flujo de datos). ▪ Actores (diagramas de casos de uso). ▪ Entidades o sus atributos (Diagrama entidad-relación). ▪ Canales (Mapa de procesos). ▪ Objetos con estados (Diagrama de estado-transición).
Verbo	Acciones, cosas que un usuario o un sistema puede hacer o eventos que puede darse.	<ul style="list-style-type: none"> ▪ Procesos (Diagrama de flujo de datos). ▪ Pasos del proceso (Mapa de procesos). ▪ Casos de uso (Diagrama de casos de uso). ▪ Relaciones (Diagrama entidad-relación). ▪ Transición (Diagrama de estado-transición). ▪ Actividades (Diagrama de actividades). ▪ Eventos (Tabla evento-respuesta).
Condicional		<ul style="list-style-type: none"> ▪ Decisión (árbol de decisión, tabla de decisión, o diagrama de actividad). ▪ Derivación (Mapa de procesos o diagrama de actividad).

Nota. Adaptado de *Relacionando la voz del cliente con los componentes del modelo de análisis* (p. 224), por K. Wieggers, 2013, Microsoft Press.

En la tabla 6 se encuentran los componentes que podría definir con base en palabras elegidas por los clientes. A medida que evolucione el aporte del cliente en requisitos y modelos escritos, debería poder vincular cada componente del modelo a un requisito de usuario específico (Wieggers & Beatty, 2013).



Ejemplo: considerando el caso de “servicios de encomiendas”, considere el siguiente párrafo sobre las necesidades del usuario, proporcionadas por el asistente. Los sustantivos únicos significativos están resaltados en negrita, los verbos están en cursiva y resaltados en negrita y las declaraciones condicionales están en negrita, cursiva y subrayado.

El **cliente** en la oficina solicita el servicio para el envío de la encomienda. Para ello el **asistente** debe **calcular** el costo del servicio con base en el peso, dimensiones y contenido. Se requiere **registrar** los datos del cliente así como la dirección de envío. **Si el cliente es una persona natural** se registra los datos

personales del cliente, pero **si es una empresa**, debe registrar los datos de la empresa. El cliente puede pagar el costo del servicio en efectivo, tarjeta de crédito y transferencia. **Si el cliente es una empresa**, tiene la posibilidad de acogerse al pago al final de mes por todas las encomiendas realizadas durante el mes. Se requiere **generar** la factura y guía que se entregará al cliente como comprobante de haber enviado la encomienda.

Los términos identificados en este párrafo serán los que determinen el modelo a desarrollar. Evidentemente que los modelos requieren de una mayor cantidad de información de la contenida en el párrafo.

En el [anexo 3 modelos de análisis](#), se describen cada uno de los modelos que se listan anteriormente. Revise detenidamente la manera en que estos aportan a una mejor comprensión de la información.



Semana 10

4.3. Los diagramas UML

El Lenguaje de Modelado Unificado (UML, por sus siglas en inglés), es un lenguaje de modelado estandarizado que consta de un conjunto integrado de diagramas, desarrollado para ayudar a los desarrolladores de sistemas y software a especificar, visualizar, construir y documentar los artefactos de los sistemas de software, así como para el modelado de negocios y otros sistemas que no son de software. UML representa una colección de las mejores prácticas de ingeniería que han demostrado ser exitosas en el modelado de grandes y complejos sistemas. UML es una parte muy importante en el desarrollo de software orientado a objetos y del proceso de desarrollo de software. El uso de UML ayuda a los equipos de proyectos a comunicarse, explorar potenciales diseños y validar el diseño arquitectónico del software (Unhelkar, 2018).

Comprender los distintos diagramas UML es parte integral de la Ingeniería de Software, esto se debe a que estos diagramas UML presentan un conjunto de artefactos de modelado que son un estándar globalmente aceptado. El conocimiento y la comprensión de los diagramas proporcionan los medios y el lenguaje para que los ingenieros de software esbozen y

visualicen sus pensamientos, así como para discutir, debatir, cuestionar, comunicar y medir su trabajo, particularmente en un equipo de proyectos.

Los diagramas UML casi nunca se usan todos juntos por una sola persona. Cada diagrama tiene un propósito específico, que los modeladores deben entender. La naturaleza específica y el propósito de un diagrama dictan cómo y dónde se usa en el modelado. Por ejemplo, algunos diagramas proporcionan una excelente manera de comprender los requisitos y el comportamiento de un sistema (por ejemplo, diagramas de actividades y casos de uso). Otros diagramas proporcionan un mecanismo robusto para modelar el almacenamiento de datos (por ejemplo, diagramas de clase). Y otro conjunto de diagramas UML ayuda a visualizar la arquitectura del software (por ejemplo, diagramas de componentes y de implementación).

A continuación, revise la infografía denominada Lista de 14 diagramas UML.

[Lista de 14 diagramas UML](#)

La infografía Lista de 14 diagramas UML que se podrían utilizar en un modelo de requisitos. Algunos de estos diagramas tienen dependencias entre sí que son importantes tanto desde la perspectiva del modelado sintáctico como semántico. Estos diagramas, y los artefactos dentro de ellos, permiten visualizaciones de varios aspectos de un sistema de software. Los diagramas se amplían aún más con las especificaciones y la documentación correspondientes.

Los analistas han empleado durante mucho tiempo los escenarios de uso para obtener los requisitos de los usuarios. La perspectiva centrada en el uso se formalizó en el enfoque del caso de uso para modelar los requisitos. Más recientemente, los defensores del desarrollo ágil introdujeron el concepto de “historia del usuario”, una declaración concisa que articula una necesidad del usuario y sirve como punto de partida para las conversaciones al detalle. El objetivo es desarrollar los requisitos con la suficiente calidad y detalle, de tal manera que los gerentes o directivos puedan construir estimaciones realistas del proyecto y el personal técnico pueda proceder con el diseño, construcción y pruebas.



A manera de resumen le recomiendo revisar el recurso educativo abierto [Tema 1 – Introducción a UML](#).

Una vez revisado el recurso, podrá comprender la importancia del modelado en el análisis y diseño de sistemas software. Inicialmente como estrategia para conocer el estado actual y luego en base a criterios modelar el sistema a desarrollar. Lo importante que resulta un modelo de requisitos basado en casos de uso, y cómo se genera un modelo físico como lo es el diagrama de clases.

4.4. La interfaz de usuario y los requisitos

Incluir diseños de interfaz de usuario en el ERS tiene ventajas e inconvenientes. En el lado positivo, explorar posibles interfaces de usuario con prototipos en papel, maquetas de trabajo, estructuras alámbricas o herramientas de simulación hace que los requisitos sean tangibles tanto para los usuarios como para los desarrolladores. Son técnicas poderosas para obtener y validar requisitos. Si los usuarios del producto tienen expectativas de cómo se verán las partes del producto y, por lo tanto, podrían sentirse decepcionados si no se cumplieran sus expectativas, esas expectativas pertenecen al ámbito de los requisitos.

En el lado negativo, las imágenes de pantalla y las arquitecturas de interfaz de usuario describen soluciones y pueden no ser realmente requisitos. Incluirlos en el ERS hace que el documento sea más grande y los documentos de grandes requisitos asustan a algunas personas. Retrasar la línea de base del ERS hasta que se complete el diseño de la interfaz de usuario puede ralentizar el desarrollo y poner a prueba la paciencia de las personas que ya están preocupadas por dedicar demasiado tiempo a los requisitos. Incluir el diseño de la interfaz de usuario en los requisitos puede dar lugar a que el diseño visual impulse los requisitos, lo que a menudo conduce a lagunas funcionales. Las personas que escriben los requisitos no necesariamente están bien calificadas para diseñar interfaces de usuario.

Los diseños de pantalla no reemplazan los requisitos funcionales y de usuario escritos. No espere que los desarrolladores deduzcan la funcionalidad subyacente y las relaciones de datos de las capturas de pantalla. Una empresa de desarrollo de Internet se metió repetidamente en problemas porque el equipo pasaba directamente de firmar un contrato con un cliente a un taller de diseño visual de ocho horas. Nunca entendieron suficientemente lo que un usuario podría hacer en cada sitio web que

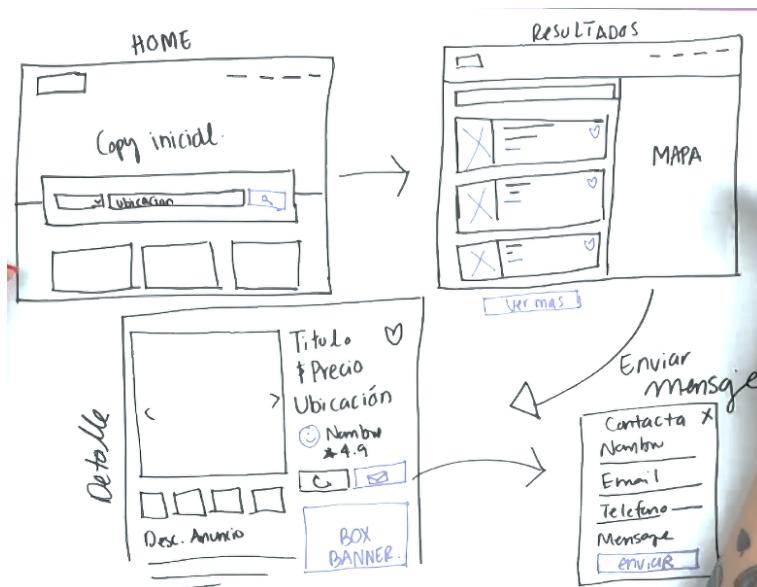
construyeron, por lo que dedicaron mucho tiempo a arreglar los sitios después de la entrega.

Si realmente desea implementar cierta funcionalidad con controles de interfaz de usuario y diseños de pantalla específicos, es apropiado e importante incluir esa información en el ERS como restricciones de diseño. Si el ERS está especificando una mejora a un sistema existente, a menudo tiene sentido incluir pantallas exactamente cómo se van a implementar. Los desarrolladores ya están limitados por la realidad actual del sistema existente, por lo que es posible saber de antemano cómo deberían verse las pantallas modificadas, y quizás también las nuevas.

Un equilibrio sensato es incluir imágenes conceptuales de pantallas seleccionadas en los requisitos sin exigir que la implementación siga con precisión esos modelos. En la figura 20 verá un bosquejo de página web de muestra. La incorporación de dichos bocetos en el ERS comunica de manera útil otra vista de los requisitos, pero deja en claro que los bocetos no son los diseños de pantalla comprometidos. Por ejemplo, un boceto preliminar de un cuadro de diálogo complejo ilustrará la intención detrás de un grupo de requisitos, pero un diseñador visual podría convertirlo en un cuadro de diálogo con pestañas para mejorar la usabilidad.

Figura 20

Ejemplo de un boceto de interfaz de usuario



Nota. Sucunuta, M.. 2023

En la figura 20 se presenta un boceto de interfaz de usuario que se puede utilizar para entender la forma en que se presentará y pedirá la información.

Los equipos que trabajan en proyectos que tienen muchas pantallas pueden encontrar más manejable documentar los detalles del diseño de la interfaz de usuario en una especificación de interfaz de usuario separada o mediante el uso de herramientas de diseño de interfaz de usuario o herramientas de creación de prototipos.

Revise [Mini Bootcamp técnicas ninja para la creación de interfaces en figma](#), para conocer de forma breve el uso de la herramienta [figma](#).

Como puede apreciar, existen varias posibilidades para crear una interfaz que le permita realizar diseños con base en la información recopilada y que puede ser aprovechada para aclarar ciertas dudas. Es preciso indicar que esto puede derivar posteriormente como una estrategia de validación (se verá más adelante en la unidad de validando los requisitos).



Semana 11

4.5. Modelo de casos de uso

El modelado de casos de uso juega un papel importante en los sistemas de software. Los casos de uso se basan en los usuarios (actores) y su propósito (objetivos) al usar el sistema. Los casos de uso documentan los requisitos desde la perspectiva de un usuario, de ahí su nombre. El caso de uso describe la secuencia de interacciones entre el sistema y un actor externo que da como resultado que el actor pueda lograr algún resultado de valor. Los nombres de los casos de uso siempre se escriben en forma de verbo seguido de un objeto. Es preciso que seleccione nombres fuertes y descriptivos para que sea evidente que a partir del nombre que el caso de uso se entregará algo valioso para el usuario (Unhelkar, 2018).

4.5.1. Actor

El modelado de casos de uso comienza con la identificación y documentación de los actores. El objetivo principal de desarrollar una solución de software es satisfacer las necesidades de estos actores (usuarios). El actor también indica cómo se utilizará el sistema, además

de proporcionar el punto de partida para el resto del modelado, diseño y desarrollo en un proyecto de software.

Un actor lo desempeña una persona externa al sistema de software, que interactúa con el sistema para lograr objetivos de negocio, por lo tanto, un actor es:

- Un rol desempeñado por un usuario típico del sistema (el actor es el rol y no la persona real que está desempeñando ese rol).
- Un rol que inicia una interacción con el sistema.
- Un rol que obtiene beneficios (logra metas) del sistema.
- Un “sistema externo” con el que interactuará el sistema en desarrollo (como una base de datos o un servicio disponible públicamente).
- Un dispositivo externo con el que interactuará el sistema en desarrollo (como una impresora o un móvil).
- Cualquier elemento que envíe un mensaje al sistema (como una entidad externa).
- Cualquier elemento que reciba un mensaje del sistema (como otro sistema).

Un actor puede participar en múltiples casos de uso porque cada actor es capaz de iniciar múltiples procesos dentro del sistema y tiene múltiples objetivos que lograr desde el sistema.

Encontrar buenos actores es la primera y más importante actividad durante el análisis en el espacio del problema. Durante la iteración inicial del modelado de casos de uso, se crea una lista de actores. No intente completar esta lista en el primer intento. La identificación y documentación de casos de uso, el dibujo de diagramas de actividad y la subsiguiente identificación de clases conducirán al refinamiento de esta lista de actores.

La identificación de los potenciales actores y casos de uso ocurre en un entorno de taller. Las siguientes son algunas de las preguntas que se pueden hacer en un taller de modelado de casos de uso para llegar a una lista preliminar de actores:

- ¿Quiénes serán los usuarios principales y secundarios del sistema?
- ¿Quiénes serán los principales beneficiarios de las interacciones con el sistema?
- ¿Quiénes serán los principales iniciadores de las interacciones con el sistema?

- ¿Con qué sistemas y dispositivos externos necesitará interactuar el sistema en desarrollo?
- ¿Hay un proceso basado en el tiempo en el sistema?

4.5.2. Caso de uso

Un caso de uso documenta una serie de interacciones de un actor con un sistema. Esta interacción está destinada a proporcionar algunos resultados concretos y medibles de valor para el actor. Los casos de uso describen lo que hace un sistema, pero no especifican cómo lo hace. Además, los casos de uso no solo documentan las interacciones del actor-sistema a través de una serie de pasos, sino que también agregan detalles como condiciones previas y posteriores para el caso de uso, referencias de interfaz de usuario y flujos alternativos.

La lista inicial de actores es un buen punto de partida para la identificación de casos de uso. Los casos de uso se descubren mejor en los mismos talleres en los que se descubren los actores. La documentación del actor también conduce al descubrimiento de casos de uso. Esto se debe a que la documentación del actor proporciona información sobre las relaciones entre el actor y el caso de uso. Los casos de uso se obtienen de la siguiente manera:

- Entrevistas y debates con usuarios y expertos en el dominio en un taller.
- Actuación de varios escenarios o “historias” contadas por los usuarios en términos de cómo usarían el sistema.
- Identificar y documentar a los actores, lo que lleva a una comprensión de sus objetivos o propósitos al usar el sistema.
- Revisión de la salida del análisis de requisitos.
- Declaraciones de problemas formales e informales.
- Ejecutar sistemas existentes (especialmente aplicaciones heredadas), si están disponibles.
- Investigar la documentación del usuario existente, si está disponible.
- Investigar la “ayuda” existente para el sistema, si está disponible.
- Investigar el dominio del problema, para obtener modelos de análisis relevantes.
- Investigar y usar literatura publicada como patrones de análisis.

Identificados los actores y los casos de uso, el modelado se empieza elaborando el diagrama de casos de uso. Los diagramas de casos de uso

proporcionan una representación visual de alto nivel de los requisitos del usuario. En la Tabla 7 se indican todos los elementos que se utilizan para desarrollar un diagrama de casos de uso.

Tabla 7

Elementos de un diagrama de casos de uso

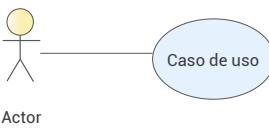


Actor

Actor: son personas o procesos que necesitan interactuar con el sistema. Se deben identificar sus papeles en el sistema. En el diagrama, se representan del siguiente modo:

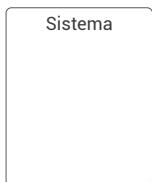


Caso de uso: los casos de uso se representan mediante elipses y corresponden a acciones generales del sistema. Se representa de la siguiente manera:

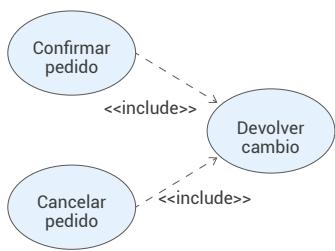


Actor

Asociación: la interacción entre los actores y los casos de uso del sistema se representan por una línea recta que une a ambos.



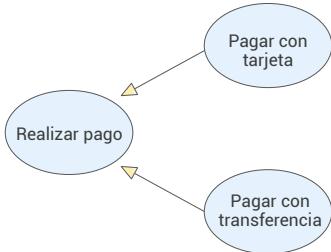
Sistema: el sistema es el software que vamos a desarrollar. Puede ser un pequeño componente cuyos actores son otros componentes, o puede ser una aplicación completa. Se representa como una caja rectangular. Dentro de ella se incluyen los casos de uso soportados por el sistema.



Inclusión: se utiliza cuando el comportamiento de un caso de uso se incluye dentro del comportamiento de otro. Se representa con una flecha de trazo discontinuo desde el caso que incluye hasta el caso incluido, con el estereotipo «include» o «use». Los casos de uso incluidos se pueden compartir, así evitamos repetirlos. También se pueden utilizar para estructurar el diagrama en varios niveles de detalle, pero no conviene abusar de ellos. Observe en la figura, tanto el caso de uso "Confirmar pedido" y "Cancelar pedido", comparten el caso de uso "Devolver cambio".



Extensión: se utiliza cuando un caso además aporta un comportamiento adicional en determinadas circunstancias o cuando se cumple cierta condición. Se representa con una flecha de trazo discontinuo que apunta al caso que queremos extender, y el estereotipo «extend». Observe la figura, el caso de uso "Registrarse" es una extensión del caso de uso "Acceder".



Generalización: se utiliza para expresar que un caso de uso especializado es una forma particular de conseguir los objetivos de otro caso de uso más general. Se representa como una flecha de línea continua acabada en punta triangular hueca que apunta al caso más general.

Nota. Sucunuta, M., 2023

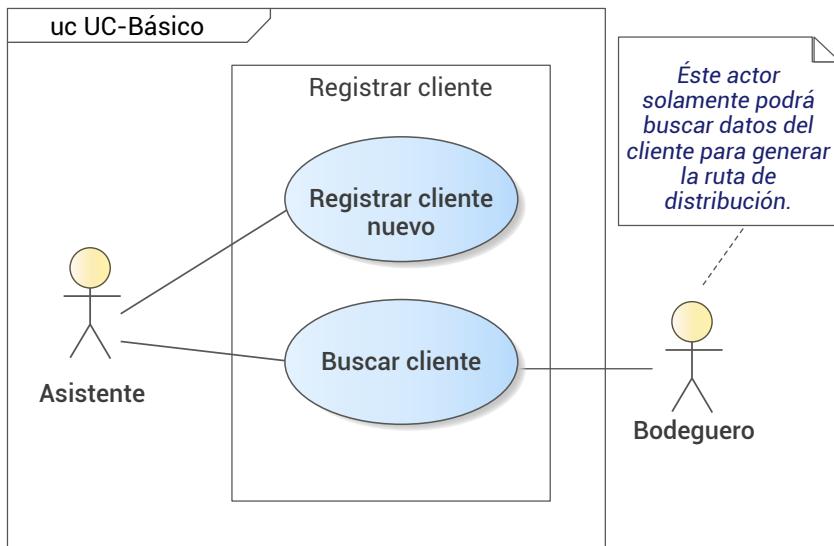
Los casos de uso dentro de los diagramas no se pueden descomponer de la misma manera que se descomponen los diagramas de flujo de datos (DFD). No hay capas o niveles de diagramas de casos de uso, todos están en el mismo nivel dentro de todo el modelo de requisitos. La documentación de los casos de uso es una rica fuente para identificar entidades de negocios que eventualmente resultan en clases.

En la figura 21 se muestra un diagrama de caso de uso, donde se identifica el marco de la caja que representa el límite del sistema. Las líneas de cada actor se conectan a los casos de uso (óvalos) con los que interactúa

el actor. Una línea de un actor a un caso de uso indica que él es el actor principal del caso de uso. El actor principal inicia el caso de uso y obtiene el valor principal de él. Una línea también va de un caso de uso a un actor secundario, que participa de alguna manera en la ejecución exitosa del caso de uso. Otros sistemas de software a menudo sirven como actores secundarios, contribuyendo entre bastidores a la ejecución del caso de uso.

Figura 21

Ejemplo de un diagrama de caso de uso



Nota. Sucunuta, M., 2023

La figura 21 muestra un diagrama de casos de uso simple que contiene actores, casos de uso y sus relaciones. Específicamente, este diagrama muestra dos actores: asistente y bodeguero. También hay dos casos de uso llamados “registrar cliente nuevo” y “buscar cliente” en el diagrama. Estos actores y los casos de uso tienen documentación asociada, que no se muestra en el diagrama. Además, las líneas que conectan a los actores con los casos de uso simplemente indican una asociación o comunicación y no una dependencia o un flujo de información. Para aclarar aún más el diagrama y hacer que el proceso previsto sea legible, siempre es útil proporcionar anotaciones, descripciones y notas adicionales.

Como se ha manifestado anteriormente, un caso de uso describe una actividad discreta e independiente que un actor puede realizar para lograr algún resultado de valor. Un caso de uso puede abarcar una serie de actividades relacionadas que tienen un objetivo común. Un escenario es una descripción de una única instancia de uso del sistema. Por lo tanto, un caso de uso es una colección de escenarios de uso relacionados, y un escenario es una instancia específica de un caso de uso. Al explorar los requisitos del usuario, puede comenzar con una declaración de caso de uso general y desarrollar escenarios de uso más específicos, o puede generalizar desde un ejemplo de escenario específico al caso de uso más amplio.

Para documentar al detalle estos escenarios se precisa hacerlo mediante una plantilla, tal como se muestra en la figura 22. La plantilla es simplemente una estructura en la cual se almacena la información que encuentre durante una discusión de caso de uso de una manera organizada y consistente. La plantilla le recuerda toda la información que debe contemplar con respecto a cada caso de uso. Si la información que pertenece a la plantilla ya existe en otro lugar, simplemente incluya la referencia. Por ejemplo, no incorporar el texto real de cada regla de negocio que afecta el caso de uso en la plantilla, solamente liste los identificadores de las reglas de negocio relevantes para que el lector pueda encontrar esa información cuando sea necesario.

Figura 22

Plantilla para documentar y describir un caso de uso.

ID Nombre			
Creado por		Fecha de creación	
Actor principal		Actores secundarios	
Descripción			
Disparador			
Precondiciones			
Postcondiciones			
Flujo normal	1. Inicio 2. 3. 4. FA-01 Flujo alterno 1 5. 6. Exc - 01 7.		
Flujos alternos	FA-01 Flujo alterno 1. 1. 2. 3 FA-01 Flujo alterno 2. 1. 2. 3.		
Excepciones	Exc-01: Excepción 1 1. 2. 3.		
Prioridad			
Frecuencia de uso			
Reglas de negocio			
Otra información			
Suposiciones			

Nota. Adaptado de *Especificación parcial de un caso de uso* (p. 150), por K. Wiegers y J. Beatty, 2013, Microsoft Press.

Los elementos esenciales de un caso de uso son los siguientes:

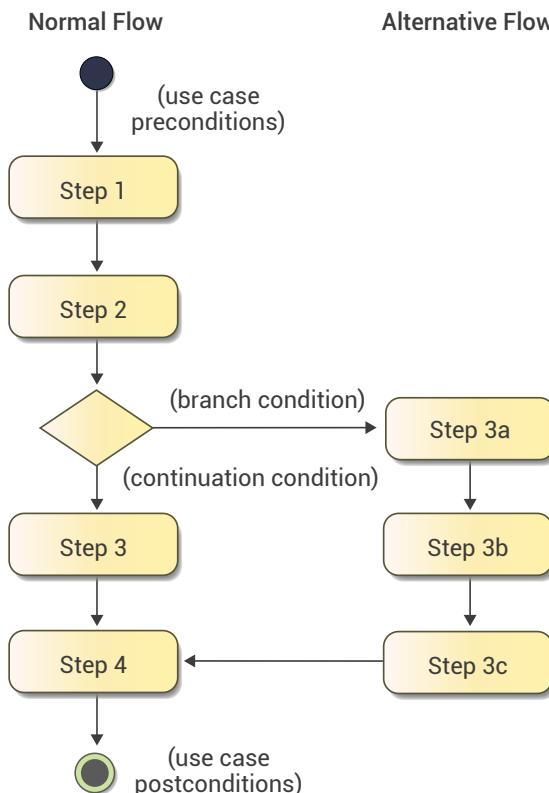
- Un identificador único y un nombre sucinto que indica el objetivo del usuario.
- Una breve descripción textual que describe el propósito del caso de uso.
- Una condición de disparo que inicia la ejecución del caso de uso.
- Cero o más condiciones previas que deben satisfacerse antes de que el caso de uso pueda comenzar.
- Una o más post-condiciones que describen el estado del sistema después de que el caso de uso se ha completado con éxito.
- Una lista numerada de pasos que muestra la secuencia de interacciones entre el actor y el sistema -un diálogo- que conduce desde las precondiciones a las postcondiciones.

Una especial atención requiere los escenarios, que se identifica en:

- El flujo normal de eventos para el caso de uso (también se denomina flujo principal, flujo básico, curso normal, escenario principal, escenario de éxito principal, escenario de día soleado y camino feliz).
- Los flujos alternativos o escenarios secundarios. Los flujos alternativos brindan el mismo resultado de negocio (a veces con variaciones) que el flujo normal, pero representan variaciones menos comunes o de menor prioridad en los detalles de la tarea o cómo se lleva a cabo. El flujo normal puede ramificarse en un flujo alternativo en algún punto de decisión en la secuencia de diálogo; podría (o no) volver a unirse al flujo normal más adelante. Los pasos en el flujo normal indican dónde el usuario puede bifurcarse en un flujo alternativo.

Figura 23

Diagrama de actividad - Secuencia de pasos del flujo normal y alterno de un caso de uso.



Nota. Adaptado de *Diagrama de actividad que ilustra la secuencia de pasos en los flujos normal y alternativo de un caso de uso* (p. 154), por K. Wiegers y J. Beatty, 2013, Microsoft Press.

Aunque muchos casos de uso se pueden describir en prosa simple, es preciso que los escenarios describan la interacción entre el actor y el caso de uso. En la figura 23 se presenta un diagrama de flujo para representar visualmente el flujo lógico en un caso de uso. El diagrama muestra los puntos de decisión y las condiciones que provocan una bifurcación del flujo normal a un flujo alternativo.

Tabla 8*Especificación del caso de uso “Solicitar un producto químico”*

ID Nombre	UC-04 Solicitar un producto químico		
Creado por	Lori	Fecha de creación	12/12/2018
Actor principal	Solicitante	Actores secundarios	Comprador. Almacén químico.
Descripción	El solicitante especifica el producto químico que desea al ingresar su nombre o número de identificación del producto químico o al importar su estructura desde una herramienta. El sistema ofrece al solicitante un contenedor del producto químico del almacén químico o le permite al solicitante que lo solicite a un proveedor.		
Disparador	El solicitante indica que él quiere solicitar un producto químico.		
Precondiciones	PRE-01: la identidad del usuario ha sido autenticada. PRE-02: el usuario ha sido autorizado para solicitar un producto químico. PRE-03: la base de datos del inventario está en línea.		
Postcondiciones	POST-01: la solicitud está guardada en el sistema. POST-02: la solicitud ha sido enviada al almacén químico o a un comprador.		
Flujo normal	<p>4.0 Solicitar un producto químico al almacén químico.</p> <ol style="list-style-type: none"> 1. El solicitante especifica el producto químico deseado. 2. El sistema lista los recipientes del producto químico deseado que se encuentran en el almacén de productos químicos, si corresponde. 3. El sistema le da al solicitante la opción de Ver el Historial de Contenedores para cualquier contenedor. (FA: 3.1). 4. El solicitante selecciona un contenedor específico o solicita realizar un pedido de proveedor (FA: 4.1), (FA: 4.2). 5. El solicitante ingresa otra información para completar la solicitud. (E: 5.1). 6. El sistema almacena la solicitud y notifica al almacén de productos químicos. 		

ID Nombre	UC-04 Solicitar un producto químico
Flujos alternos	<p>4.1 Solicitar un producto químico a un proveedor.</p> <ol style="list-style-type: none"> 1. El solicitante busca en los catálogos de proveedores el producto químico (E: 4.1.E1). 2. El sistema muestra una lista de proveedores para el producto químico con el tamaño, los grados y los precios de los contenedores disponibles. 3. El solicitante selecciona un proveedor, el tamaño del contenedor, los grados y la cantidad de contenedores. 4. El solicitante ingresa otra información para completar la solicitud. 5. El sistema almacena la solicitud y notifica al comprador.
Excepciones	<p>4.1.E1 El producto químico no está disponible comercialmente.</p> <ol style="list-style-type: none"> 1. El sistema presenta el mensaje: no hay proveedores para ese producto químico. 2. El sistema le pregunta al solicitante si desea solicitar otro producto químico (3a) o salir (4a). <ul style="list-style-type: none"> 3a. El solicitante responde pedir otro producto químico. 3b. El sistema comienza el flujo normal de nuevo. 4a. Solicitante pide salir. 4b. El sistema termina caso de uso.
Prioridad	Alta
Frecuencia de uso	Aproximadamente 5 veces por semana por cada producto químico, 200 veces por semana por el personal de almacén químico.
Reglas de negocio	RNE.28, RNE.31
Otra información	El sistema debe poder importar una estructura química en la forma codificada estándar desde cualquiera de los paquetes de dibujos químicos compatibles.
Suposiciones	Se asume que las estructuras químicas importadas son válidas.

Nota. Adaptado de Especificación parcial del caso de uso “Solicitar un producto químico” del sistema de seguimiento de productos químicos, (p. 150). K. Wiegers, 2013. Microsoft Press.

En la tabla 8, se muestra la descripción del caso de uso “solicitar un producto químico”, donde se describe toda la información que requiere

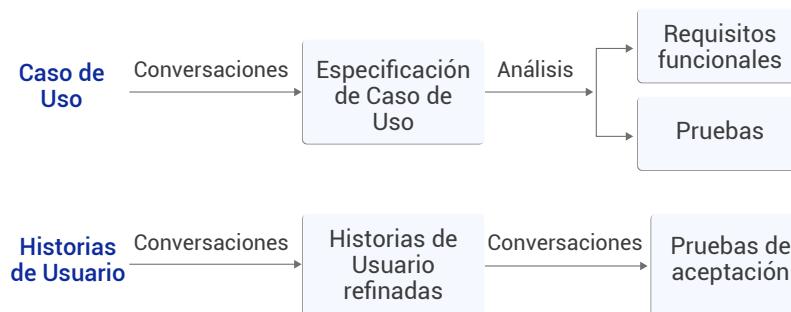
la plantilla, esto se logra previo análisis de la información obtenida de los Interesados, en este caso del “solicitante”. Es preciso recalcar el escenario que se describe tanto en el flujo normal, alterno y excepciones, resaltando la interactividad entre el actor y el sistema.

4.6. Modelo ágil

Los casos de uso se parecen mucho a las historias de usuarios. Ambos se centran en comprender qué necesitan lograr los diferentes tipos de usuarios a través de las interacciones con un sistema de software. Sin embargo, los dos procesos se mueven en diferentes direcciones desde un punto de partida similar, como se ilustra en la figura 24.

Figura 24

Enfoque de los Casos de Uso y las Historias de Usuario.



Nota. Adaptado de *Cómo los requisitos del usuario conducen a requisitos y pruebas funcionales con el enfoque de caso de uso y el enfoque de historia de usuario* (p. 146), por K. Wiegers y J. Beatty, 2013, Microsoft Press.

En los proyectos ágiles, una historia de usuario sirve como marcador de posición para futuras conversaciones que se dan entre desarrolladores, representantes de clientes y un BA. Estas conversaciones revelan la información adicional que los desarrolladores deben saber para poder implementar la historia. Refinar las historias de los usuarios a través de conversaciones conduce a una colección de historias más pequeñas y enfocadas que describen partes individuales de la funcionalidad del sistema. Las historias de usuario que son demasiado grandes para

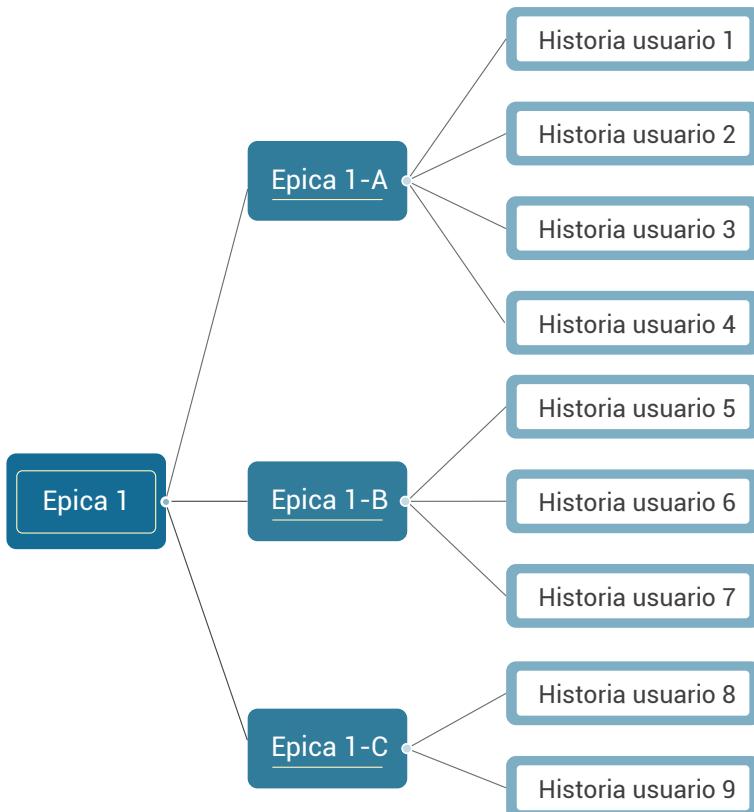
implementarlas en una iteración de desarrollo ágil (llamadas épicas) se dividen en historias más pequeñas que se pueden implementar en una sola iteración.

Los métodos de desarrollo ágil intentan abordar algunas limitaciones del modelo en cascada. Los métodos ágiles se centran en el desarrollo iterativo e incremental, dividiendo el desarrollo de software en ciclos cortos llamados iteraciones (o, en el método ágil conocido como Scrum, "sprints"). Las iteraciones pueden ser tan cortas como una semana o tan largas como un mes. Durante cada iteración, el equipo de desarrollo agrega un pequeño conjunto de funcionalidades en función de las prioridades establecidas por el cliente, lo prueba para asegurarse de que funciona correctamente y lo valida con los criterios de aceptación establecidos por el cliente. Los incrementos posteriores modifican lo que ya existe, enriquecen las características iniciales, agregan otras nuevas y corrigen los defectos que fueron descubiertos. La participación continua de los clientes permite que el equipo detecte problemas y cambios de dirección con anticipación, lo que guía a los desarrolladores a ajustar su rumbo antes de que estén demasiado lejos. El objetivo es tener una parte de software potencialmente entregable al final de cada iteración, incluso si constituye solo una pequeña porción del producto deseado en última instancia.

Muchas de las prácticas de requisitos aplicadas en proyectos ágiles también funcionan bien, y son una buena idea para, proyectos que siguen cualquier otro ciclo de vida de desarrollo. Entre ellas:

- Involucramiento del cliente.
- Documentación con menos detalle que los desarrollos tradicionales.
- Desarrollo del *backlog* priorizado.
- Requisitos de alto nivel en forma de Historias de usuario.
- Descomposición.
- Esperar cambios.
- Retroalimentación.

Figura 25
Enfoque ágil.



Nota. Sucunuta, M., 2023

Debido a que las épicas abarcan iteraciones, deben dividirse en conjuntos de historias más pequeñas. A veces, las épicas son lo suficientemente grandes como para que se deban subdividir en múltiples épicas, cada una de las cuales se divide luego en múltiples historias hasta que cada historia resultante pueda estimarse de manera confiable y luego implementarse y probarse en una sola iteración tal como se observa en la figura 25. Dividir las épicas en épicas más pequeñas y luego en historias de usuarios a menudo se denomina descomposición de historias (IIBA, 2015).

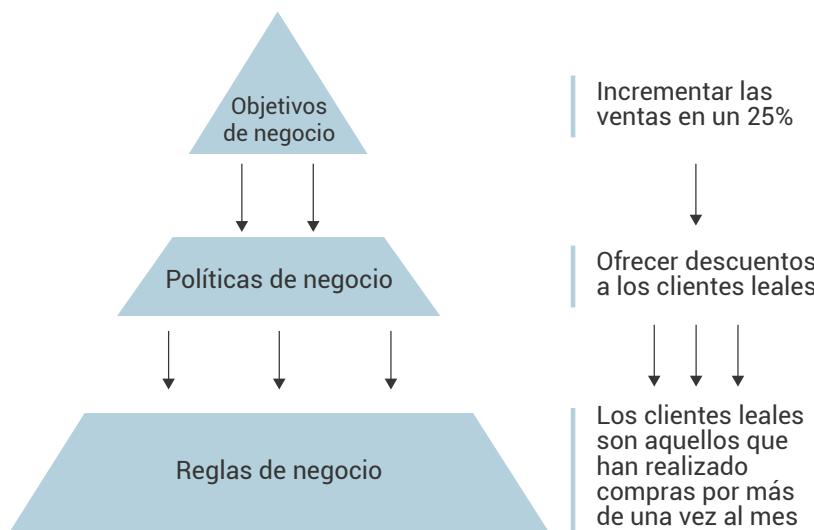
Para desarrollar éste enfoque puede utilizar diferentes estrategias que recomiendan especialmente los enfoques ágiles. Una de estas estrategias que permite definir de forma jerárquica y debidamente organizado el backlog del producto es la descomposición funcional. Ya que a partir de las características se identifican las épicas, historias de usuario y tareas. En el [anexo 5](#) se indica como utilizar ésta técnica.

4.7. Reglas de negocio

Desde el punto de vista organizacional, las reglas de negocio son importantes para el funcionamiento eficiente de una organización. Las organizaciones que carecen de reglas definidas y estándares de procedimiento con frecuencia descubrirán un lugar de trabajo caótico con resultados desiguales, baja moral de los empleados y consumidores insatisfechos. Por lo tanto, las reglas de negocio son declaraciones que regulan la toma de decisiones para ayudar a una empresa a alcanzar sus objetivos.

Figura 26

Objetivos, políticas y reglas de negocio.



Nota. Adaptado de *The Software Requirements Memory Jogger* (p. 9), por Gottesdiener, E., 2005, Goal/Opc.

Tal como se observa en la figura 26, y desde el punto de vista organizacional, las reglas de negocio se derivan de las políticas y estas a su vez de los objetivos de negocio.

En este sentido, cada organización opera de acuerdo con un amplio conjunto de políticas, leyes y estándares de la industria. Industrias como la banca, la aviación y la fabricación de dispositivos médicos deben cumplir con varias regulaciones gubernamentales. Dichos principios de control se conocen colectivamente como reglas de negocio o lógica de negocio. Las reglas de negocio a menudo se aplican mediante la implementación manual de políticas y procedimientos. Sin embargo, en muchos casos, las aplicaciones de software también deben hacer cumplir estas reglas.

Las reglas de negocio son una excelente fuente de requisitos porque dictan las propiedades que el sistema debe poseer para cumplir con las reglas. Las reglas de negocio pueden ser el origen de varios tipos de requisitos. La tabla 9 ilustra y proporciona ejemplos de cómo las reglas de negocio influyen en varios tipos de requisitos.

Tabla 9

Influencia de las reglas de negocio en los requisitos

Tipo de requisito	Influencia de las reglas de negocio	Ejemplo
Requisitos de negocio	Las regulaciones gubernamentales pueden conducir a los objetivos de negocio necesarios para un proyecto.	El sistema de seguimiento de productos químicos debe permitir el cumplimiento de todas las regulaciones federales y estatales sobre el uso de químicos y la eliminación de residuos en un plazo de cinco meses.
Requisitos de usuario	Las políticas de privacidad determinan qué usuarios pueden y no pueden realizar ciertas tareas con el sistema.	Solo los gerentes de laboratorio están autorizados a generar informes de exposición química para cualquier persona que no sean ellos mismos.
Requisito funcional	La política de la compañía es que todos los proveedores deben estar registrados y aprobados antes de que se pague una factura.	Si se recibe una factura de un proveedor no registrado, el sistema de proveedores enviará por correo electrónico las versiones editables en PDF del proveedor del formulario de admisión del proveedor y del formulario W-9.

Tipo de requisito	Influencia de las reglas de negocio	Ejemplo
Atributos de calidad	Las regulaciones de agencias gubernamentales, como OSHA y EPA, pueden dictar requisitos de seguridad, los cuales deben ser implementados a través de la funcionalidad del sistema.	El sistema debe mantener registros de entrenamiento de seguridad, que debe verificar para asegurar que los usuarios estén debidamente capacitados antes de que puedan solicitar un producto químico peligroso.

Nota. Adaptado de *Cómo las reglas de negocio pueden influir en varios tipos de requisitos de software* (p. 168), por K. Wiegers y J. Beatty, 2013, Microsoft Press.

No todas las empresas tratan sus reglas de negocio como un activo empresarial valioso. Ciertos departamentos pueden documentar sus reglas, pero muchas empresas carecen de un esfuerzo unificado para documentar las reglas de negocio en un repositorio común accesible para la organización de TI. Si las reglas de negocio no se documentan y administran adecuadamente, solo existirá en la mente de determinadas personas. Un BA necesita saber a quién llamar para conocer las reglas que afectan al proyecto. Las personas pueden tener interpretaciones conflictivas de las reglas, lo que puede llevar a que diferentes aplicaciones de software apliquen la misma regla de negocio de manera inconsistente o la pasen por alto por completo. Tener un repositorio de reglas de negocio facilita que todos los proyectos que se ven afectados por ciertas reglas aprendan sobre ellas y se implementen de manera consistente.

El [Business Rules Group](#) proporciona definiciones para las reglas de negocio desde la perspectiva tanto del negocio como de los sistemas de información:

- Desde la perspectiva de negocio: “Una regla de negocio es una guía de que existe una obligación con respecto a la conducta, acción, práctica o procedimiento dentro de una actividad o esfera en particular”.
- Desde la perspectiva del sistema de información: “Una regla de negocio es una declaración que define o restringe algún aspecto del negocio. Su objetivo es reafirmar la estructura empresarial o controlar o influir en el comportamiento de la empresa”.

Para efectos de la especificación de requisitos, simplemente identifique y documente las reglas que pertenecen al sistema y vincúlelas a los requisitos específicos que las implementará. Se han propuesto numerosos

esquemas de clasificación para organizar las reglas de negocio (Morgan, 2002), pero la taxonomía que se presenta en la figura 27, con cinco tipos de reglas funcionan en la mayoría de las situaciones. Una sexta categoría son términos, palabras definidas, frases y abreviaturas que son importantes para el negocio.

Figura 27
Taxonomía para reglas de negocio



Nota. Adaptado de *Una taxonomía de reglas de negocio simple* (p. 169), por K. Wieggers y J. Beatty, 2013, Microsoft Press.

Registrar las reglas de negocio de manera coherente es más importante que tener discusiones acaloradas sobre cómo clasificar cada una de ellas con precisión. Sin embargo, una taxonomía es útil para identificar reglas de negocios en las que quizás no hubiera pensado de otra manera. Clasificar las reglas también le da una idea de cómo podría aplicarlas en una aplicación de software. Por ejemplo, las restricciones a menudo conducen a la funcionalidad del sistema que hace cumplir las restricciones, y las acciones habilitadoras conducen a la funcionalidad para hacer que algo suceda bajo ciertas condiciones (Wieggers & Beatty, 2013). Veamos algunos ejemplos de estos cinco tipos de reglas de negocio en la siguiente infografía.

Definición y ejemplos de reglas de negocio

Las reglas de negocio se pueden representar usando herramientas tales como árboles de decisión y tablas de decisión (particularmente cuando está involucrada una lógica compleja) y matrices. Una plantilla para documentar las reglas de negocio deberá tener lo siguiente:

- Un identificador.
- Una definición de la regla.
- Pertener a una categoría (Hecho, restricción, acción facilitadora, inferencia o cálculo).
- Pertener a un grupo.
- Identificar si es estática o dinámica.
- Fecha en que será efectiva (si es el caso), Fuente (de ser necesario).
- En qué caso de uso se utiliza (en el momento apropiado).

Tabla 10

Ejemplo de reglas de negocio

ID	Definición de la regla	Categoría	Grupo	Estática o dinámica
HEC-001	Cada contenedor de productos químicos tiene un código de barras de identificador único.	Hecho	Política corporativa	Estática
HEC-002	Cada elemento de línea en un pedido representa una combinación específica de químicos, grado, tamaño del envase, y el número de contenedores.	Hecho	Política corporativa	Estática
REC-012	Todas las aplicaciones de software deben cumplir con las regulaciones gubernamentales para el uso por personas con discapacidad visual	Restricción	Política corporativa	Dinámica

Nota. Sucunuta, M., 2023

En la tabla 10 se indica la forma en que de una manera sencilla se pueden documentar las reglas de negocio. Cada regla de negocio considera un identificador único que le permite vincular a los requisitos. Por ejemplo, la plantilla para casos de uso contiene un campo para reglas de negocio que influyen en el caso de uso. En lugar de incluir la definición de regla en la descripción del caso de uso, simplemente, ingrese los identificadores de las reglas. De esta manera, no tiene que preocuparse de que la especificación del caso de uso se vuelva obsoleta si la regla se actualiza.

¿Qué le pareció la temática abordada?, interesante verdad, ahora le invito a que realices las siguientes actividades de aprendizaje.



Actividades de aprendizaje recomendadas

Estimado estudiante, finalizada la presente unidad, le recomiendo realizar las siguientes actividades con el objeto de profundizar los conocimientos.

1. Acceda al laboratorio virtual de la UTPL y verifique la disponibilidad de la herramienta Enterprise Architect.
2. Desarrolle el diagrama de casos de uso, del caso de estudio que se indica en el [anexo 1 caso de estudio](#). (registro de encomienda).
3. Desarrolle la descripción de cada caso de uso del diagrama desarrollado previa solicitud del tutor (revise en la plataforma de estudio).
4. Genere el reporte de la descripción de los casos de uso.
5. Socialice en la plataforma sobre los aprendizajes del uso de la herramienta.
6. Finalmente, realice la autoevaluación 4.



Autoevaluación 4

1. El análisis de requisitos se representa en los modelos de:
 - a. Requisitos.
 - b. Negocio.
 - c. Diseño.
2. Los modelos de análisis de requerimientos se representan mediante:
 - a. Mapas conceptuales.
 - b. Casos de uso de negocio.
 - c. Diagramas y/o texto estructurado.
3. Cuando se desea especificar sobre la información que entra y sale de un sistema, se recomienda construir un:
 - a. Mapa de diálogo.
 - b. Diagrama de contexto.
 - c. Tabla evento-respuesta.
4. El modelo que permite conocer quienes interactúan directamente con el sistema, es:
 - a. Tabla de actores.
 - b. Glosario.
 - c. Casos de uso.
5. Cuando se requiere modelar el negocio, es necesario:
 - a. Establecer políticas de negocio.
 - b. Combinar entre mapa de relaciones y/o mapa de procesos.
 - c. Priorizar requerimientos.
6. Cuando se desea adicionar detalle a los requerimientos de usuario, es conveniente desarrollar:
 - a. Casos de uso.
 - b. Diagrama de contexto.
 - c. Políticas de negocio.

7. Un mapa de procesos permite mostrar:
 - a. El tipo de información y productos que se intercambian entre clientes externos.
 - b. Una secuencia de pasos, entradas y salidas necesarias para manejar un proceso de negocio.
 - c. El sistema en su entorno.
8. Son reglas que sirven para limitar las acciones que el sistema o los usuarios deben realizar. Se les conoce como:
 - a. Restricciones.
 - b. Hechos.
 - c. Inferencias.
9. Son afirmaciones verdaderas acerca del negocio, describen asociaciones o relaciones entre los términos del negocio. A estas reglas se les conoce como:
 - a. Restricciones.
 - b. Hechos.
 - c. Inferencias.
10. Cuando existe un caso de uso padre del cual uno o más casos de uso hijos heredan sus características y especializan cierto comportamiento, se da una:
 - a. Relación de inclusión.
 - b. Relación de extensión.
 - c. Relación de generalización.

[Ir al solucionario](#)

- Resultado de aprendizaje 4**
- Documenta los diferentes requisitos mediante el uso de estándares de documentación.

El resultado de aprendizaje está orientado a que en calidad de analista de negocio elija y parametrice con base en el tipo de proyecto, la mejor estrategia para especificar requisitos y los documente con criterios de calidad. Para ellos se proponen alternativa con base en estándares y modelos, pero es importante lo que proponen las herramientas de gestión. Se utilizará Enterprise Architect, como herramienta de modelado y gestión de requisitos, lo que le permitirá ampliar sus expectativas en cuanto a documentación.

Contenidos, recursos y actividades de aprendizaje



Semana 12

Unidad 5. Documentando los requisitos

5.1. ¿Por qué documentar?

El resultado del desarrollo de requisitos es un acuerdo documentado entre los interesados sobre el producto a ser construido. Como se vio en los apartados anteriores, el documento de visión y alcance contiene los requisitos del negocio, y los requisitos de usuario pueden ser capturados en forma de casos de uso o historias de usuarios. Los requisitos funcionales y no funcionales del producto a menudo se documentan en la especificación de requisitos de software o ERS (SRS, por sus siglas en inglés), documento que se entrega a aquellos que deben diseñar, construir y verificar la solución. Los requisitos se registran de forma organizada para que los interesados clave del proyecto puedan ayudar a revisar y asegurar que están de acuerdo.

A pesar de esto, no todo el mundo está de acuerdo en que vale la pena dedicarle tiempo a documentar requisitos. A pesar de que en ciertos proyectos altamente volátiles en los que no se está seguro de con qué solución va a terminar, tratar de mantenerse al día con los cambios en

los detalles de los requisitos añade poco valor. Sin embargo, el costo de registrar el conocimiento es pequeño comparado con el costo de adquirir ese conocimiento o regenerarlo en algún momento en el futuro. Los actos de especificación y modelización ayudan a los participantes del proyecto a reflexionar y precisamente exponer cosas importantes que en una discusión verbal puede resultar ambigua. Si está 100 % seguro de que ningún interesado nunca necesitará una información específica más allá de la duración de sus propios recuerdos a corto plazo, entonces no necesita guardarlo. De lo contrario, guárdelo en algún tipo de medio.

Considere también que en la especificación nunca obtendrá requisitos perfectamente definidos. Recuerde que está escribiendo requisitos para ciertas audiencias. La cantidad de detalles, el tipo de información que usted proporciona y la forma en que lo organiza deben ser todos destinados a satisfacer las necesidades de sus audiencias. Los analistas, naturalmente, escriben los requisitos desde su propio punto de vista, pero realmente se deben escribir lo más significativo para aquellos que tienen que entender los requisitos y hacer el trabajo basado en ellos. Por eso es importante que los representantes de esas audiencias revisen los requisitos para asegurarse de que se satisface sus necesidades.

No espere la mejor documentación de requisitos para reemplazar las discusiones a lo largo del proyecto. Mantenga abiertas las líneas de comunicación entre el BA, el equipo de desarrollo, los representantes de los clientes y otros interesados para que puedan abordar rápidamente los innumerables problemas que se presentarán.

Puede representar los requisitos de software de varias maneras, incluyendo:

- Lenguaje natural, bien estructurado y cuidadosamente escrito.
- Modelos visuales que ilustran procesos de transformación, estados del sistema y cambios entre ellos, relaciones de datos, flujos lógicos, y similares.
- Especificaciones formales que definen requisitos mediante el uso de lenguajes de especificación matemáticamente precisos.

Las especificaciones formales proporcionan mayor rigor y precisión, pero pocos desarrolladores de software y aún menos los clientes están expuestos a ellos. La mayoría de los proyectos no exigen este nivel de

formalidad, pero seguramente que los diseñadores de sistemas de alto riesgo como los sistemas de control de centrales nucleares utilizan métodos de selección formales. El lenguaje natural estructurado, seguido de modelos visuales y otras técnicas de representación, sigue siendo la forma más práctica para la mayoría de los proyectos de software para documentar sus necesidades.

- Los clientes, el departamento de *marketing* y el personal de ventas necesitan saber qué producto pueden esperar para desarrollar sus actividades.
- Los administradores del proyecto basan sus estimaciones de calendario, esfuerzo y recursos en los requisitos.
- Los equipos de desarrollo de *software* necesitan saber qué construir.
- Los probadores lo utilizan para desarrollar pruebas basadas en requisitos, planes de prueba y procedimientos de prueba.
- El personal de mantenimiento y el personal de soporte lo usan para entender lo que cada parte del producto se supone debe hacer.
- Escritores de documentación, manuales de usuario y pantallas de ayuda se basan en el ERS y el diseño de la interfaz de usuario.
- El personal de capacitación utiliza el ERS y la documentación del usuario para desarrollar materiales educativos.
- El personal jurídico asegura que los requisitos cumplen con las leyes y regulaciones.
- Los subcontratistas basan su trabajo en los requisitos especificados y pueden ser considerados legalmente.



No tiene que escribir el ERS para todo el producto antes de comenzar el desarrollo, pero debe capturar los requisitos para cada incremento antes de crear ese incremento. El desarrollo incremental es apropiado cuando se desea obtener rápidamente alguna funcionalidad de manos de los usuarios. La retroalimentación de usar los primeros incrementos dará forma al resto del proyecto. Sin embargo, cada proyecto debe basar un acuerdo para cada

conjunto de requisitos antes de que el equipo los implemente. La línea base es el proceso de transición de un ERS en desarrollo en uno que ha sido revisado y aprobado. Trabajar con un conjunto acordado de requisitos minimiza la falta de comunicación y la repetición innecesaria.

Es importante organizar y escribir el ERS para que los diversos actores puedan entenderlo. Tenga en cuenta las siguientes sugerencias de legibilidad:

- Utilice una plantilla apropiada para organizar toda la información.
- Etiquete y diseñe secciones, subsecciones y requisitos individuales de forma consistente.
- Utilice el énfasis visual (negrita, subrayado, cursiva, color y fuentes) de manera consistente y juiciosa. Recuerde que el resultado del color puede no ser visible para las personas con ceguera de color o cuando se imprime en escala de grises.
- Crear una tabla de contenido para ayudar a los lectores a encontrar la información que necesitan.
- Numere todas las figuras y tablas, póngales subtítulos, y refiérase a ellos por número.
- Si está almacenando los requisitos en un documento, defina la facilidad de referencia cruzada del procesador de textos en lugar de las páginas codificadas o números de sección para hacer referencia a otras ubicaciones dentro de un documento.
- Si está utilizando documentos, defina hipervínculos para que el lector salte a las secciones relacionadas en el SRS o en otros archivos.
- Si está almacenando los requisitos en una herramienta, utilice vínculos para que el lector navegue en la información relacionada.
- Incluir representaciones visuales de la información cuando sea posible para facilitar la comprensión.
- Recurrir a un editor especializado para asegurarse de que el documento es coherente y utiliza un vocabulario y plan coherente.

5.2. Documento de Especificación de Requisitos de Software

El equipo de desarrollo debe adoptar una estrategia que le permita documentar los requisitos de forma bien organizada, de entre las opciones está la de utilizar una plantilla. Existen varias plantillas ERS que se puede utilizar para documentar, basta con buscar en la web y tenemos formatos muy parecidos. La figura 28 ilustra una plantilla ERS que funciona bien para muchos tipos de proyectos (Wiegers & Beatty, 2013).

Figura 28

Plantilla para especificar requisitos de software

- | |
|--|
| <ul style="list-style-type: none">1. Introducción<ul style="list-style-type: none">1.1. Propósito1.2. Convenciones del documento1.3. Alcance del proyecto1.4. Referencias2. Descripción general<ul style="list-style-type: none">2.1. Perspectiva del producto2.2. Clases de usuario y características2.3. Ambiente de operación2.4. Restricciones de diseño e implementación2.5. Suposiciones y dependencias3. Características del sistema<ul style="list-style-type: none">3.1. Característica del sistema X<ul style="list-style-type: none">3.1.1. Descripción3.1.2. Requisito funcional3.2.<ul style="list-style-type: none">3.2.1.3.2.2.4. Requisitos de datos<ul style="list-style-type: none">4.1. Modelo lógico de datos4.2. Diccionario de datos4.3. Reportes4.4. Adquisición de datos, integridad, retención y disposición5. Requisitos de interfaz externa<ul style="list-style-type: none">5.1. Interfaz de usuario5.2. Interfaz de software5.3. Interfaz de hardware5.4. Interfaz de comunicación6. Atributos de calidad<ul style="list-style-type: none">6.1. Usabilidad6.2. Rendimiento6.3. Seguridad6.4. Disponibilidad6.5. Robustez6.6. Mantenibilidad6.7. Otros7. Internacionalización y ubicación de requisitos8. Otros requisitos |
|--|

Apéndice A: Glosario
Apéndice B: Modelos de análisis

Nota. Adaptado de *Software Requirements* (p. 191), por Wiegers, K. y Beatty, J., 2013. Microsoft Press.

En el [anexo 4 Descripción de la plantilla de ERS](#), se describe cada una de las partes que se indican en la plantilla de la figura 28. El desarrollo de la documentación requiere de elementos clave que son necesario analizar, debido a que son parte del documento ERS. Por ese motivo es necesario analizar al detalle, temas como :

- Etiquetado de requisitos.
- Características de la declaración de requisitos.
- Lineamientos para escribir requisitos.



Semana 13

5.3. Etiquetado de requisitos

Los requisitos requieren de un identificador único. Esto permitirá hacer referencia a requisitos específicos cuando exista una solicitud de cambio, un historial de modificaciones, una referencia cruzada o una matriz de trazabilidad de requisitos. También permite reutilizar los requisitos en múltiples proyectos. Los requisitos identificados de forma única facilitan la colaboración entre los miembros del equipo cuando discuten los requisitos, como en una reunión de revisión por pares. A continuación, se describen varios métodos de etiquetado de requisitos.

5.3.1. Número secuencial

Es el enfoque más simple, a cada requisito se lo identifica con un número de secuencia único, por ejemplo UC-1, UC-2, UC-3, ... Las herramientas de gestión de requisitos asignan tal identificador cuando se añade un nuevo requisito. Un número no se reutiliza si se elimina un requisito. Este enfoque de numeración simple no proporciona ningún agrupamiento lógico o jerárquico de los requisitos relacionados, el número no implica ningún tipo de ordenación, y las etiquetas no dan ninguna pista en cuanto a lo que cada requisito se trata. Facilita la retención de un identificador único si mueve los requisitos en un documento.

5.3.2. Numeración jerárquica

Lo más común, si los requisitos funcionales aparecen en la sección 3.2 de su ERS, todos ellos tendrán etiquetas que comienzan con 3.2. Más dígitos

que indican un requisito más detallado, de menor nivel, por lo que usted sabe que 3.2.4.3 es un requisito hijo de 3.2.4. Este método es simple, compacto y familiar.

La numeración jerárquica plantea algunos problemas. Las etiquetas pueden crecer hasta muchos dígitos, incluso en un ERS de tamaño mediano. Las etiquetas numéricas no le dicen nada sobre la intención de un requisito. Si está utilizando un procesador de textos, normalmente este esquema no genera etiquetas persistentes.

Si inserta un nuevo requisito, se incrementarán los números de los siguientes requisitos en esa sección. Elimine o mueva un requisito y los números que le siguen en esa sección se reducirán. Estos cambios interrumpen cualquier referencia a esos requisitos en otras partes del sistema. Por ejemplo:

Figura 29

Ejemplo de numeración jerárquica para los requisitos

-
- 1. Gestión de clientes desde el sitio web
 - 1.1 Registrar datos personales
 - 1.2 Administrar las credenciales que se asignarán a los clientes. Se asignará un usuario y clave y deberá ajustarse a un proceso de validación.
 - 1.2.1 Si el cliente olvidó sus credenciales, el sistema le permitirá generar nuevas credenciales.
 - 1.2.2 El cliente podrá cambiar su clave, para ello deberá ingresar al portal web y luego realizar el cambio.
-

Nota. Sucunuta, M.. 2023

5.3.3. Etiqueta textual jerárquica

Este etiquetado se sugiere para etiquetar requisitos individuales. Tenga en cuenta este requisito: "El sistema pedirá al usuario que confirme cualquier solicitud para imprimir más de 10 copias". Este requisito podría ser etiquetado como Print.ConfirmCopies. Esto indica que forma parte de la

función de impresión y se refiere al número de copias que se van a imprimir. Las etiquetas textuales jerárquicas están estructuradas, son significativas y no se ven afectadas, añadiendo, eliminando o moviendo otros requisitos. Este método también es adecuado para etiquetar reglas de negocio si las mantiene manualmente, en lugar de hacerlo en un repositorio o herramienta de reglas de negocio.

El uso de etiquetas de texto jerárquico ayuda a resolver el problema de las relaciones padre-hijo entre los requisitos. Si el padre está escrito como un requisito funcional, la relación entre los hijos y el padre puede ser confusa. Una buena convención es escribir el requerimiento de los padres para que parezca un título, un encabezado o un nombre de entidad, en lugar de parecer un requisito funcional en sí mismo. Los requisitos de los hijos de ese padre, en conjunto, entregan la capacidad descrita en el padre. Por ejemplo:

Figura 30
Uso de etiquetas de texto jerárquico

Producto: Pedidos de productos desde el sitio web.

.Carro: El sitio web utilizará un carrito de compras que contiene los productos que el cliente selecciona para comprar.

.Descuento: El carro de comprar proporcionará un campo de código de descuento. Cada código de descuento proporciona un porcentaje de descuento específico o un monto de descuento fijo en dólares en artículos específicos del carro.

.Error: Si el cliente ingresa un código de descuento incorrecto, el sitio web presentará un mensaje de error.

Nota. Sucunuta, M.. 2023

Se aprecia el primer requisito funcional, se etiqueta **Producto. Carro**. El ID completo para el tercer requisito es **Producto. Descuento**. Este esquema jerárquico evita los problemas de mantenimiento con la numeración jerárquica, pero las etiquetas son más largas y hay que pensar en nombres

significativos para ellos, tal vez construyendo a partir del nombre entidades relevantes. Puede ser difícil mantener la singularidad, especialmente si tiene varias personas trabajando en el conjunto de requisitos. Puede simplificar el esquema combinando la técnica de nomenclatura jerárquica con un sufijo de número de secuencia para pequeños conjuntos de requisitos: **Producto**. **Carro.01**, **Producto**. **Carro.02** y así sucesivamente.

5.4. Características de la declaración de requisitos

La mejor manera de saber si sus requisitos poseen los atributos deseados es que varios interesados los revisen. Diferentes actores identificarán diferentes tipos de problemas.

5.4.1. Características de las declaraciones de requisitos

En un mundo ideal, cada requisito de negocio, usuario, funcional y no funcional debe contar con las cualidades descritas en las siguientes secciones.

- **Completo**

Cada requisito debe contener toda la información necesaria para que el lector la entienda. En el caso de los requisitos funcionales, significa proporcionar la información que el desarrollador necesita para poder implementarla correctamente. Si sabe que le falta cierta información, use TBD (a determinar) como un indicador estándar para resaltar estas lagunas, o regístrelas en un sistema de seguimiento de problemas para hacer un seguimiento.

- **Correcto**

Cada requisito debe describir con precisión una capacidad que satisfaga la necesidad de algún interesado y debe describir claramente la funcionalidad que se construirá. Usted tendrá que ir a la fuente del requisito para comprobar su corrección. Este podría ser un usuario que proporcionó el requisito inicial, un requisito de sistema de nivel superior, un caso de uso, una regla de negocio u otro documento. Un requisito de bajo nivel que entra en conflicto con su matriz no es correcto. Para evaluar la exactitud de los requisitos de los usuarios, los representantes de los usuarios o sus sustitutos deben revisarlos.

- **Factible**

Debe ser posible implementar cada requisito dentro de las capacidades y limitaciones conocidas del sistema y su entorno operativo, así como dentro de las limitaciones de tiempo, presupuesto y personal del proyecto. Un desarrollador que participa durante la obtención puede proporcionar una verificación de la realidad de lo que se puede y no se puede hacer técnicamente y lo que se puede hacer solo a un costo excesivo o esfuerzo. Los enfoques de desarrollo incremental y los prototipos de prueba de concepto son dos maneras de evaluar la factibilidad de los requisitos. Si un requerimiento necesita ser cortado porque no es factible, entienda el impacto en la visión y alcance del proyecto.
- **Necesario**

Cada requisito debe describir una capacidad que proporcione al interesado el valor comercial por anticipado, que diferencie el producto en el mercado o que se requiera para la conformidad con una norma, política o norma externa. Cada requisito debe provenir de una fuente que tiene la autoridad para proporcionar los requisitos. Trace los requisitos funcionales y no funcionales de nuevo a la entrada específica de voz del usuario, como un caso de uso o una historia de usuario. Usted debe ser capaz de relacionar cada requisito con un objetivo de negocio que indica claramente por qué es necesario. Si alguien pregunta por qué se incluye un requisito particular, debe haber una buena respuesta.
- **Priorizado**

Priorizar los requisitos de negocio según cuáles son los más importantes para alcanzar el valor deseado. Asigne una prioridad de implementación a cada requisito funcional, requerimiento del usuario, caso de uso o función para indicar cuán esencial es para un lanzamiento de producto en particular. Si todos los requisitos son igualmente importantes, el director del proyecto no sabe cómo responder mejor a los rebasamientos de planificaciones, las pérdidas de personal o los nuevos requisitos que se presentan. La priorización de los requisitos debe ser una actividad de colaboración que abarque múltiples perspectivas de los interesados.

- **No ambiguo**

El lenguaje natural es propenso a dos tipos de ambigüedad. Un tipo que puedo verme, cuando puedo pensar en más de una manera de interpretar un requisito dado. El otro tipo de ambigüedad es más difícil de detectar. Eso es cuando diferentes personas leen el requisito y llegar a diferentes interpretaciones de esta. El requisito tiene sentido para cada uno de ellos, pero significa algo diferente a cada uno de ellos. Las inspecciones son una buena manera de detectar las ambigüedades (Wiegert 2002). Una revisión formal por pares, como una inspección (en lugar de simplemente distribuir los requisitos a las personas a examinar por su cuenta) ofrece una oportunidad para cada participante para comparar su comprensión de cada requisito a alguien más. “Comprendible” se relaciona con “inequívoco”: los lectores deben entender lo que cada requisito está diciendo.

Nunca eliminarás toda la ambigüedad de los requisitos: esa es la naturaleza del lenguaje humano. La mayoría de las veces, personas razonables pueden sacar las conclusiones correctas de incluso un requisito ligeramente difuso. Obtener un poco de ayuda de sus colegas a través de revisiones se limpiará un montón de los peores problemas, sin embargo.

- **Verificable**

¿Puede un probador diseñar pruebas u otros métodos de verificación para determinar si cada requisito se implementa correctamente? Si un requisito no es verificable, decidir si se implementó correctamente se convierte en una cuestión de opinión, no en un análisis objetivo. Los requisitos que son incompletos, inconsistentes, infalsificables o ambiguos tampoco son verificables. Los probadores son buenos para examinar los requisitos de variabilidad. Incluya en sus requisitos revisiones de pares para detectar problemas temprano.

5.4.2. Características de las colecciones de requisitos

No es suficiente tener excelentes declaraciones de requisitos individuales. Los conjuntos de requisitos que se agrupan en una línea de base para una liberación o iteración específica deben presentar las características descritas en las siguientes secciones, ya sea que estén registradas en un

documento ERS, una herramienta de gestión de requisitos, un conjunto de historias de usuario y pruebas de aceptación o cualquier otra forma.

- **Completo**

Ningún requisito o información necesaria debe estar ausente. En la práctica, nunca documentará todos los requisitos de un sistema. Siempre hay algunos requisitos implícitos o asumidos, aunque tienen más riesgo que los requisitos explícitamente establecidos. Los requisitos que faltan son difíciles de detectar porque no están allí. La sección “Evitar la incompletitud” más adelante en este capítulo sugiere algunas maneras de identificar los requisitos que faltan. Cualquier especificación que contenga TBDs es incompleta.

- **Consistente**

Los requisitos consistentes no entran en conflicto con otros requisitos del mismo tipo o con requisitos de negocio, de usuario o de sistema de nivel superior. Si no resuelven las contradicciones entre los requisitos antes de bucear en la construcción, los desarrolladores tendrán que lidiar con ellos. Grabar el originador de cada requisito le permite saber con quién hablar si descubre conflictos. Puede ser difícil detectar inconsistencias cuando la información relacionada se almacena en diferentes ubicaciones, como en un documento de visión y alcance y en una herramienta de administración de requisitos.

- **Modificable**

Siempre puede reescribir un requisito, pero debe mantener un historial de cambios realizados en cada requisito, especialmente después de que se basan. También necesita saber acerca de las conexiones y dependencias entre los requisitos para que pueda encontrar todos los que deben cambiarse juntos. La habilidad Modi dicta que cada requisito debe ser etiquetado de manera única y expresado por separado de otros, de modo que pueda referirse a él sin ambigüedad.

- **Trazable**

Un requisito rastreable puede vincularse tanto a su origen como hacia delante a requisitos derivados, elementos de diseño, código que lo implementa y pruebas que verifican su implementación. Tenga en cuenta que en realidad no tiene que definir todos estos vínculos

de rastreo para un requisito de tener las propiedades que lo hacen rastreable. Los requisitos rastreables están etiquetados únicamente con identificadores persistentes. Se escriben de una manera estructurada, no en párrafos largos de la narrativa. Evite combinar varios requisitos juntos en una sola sentencia, ya que los diferentes requerimientos podrían rastrearse a diferentes componentes de desarrollo.

5.5. Lineamientos para escribir requisitos

No existe una fórmula o estrategia para escribir excelentes requisitos. Los mejores profesores son la experiencia y la retroalimentación de los destinatarios de los requisitos. Recibir retroalimentación de colegas es una gran ayuda porque puede aprender de lo que hizo en la escritura. Esta es la razón por la cual los exámenes por pares de los documentos de requisitos son tan críticos. Para comenzar con las revisiones, compórtese con un analista de negocios y empiece a intercambiar los requisitos para su revisión.

Cuando decimos “escritura de requisitos”, la gente inmediatamente piensa en escribir requisitos textuales en lenguaje natural. Es mejor traducir mentalmente la frase “requisitos de escritura” a “representar el conocimiento de requisitos”. En muchos casos, las técnicas de representación alternativas pueden presentar información de manera más efectiva que el texto en línea recta. (Wigers, 2006). El BA debe elegir una combinación adecuada de métodos de comunicación que asegure un entendimiento claro y compartido de las necesidades de los interesados y la solución a construir.

Los requisitos se pueden mejorar, y siempre hay formas equivalentes de expresarlos. Los dos objetivos importantes en la redacción de los requisitos son los siguientes:

- Cualquiera que lea el requisito llega a la misma interpretación que cualquier otro lector.
- La interpretación de cada lector coincide con lo que el autor pretendía comunicar. Estos resultados son más importantes que la pureza de estilo o se adaptan dogmáticamente a alguna regla o convención arbitraria.

5.5.1. Perspectiva del sistema o del usuario

Puede escribir requisitos funcionales desde la perspectiva de lo que hace el sistema o lo que el usuario puede hacer. Debido a que la comunicación efectiva es la meta general, es necesario mezclar estos estilos, expresando cada requisito en cualquier estilo que sea más claro. El estado de los requisitos de manera consistente, como “El sistema deberá” o “El usuario deberá”, seguido de un verbo de acción, seguido por el resultado observable. Especifique la acción o condición de activación que hace que el sistema realice el comportamiento especificado. Una plantilla genérica para un requisito escrito desde la perspectiva del sistema es:

[Precondición opcional] [evento de disparo opcional] el sistema [deberá responder al sistema].

Ejemplo:

Si el producto químico solicitado se encuentra en el almacén de productos químicos, el sistema deberá mostrar una lista de todos los contenedores del producto químico que se encuentran actualmente en el almacén.

Al escribir requisitos funcionales desde la perspectiva del usuario, la siguiente estructura general funciona bien:

La [clase de usuario o nombre del actor] será capaz de [hacer algo] [a algún objeto] [condiciones de calificación, tiempo de respuesta o declaración de calidad].

Fraseos alternativos son:

“El sistema dejará (o permitirá, permitirá o habilitará) [un nombre de clase de usuario particular] a [hacer algo]”.

A continuación, se muestra un ejemplo de requisito funcional escrito desde la perspectiva del usuario:

El Químico podrá reorganizar cualquier producto químico que haya ordenado en el pasado, recuperando y editando los detalles del pedido.

Observe cómo este requisito utiliza el nombre de la clase de usuario -Químico- en lugar del genérico “usuario”. Hacer el requisito tan explícito como sea posible reduce la posibilidad de una interpretación errónea.

5.5.2. Estilo de escritura

Ajuste su estilo de escritura para poner primero la línea de fuerza “la declaración de la necesidad o funcionalidad” seguida de detalles de apoyo (justificación, origen, prioridad y otros atributos de requisito). Esta estructura ayuda a los lectores que solo están examinando un documento, mientras que para aquellos lectores que necesitan todos los detalles, siga siendo útil. Incluir tablas, listas estructuradas, diagramas y otros elementos visuales ayuda a romper una letanía monótona de requisitos funcionales y proporciona una comunicación más rica a aquellos que aprenden mejor de diferentes maneras.

Considere las siguientes sugerencias al realizar la declaración de requisitos para obtener la máxima eficacia de la comunicación.

5.5.2.1. Claridad y conciso:

Escribir los requisitos en frases completas usando gramática, ortografía y puntuación apropiadas. Mantenga frases y párrafos cortos y directos. Escribir los requisitos en un lenguaje sencillo y directo adecuado al dominio del usuario, evitando la jerga. Definir los términos especializados en un glosario.

Otra buena pauta es escribir de forma concisa. Frases como “necesita proporcionar al usuario la capacidad de” se puede condensar en “debe”. Para cada información de los requisitos, pregúntese: “¿Qué haría el lector con esta información?” Si no está seguro de que algunos interesados encontrarían esa información valiosa, tal vez no la necesite. Sin embargo, la claridad es más importante que la concisión.

Los requisitos precisos aumentan la probabilidad de que las personas reciban lo que esperan; los requisitos menos específicos ofrecen al desarrollador más argumentos para la interpretación. A veces esa falta de especificidad está bien, pero en otros casos puede conducir a demasiada variabilidad en el resultado. Si un desarrollador que revisa el ERS no está claro sobre la intención del cliente, considere incluir información adicional para reducir el riesgo de problemas más adelante.

5.5.2.2. La palabra clave “debe”

Una convención tradicional es usar la palabra clave para describir alguna capacidad del sistema. La gente a veces se opone a la palabra “debe”. “No es así como la gente habla”, protestan. ¿Y qué? Las declaraciones “Deberán” indicar claramente la funcionalidad deseada, de acuerdo con su objetivo general de comunicación clara y efectiva. Es posible que prefiera decir “debe”, “necesita” o algo similar, pero sea coherente. A veces leo especificaciones que contienen una mezcla aleatoria y confusa de verbos de requisitos: puede, debe, puede, puede, va a, debería, podría, necesita, tiene que, debería proporcionar, y otros. Nunca sé si hay diferencias entre los significados de estos o no. Los matices entre los diferentes verbos también hacen que el documento sea mucho más difícil para los equipos interculturales de interpretar de manera consistente. Es mejor que se pegue con una palabra clave como “debe”.

Algunos autores de requisitos deliberadamente utilizan diferentes verbos para implicar distinciones sutiles. Ellos usan ciertas palabras clave para significar prioridad: “debe” significa requerido, “debería” significa deseado, y “puede” significa opcional. Consideramos que tales convenciones son peligrosas. Es más claro siempre decir “debe” y asignar explícitamente prioridad alta, media o baja a cada requisito. Además, las prioridades cambiarán a medida que progresen las iteraciones, por lo que no las vinculan al enunciado de los requisitos. El “deber” de hoy podría convertirse en “debe” de mañana. Otros autores usan “debe” para indicar un requisito y “voluntad” para designar una expectativa de diseño. Tales convenciones corren el riesgo de que algunos lectores no entiendan las distinciones entre las palabras que las personas usan indistintamente en la conversación cotidiana; Es mejor evitarlos.

5.5.2.3. Voz activa

Escriba en la voz activa para dejar claro qué entidad está tomando la acción descrita. Muchas escrituras de negocios y científicas están en la voz pasiva, pero nunca es tan clara y directa como usar la voz activa. El siguiente requisito está escrito en voz pasiva:

Tras el envío de la actualización del producto, el número de serie se actualizará en la línea del contrato.

El fraseo “se actualizará” es indicativo de voz pasiva. Denota el destinatario de la acción (número de serie), pero no el ejecutante de la acción. Es decir, este fraseo no ofrece ninguna pista sobre quién o qué actualiza el número de serie. ¿El sistema lo hará automáticamente, o se espera que el usuario actualice el número de serie? Reescribir este requisito en voz activa hace que el actor sea explícito y también aclara el evento desencadenante:

Cuando Fulfillment confirme que envió una actualización del producto, el sistema actualizará el contrato del cliente con el nuevo número de serie del producto.

5.5.2.4. Requisitos individuales

Evite escribir párrafos narrativos largos que contengan múltiples requisitos. Los lectores no deben recoger los requisitos individuales incrustados en una masa de lenguaje descriptivo libre. Distinguir claramente los requisitos individuales de la información contextual o de fondo. Esa información es valiosa para los lectores, pero necesitan reconocer inequívocamente las declaraciones de requisitos reales. Una vez revisé una amplia especificación de requisitos escrita en forma de párrafos largos. Pude leer una página completa y entenderlo, pero tuve que trabajar duro para seleccionar los requisitos discretos. Otros lectores podrían llegar a conclusiones diferentes de exactamente qué requisitos estaban acechando en esa masa de texto.

Palabras como “y”, “o”, “adicionalmente” o “también” en un requisito sugieren que varios requerimientos podrían haber sido combinados. Esto no significa que usted no puede usar “y” en un requisito; solo asegúrese de que la conjunción está uniendo dos partes de un solo requisito en lugar de dos requisitos separados. Si usas diferentes pruebas para verificar las dos partes, divide la oración en requisitos separados.

Evite usar “y / o” en un requisito; Deja la interpretación al lector, como en este caso:

El sistema debe permitir la búsqueda por número de pedido, número de factura y / o número de orden de compra del cliente.

Este requisito permitiría al usuario ingresar uno, dos o tres números a la vez al realizar una sola búsqueda. Puede que no sea lo que se pretende.

Las palabras “a menos que”, “excepto” y “pero” también indican la presencia de requisitos múltiples:

La tarjeta de crédito del comprador se cobrará por el pago, a menos que la tarjeta de crédito haya expirado.

Si no se especifica qué sucede cuando la cláusula “a menos que” sea verdadera es una fuente común de requisitos que faltan. Divila esto en dos requisitos para abordar el comportamiento de las dos condiciones de la tarjeta de crédito que está activo y expiró:

Si la tarjeta de crédito del comprador en el archivo está activa, el sistema cargará el pago a esa tarjeta.

y

Si la tarjeta de crédito del comprador ha caducado, el sistema permitirá al comprador actualizar la información de la tarjeta de crédito actual o introducir una nueva tarjeta de crédito para el pago.

5.5.2.5. Niveles de detalle

Los requisitos deben especificarse con un nivel de precisión que proporcione a los desarrolladores y probadores la suficiente información para implementarlos correctamente.

- **Detalles apropiados**

Una parte importante del análisis de los requisitos es descomponer un requisito de alto nivel en suficiente detalle para aclararlo y darle cuerpo. No hay una sola respuesta correcta a la pregunta común: “¿Cuán detallados deben ser los requisitos?” Proporcione suficientes detalles para minimizar el riesgo de malentendidos, basándose en el conocimiento y la experiencia del equipo de desarrollo. Si un desarrollador puede pensar en varias formas posibles de satisfacer un requisito y todos son aceptables, la especificidad y los detalles son correctos. De acuerdo Wieggers (2006) debe incluir más detalles cuando:

- a. El trabajo se está haciendo para un cliente externo.
- b. El desarrollo o las pruebas serán externalizados.
- c. Los miembros del equipo del proyecto están geográficamente dispersos.
- d. Las pruebas del sistema se basarán en los requisitos.
- e. Se necesitan estimaciones exactas.

- f. La trazabilidad de los requisitos es necesaria.

Es seguro incluir menos detalles cuando:

- g. El trabajo se está haciendo internamente para su empresa.
- h. Los clientes están muy involucrados.
- i. Los desarrolladores tienen experiencia considerable en el dominio.
- j. Los precedentes están disponibles, como cuando se está reemplazando una aplicación anterior.
- k. Se utilizará una solución de paquete.

- **Granularidad consistente**

Los autores de requisitos a menudo se esfuerzan por encontrar el nivel adecuado de granularidad para escribir requisitos funcionales. No es necesario especificar todos sus requisitos al mismo nivel de detalle. Por ejemplo, puede profundizar en un área que presenta mayor riesgo que otros. Sin embargo, dentro de un conjunto de requisitos relacionados, es una buena idea tratar de escribir requisitos funcionales en un nivel consistente de granularidad.

Una guía útil es escribir requisitos individualmente verificables. Incluso se ha propuesto el recuento de requisitos de prueba como una métrica para el tamaño del producto de software. Si puede pensar en un pequeño número de casos de prueba relacionados para verificar que un requisito se implementó correctamente, es probable que en una granularidad adecuada. Si usted prevé numerosas y diversas pruebas, tal vez varios requisitos se combinan y deben ser separados.

He visto declaraciones de requisitos en el mismo ERS que varían ampliamente en su alcance. Por ejemplo, las dos funciones siguientes se dividieron como requisitos separados:

- El sistema interpretará la combinación de teclas Ctrl + S como Archivo Guardar.
- El sistema interpretará la combinación de teclas Ctrl + P como Archivo Imprimir.

Estos requisitos son muy finos. Necesitarán pocas pruebas para verificar el comportamiento correcto. Se puede imaginar una lista tediosamente larga

de requisitos similares, que sería mejor expresarse en forma de una tabla que enumera todos los atajos de teclas y cómo el sistema los interpreta.

Sin embargo, ese mismo ERS también contenía un requisito funcional que parecía bastante amplio en su alcance:

El producto responderá a las instrucciones de edición introducidas por voz.

Este requisito único-aparentemente no más grande o pequeño que todos los demás en el ERS-estipuló la inclusión de un complejo subsistema de reconocimiento de voz-prácticamente un producto completo. La verificación de este requisito en el sistema de trabajo podría requerir cientos de pruebas. El requisito tal como se indica aquí podría ser apropiado en el alto nivel de abstracción que se encuentra en un enunciado de visión o un documento de requisitos del mercado, pero el requisito de reconocimiento de voz claramente exige mucho más detalle de funcionalidad.

¿Qué le pareció la temática abordada? Interesante verdad, ahora le invito a que realice las siguientes actividades de aprendizaje.



Actividades de aprendizaje recomendadas

Estimado estudiante finalizada la presente unidad, le recomiendo realizar las siguientes actividades con el objeto de profundizar los conocimientos.

1. Acceda al laboratorio virtual de la UTPL y verifique la disponibilidad de la herramienta Enterprise Architect.
2. Desarrolle los modelos de requisitos que se indica en el [anexo 4 Descripción de la plantilla de ERS](#), luego genere la documentación respectiva.
3. Socialice con sus compañeros la experiencia del uso de la herramienta, mediante las actividades planificadas en plataforma educativa.
4. Finalmente, estimado estudiante le invito a realizar la autoevaluación 5.



Autoevaluación 5

1. El documento que sirve como base para el diseño e implementación del sistema se lo conoce como:
 - a. Carta de compromiso.
 - b. Visionamiento.
 - c. Especificación de Requerimientos de Software.
2. El responsable directo de la elaboración del documento "Especificación de Requerimientos de Software" es el:
 - a. Analista.
 - b. Cliente.
 - c. Usuario.
3. Al proceso de elaboración, refinamiento y organización de los requerimientos plasmados en un documento, se conoce como:
 - a. Especificar requisitos.
 - b. Validar requerimientos.
 - c. Verificar requerimientos.
4. Una de las fuentes para elaborar el documento de requisitos de usuario es:
 - a. Plantilla de requerimientos.
 - b. Base de datos.
 - c. Visión del producto.
5. Al documento que actúa entre puente entre la definición del negocio y los requerimientos de software, se le conoce como:
 - a. Plantilla de requerimientos.
 - b. Requerimientos de usuario.
 - c. Visión del producto.

6. El ERS con respecto a la validación y verificación se convierte en:
 - a. Línea base.
 - b. Plan de prueba.
 - c. Casos de prueba.
7. Para desarrollar el documento de requerimientos de usuario, inicialmente se recomienda:
 - a. Identificar las fuentes.
 - b. Entrevistar a los interesados.
 - c. Desarrollar la arquitectura de la aplicación.
8. Los requisitos no funcionales están directamente relacionados con:
 - a. Atributos de calidad.
 - b. Casos de uso.
 - c. Escenarios.
9. Las limitaciones de diseño e implementación en el ERS:
 - a. No se consideran.
 - b. Se consideran de forma general.
 - c. Son irrelevantes.
10. Cuando se puede determinar los componentes que dan origen al requerimiento, entonces este es:
 - a. Verificable.
 - b. Completo.
 - c. Trazable.

[Ir al solucionario](#)



Unidad 6. Validando los requisitos

6.1. Verificación y validación

La validación de requisitos es el cuarto componente del desarrollo de requisitos, junto con la obtención, análisis y especificación. Algunos autores usan el término “verificación” para este paso. Sin embargo, vamos a adoptar la terminología del Cuerpo de Conocimientos de Ingeniería de Software y referimos este aspecto del desarrollo de requisitos como “validación” (Abran et al., 2004).

La verificación de los requisitos para asegurar que tengan todas las propiedades deseadas de requisitos de alta calidad es también una actividad esencial. Precisamente, la validación y la verificación son dos actividades diferentes en el desarrollo de software. La verificación determina si el producto con las actividades de desarrollo cumple con sus requisitos (hacer lo correcto). La validación evalúa si un producto satisface las necesidades del cliente (haciendo lo correcto).

Al extender estas definiciones a los requisitos, la verificación determina si ha escrito correctamente los requisitos: sus requisitos tienen las propiedades deseables. La validación de los requisitos evalúa si se han escrito los requisitos adecuados: se remontan a los objetivos de negocio.

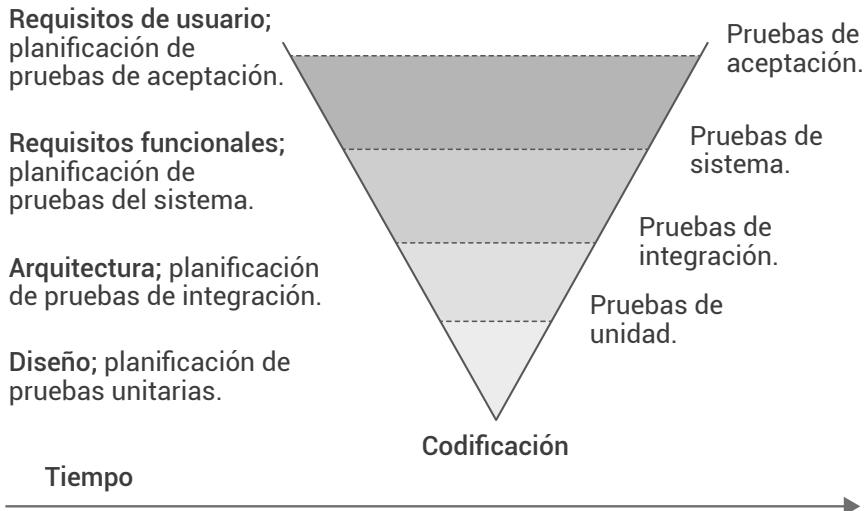
Estos dos conceptos están estrechamente entrelazados. Para simplificar en este capítulo, hablamos de validar los requisitos, pero las técnicas que describimos contribuyen tanto a tener los requisitos correctos como a tener requisitos de alta calidad.

Para conocer al detalle los conceptos asociados a la verificación y validación, revise el recurso educativo abierto: [Verificación y Validación](#) del curso Fundamentos de la Ingeniería de Sistemas.

Como podrá apreciar, existen diferentes tipos de pruebas que se pueden realizar al momento del desarrollo del producto software, así como la

adecuada gestión de los riesgos como estrategia de la validación. En los apartados siguientes se analizan cada uno de estos temas.

Figura 31
Modelo en V de desarrollo de software



Nota. Adaptado de *Software Requirements* (p. 330), por Wiegers, K. y Beatty, J., 2013. Microsoft Press.

En la figura 31 se muestra cómo las actividades de prueba comienzan en paralelo con las actividades de desarrollo. Se aprecia que las pruebas de aceptación se derivan de los requisitos del usuario, las pruebas del sistema se basan en los requisitos funcionales y las pruebas de integración se basan en la arquitectura del sistema. Este modelo es aplicable ya sea que las actividades de desarrollo de software que se prueban sean para el producto como un todo, una versión en particular o un solo incremento de desarrollo.

La validación de los requisitos permite a los equipos construir una solución correcta que cumpla con los objetivos de negocio establecidos. Las actividades de validación de requisitos intentan asegurar que:

- Los requisitos de software describen con precisión las capacidades y propiedades del sistema pretendido que satisfarán las necesidades de las diversas partes interesadas.
- Los requisitos de software se derivan correctamente de los requisitos empresariales, los requisitos del sistema, las reglas empresariales y otras fuentes.

- Los requisitos son completos, factibles y verificables.
- Todos los requisitos son necesarios, y todo el conjunto es suficiente para cumplir con los objetivos de negocio.
- Todas las representaciones de los requisitos son consistentes entre sí.
- Los requisitos proporcionan una base adecuada para proceder con el diseño y la construcción.

La validación no es una sola fase discreta que se realiza después de obtener y documentar todos los requisitos. Algunas actividades de validación, tales como revisiones incrementales de los requisitos, están adheridas a lo largo de los procesos iterativos de obtención, análisis y especificación. Otras actividades, como las inspecciones formales, proporcionan una puerta de calidad antes de basar un conjunto de requisitos. Incluya las actividades de validación de requisitos como tareas en su plan de proyecto. Por supuesto, solo se pueden validar los requisitos que se han documentado, no los requisitos implícitos que solo existen en la mente de alguien.

6.2. Revisión de los requisitos

En cualquier momento alguien que no sea el autor de un producto de trabajo examina el producto para determinar problemas, se está llevando a cabo una revisión por pares. Revisar los requisitos es una técnica poderosa para identificar requisitos ambiguos o no verificables, requisitos que no se definen claramente para que comience el diseño y otros problemas.

Diferentes tipos de revisión por pares van por una variedad de nombres. Las revisiones informales son útiles para educar a otras personas sobre el producto y recolectar retroalimentación no estructurada. Sin embargo, no son sistemáticos, minuciosos o realizados de manera consistente. Los enfoques de revisión informal incluyen:

- Un “*peer deskcheck*”, en el que se pide a un colega mirar por encima el producto de trabajo.
- Un “*pasoound*”, en el que se invita a varios colegas a examinar simultáneamente una entrega.
- Un “*walkthrough*”, durante el cual el autor describe una entrega y solicita comentarios sobre ella.

Las revisiones informales son buenas para detectar errores, inconsistencias y lagunas. Pueden ayudarle a localizar declaraciones que no cumplen con las características de los requisitos de alta calidad. Pero es difícil para un revisor captar todos los requisitos ambiguos por su cuenta. Él podría leer un requisito y pensar que lo entiende, pasando al siguiente sin un segundo pensamiento. Otro revisor puede leer el mismo requisito, llegar a una interpretación diferente, y tampoco pensar que hay un problema. Si estos dos revisores nunca discuten el requisito, la ambigüedad pasará desapercibida hasta más adelante en el proyecto.

Las evaluaciones formales por pares siguen un proceso bien definido. Una revisión de requisitos formales produce un informe que identifica el material examinado, los revisores y el juicio del equipo de revisión sobre si los requisitos son aceptables. El producto principal es un resumen de los defectos encontrados y los problemas planteados durante la revisión. Los miembros de un equipo de revisión formal comparten la responsabilidad por la calidad de la revisión, aunque en última instancia los autores son responsables de la calidad de los resultados que crean.

El tipo mejor establecido de revisión formal de pares se llama inspección. La inspección de documentos de requisitos es una de las técnicas de calidad de software de mayor apalancamiento disponibles. Varias compañías han evitado hasta 10 horas de trabajo por cada hora que invirtieron en la inspección de documentos de requisitos y otros productos de software.

La inspección detallada de grandes conjuntos de requisitos es tediosa y consume mucho tiempo. Sin embargo, los equipos que han adoptado inspecciones de requisitos están de acuerdo en que cada minuto que pasan vale la pena. Si no tiene tiempo para inspeccionar todo, utilice el análisis de riesgos para diferenciar los requisitos que exigen inspección de los menos críticos, menos complejos o nuevos para los que será suficiente una revisión informal. Las inspecciones no son baratas. Ni siquiera son tan divertidas. Pero son más baratos —y más divertidos— que la alternativa de gastar mucho esfuerzo y los problemas de buena voluntad del cliente que se encuentran mucho después.

A continuación, se detallan las estrategias de inspección, lista de defectos y algunos consejos para la revisión de requisitos.

6.2.1. Proceso de inspección

La inspección es un proceso bien definido y de múltiples etapas. Se trata de un pequeño equipo de participantes que examinan cuidadosamente un producto de trabajo por defectos y oportunidades de mejora. Las inspecciones sirven como una puerta de calidad a través de la cual las entregas del proyecto deben realizarse antes de que se defina la línea base. Hay varias formas de inspección, pero cualquiera de ellas es una técnica de gran calidad. La siguiente descripción se basa en la técnica de inspección de Fagan (Fagan, 2002).

1. Participantes

Asegúrese de tener todas las personas necesarias en una reunión de inspección antes de continuar. Los participantes en una inspección deben representar cuatro perspectivas:

- El autor del producto de trabajo y quizás los compañeros del autor. El analista de negocios que escribió el documento de requisitos proporciona esta perspectiva. Incluya otro analista de negocio experimentado si puede, porque sabrá del tipo de errores en los requisitos.
- Las personas que son las fuentes de información que alimentaron el elemento que se está inspeccionando. Estos participantes podrían ser representantes de usuarios reales o el autor de una especificación predecesora. En ausencia de una especificación de nivel superior, la inspección debe incluir representantes de los clientes, tales como *product-champions*, para asegurar que los requerimientos describen sus necesidades correcta y completamente.
- Personas que harán el trabajo basado en el ítem que se está inspeccionando. Para un ERS, puede incluir un desarrollador, un probador, un administrador de proyectos y un escritor de documentación del usuario porque detectarán diferentes tipos de problemas. Un probador es más probable para escoger un requisito no verificable. Un desarrollador puede detectar los requisitos que son técnicamente imposibles.
- Las personas responsables de los sistemas de interfaz que se verán afectados por el elemento que se está inspeccionando. Estos

inspectores buscarán problemas con los requisitos de la interfaz externa. También pueden detectar efectos de ondulación, en los que el cambio de un requisito en el ERS que se está inspeccionando afecta a otros sistemas.

Trate de limitar el equipo a siete o menos inspectores. Esto podría significar que algunas perspectivas no estarán representadas en cada inspección. Los grandes equipos se enredan fácilmente en discusiones paralelas, resolución de problemas y debates sobre si algo es realmente un error. Esto reduce la velocidad a la que cubren el material durante la inspección y aumenta el costo de detectar cada defecto.

El director normalmente no debe asistir a una reunión de inspección, a menos que este contribuyendo activamente al proyecto y su presencia es aceptable para el autor. Una inspección eficaz que revela muchos defectos podría crear una mala impresión del autor a un gerente hipercrítico. Además, la presencia del gerente podría estimular la discusión de otros participantes.

2. Roles de la inspección

Todos los participantes en una inspección, incluyendo el autor, buscan defectos y oportunidades de mejora. Algunos de los miembros del equipo de inspección desempeñan las siguientes funciones específicas durante la inspección.

- **Autor**

El autor crea o mantiene el producto de trabajo que está siendo inspeccionado. El autor de un documento de requisitos suele ser el analista de negocios que obtuvo las necesidades del cliente y escribió los requisitos. Durante las revisiones informales, tales como tutoriales, el autor a menudo lidera la discusión. Sin embargo, el autor asume un papel más pasivo durante una inspección. El autor no debe asumir ninguna de las otras funciones asignadas: moderador, lector o grabador. Al no tener un papel activo, el autor puede escuchar los comentarios de otros inspectores. De esta manera el autor puede detectar errores que otros inspectores no ven.

- **Moderador**

El moderador planifica la inspección con el autor, coordina las actividades y facilita la reunión de inspección. El moderador distribuye los materiales a inspeccionar, junto con cualquier documento predecesor relevante,

a los participantes unos días antes de la reunión de inspección. Las responsabilidades del moderador incluyen iniciar la reunión a tiempo, alentar las contribuciones de todos los participantes y mantener la reunión enfocada en detectar los principales defectos en lugar de resolver problemas o distraerse por problemas estilísticos menores y errores tipográficos. El moderador hace un seguimiento de los cambios propuestos con el autor para asegurarse de que las cuestiones que salieron de la inspección se abordaron adecuadamente.

- **Lector**

Durante la reunión de inspección, el lector parafrasea los requisitos y los elementos del modelo que se examinan uno a la vez. Los otros participantes señalan entonces los posibles defectos y problemas que ven. Al establecer un requisito en sus propias palabras, el lector proporciona una interpretación que puede diferir de la que poseen otros inspectores. Esta es una buena manera de revelar una ambigüedad, un posible defecto, o una suposición. También subraya el valor de tener a alguien que no sea el autor como lector. En los tipos menos formales de revisiones por pares, el papel del lector se omite, con el moderador.

- **Grabador**

El grabador usa formulario estándar para documentar los problemas planteados y los defectos encontrados durante la reunión. El grabador debe revisar en voz alta o visualmente compartir lo que escribió para confirmar su exactitud. Los otros inspectores deben ayudar al grabador a captar la esencia de cada asunto de una manera que comunique claramente al autor la ubicación y la naturaleza de la cuestión para que pueda abordarla de manera eficiente y correcta.

3. Criterio de entrada

Está listo para inspeccionar un documento de requisitos cuando satisface requisitos previos específicos. Estos criterios de entrada establecen algunas expectativas claras para los autores a seguir mientras se prepara para una inspección. También mantienen al equipo de inspección de pasar tiempo en asuntos que deben ser resueltos antes de la inspección. El moderador utiliza los criterios de entrada como una lista de verificación antes de decidir continuar con la inspección. A continuación, se presentan algunos criterios de entrada de inspección sugeridos para los documentos de requisitos:

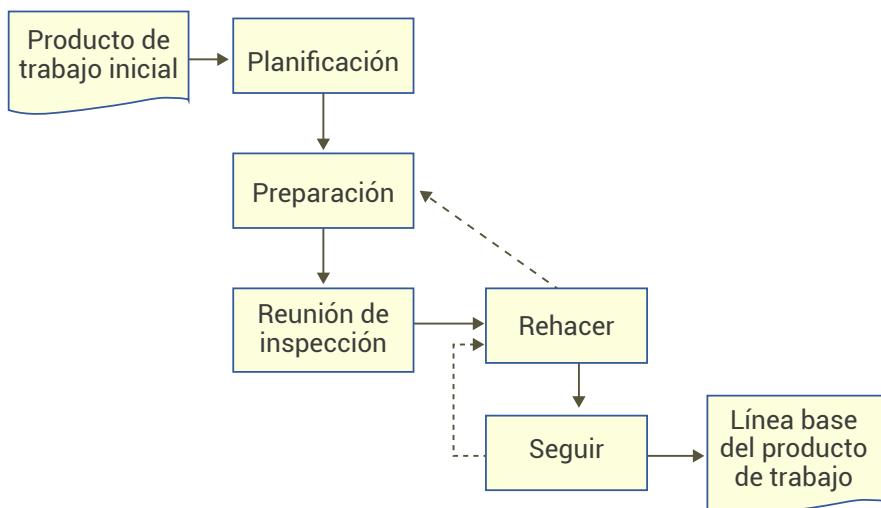
- El documento se ajusta a la plantilla estándar y no tiene problemas ortográficos, gramaticales o de formato obvios.
- Los números de línea u otros identificadores únicos se imprimen en el documento para facilitar la referencia a ubicaciones específicas.
- Todos los problemas abiertos están marcados como TBD (a determinar) o accesibles en una herramienta de seguimiento de problemas.
- El moderador no presentó más de tres defectos mayores en un examen de diez minutos de una muestra representativa del documento.

4. Etapas de la inspección

Una inspección es un proceso de varios pasos, como se ilustra en la figura 32. Puede inspeccionar pequeños conjuntos de requisitos a la vez, tal vez los asignados a una iteración de desarrollo específica, cubriendo con ello la totalidad de la recolección de requisitos.

Figura 32

Pasos de un proceso de Inspección



Nota. Adaptado de *Software Requirements* (p. 336), por Wiegers, K. y Beatty, J., 2013. Microsoft Press.

■ Planificación

El autor y el moderador planifican la inspección juntos. Ellos determinan quién debe participar, qué materiales los inspectores deben recibir antes de la reunión de inspección, el tiempo total de la reunión necesario para cubrir el material, y cuándo la inspección debe ser programada. El número de páginas revisadas por hora tiene un gran impacto en cuántos defectos se encuentran, debido a que ningún equipo tiene tiempo disponible para inspecciones de requisitos, una tasa de inspección adecuada basada en el riesgo de pasar por alto los defectos mayores. Dos a cuatro páginas por hora es una guía práctica, aunque la tasa óptima para la máxima eficacia de detección de defectos es aproximadamente la mitad de esa tasa. Ajuste esta tasa basándose en los siguientes factores:

- Los datos de inspección anteriores del equipo, que muestran la efectividad de la inspección en función de la tasa.
- La cantidad de texto en cada página.
- La complejidad de los requisitos.
- La probabilidad y el impacto de tener errores sin ser detectados.
- Qué tan crítico es el material que se está inspeccionando para el éxito del proyecto.
- El nivel de experiencia de la persona que escribió los requisitos.

■ Preparación

Antes de la reunión de inspección, el autor debe compartir información de antecedentes con los inspectores para que entiendan el contexto de los elementos que se están inspeccionando y conocer los objetivos del autor para la inspección. Cada inspector examina el producto para identificar posibles defectos y problemas, utilizando la lista de verificación de los defectos de requisitos típicos u otras técnicas de análisis. Hasta el 75 por ciento de los defectos encontrados por una inspección se descubren durante la preparación, así que no omita este paso. Planee dedicar al menos la mitad de tiempo a la preparación individual para las reuniones de inspección del equipo.

■ Reunión de inspección

Durante una reunión de inspección, el lector lleva a los otros inspectores a través del documento, describiendo un requisito a la vez con sus propias palabras. A medida que los inspectores plantean posibles defectos y otros

problemas, el grabador los captura en la lista de elementos de acción para el autor de los requisitos. El propósito de la reunión es identificar tantos defectos importantes como sea posible. La reunión de inspección no debe durar más de dos horas; las personas cansadas no son inspectores eficaces. Si necesita más tiempo para cubrir todo el material, programe reuniones adicionales.

Después de examinar todo el material, el equipo decide si acepta el documento de requisitos tal como está, lo acepta con revisiones menores o indica que se necesita una revisión mayor. Un resultado de la “revisión mayor necesaria” podría sugerir que el proceso de desarrollo de requisitos tiene algunas deficiencias o que el analista de negocio que redactó los requisitos necesita capacitación adicional. Considere la posibilidad de realizar una retrospectiva para explorar cómo se puede mejorar el proceso antes de la siguiente actividad de especificación. Si se requieren revisiones importantes, el equipo puede optar por reexaminar partes del producto que requieran una reelaboración extensa.

A veces, los inspectores solo reportan problemas superficiales. Además, los inspectores se desvían fácilmente a discutir si un problema es realmente un defecto, debatiendo cuestiones de alcance del proyecto, y soluciones de lluvia de ideas a los problemas. Estas actividades pueden ser útiles, pero distraen la atención del objetivo principal de encontrar defectos significativos y oportunidades de mejora.

- **Rehacer**

Casi todas las actividades de control de calidad revelan algunos defectos. El autor debe planear pasar algún tiempo reelaborando los requisitos después de la reunión de inspección. Los defectos de requisito no corregidos serán costosos, por lo que este es el momento de resolver las ambigüedades, eliminar la borrosidad y sentar las bases para un proyecto de desarrollo exitoso.

- **Seguir**

En este paso de inspección final, el moderador o un individuo designado trabaja con el autor para asegurarse de que todos los problemas abiertos se resolvieron y que los errores se corrigieron correctamente. El seguimiento lleva al cierre del proceso de inspección y permite al moderador determinar si los criterios de salida de la inspección han sido satisfactorios. El paso

de seguimiento podría revelar que algunas de las modificaciones hechas fueron incompletas o no se realizaron correctamente, lo que condujo a una reelaboración adicional.

5. Criterio de salida

- El proceso de inspección debe definir los criterios de salida que deben ser satisfechos antes de que el moderador declare completo el proceso de inspección (no solo la reunión). Estos son algunos posibles criterios de salida para las inspecciones de requisitos:
 - Todas las cuestiones planteadas durante la inspección se han abordado.
 - Cualquier cambio realizado en los requisitos y productos de trabajo relacionados se realizó correctamente.
 - Se han resuelto todos los problemas abiertos o se ha documentado el proceso de resolución de cada problema abierto, la fecha de destino y el propietario.

6.2.2. Lista de verificación de defectos

Para ayudar a los revisores a buscar errores típicos en los productos que revisan, es necesario elaborar una lista de verificación de defectos para cada tipo de documento de requisitos que sus proyectos crean. Dichas listas de verificación llaman la atención de los revisores con los problemas frecuentes de los requisitos históricos. Las listas de verificación sirven como recordatorios. Con el tiempo, las personas internalizar los elementos y buscan los temas adecuados en cada revisión.

La figura 33 se aprecia una lista de verificación de revisión de requisitos. Si crea representaciones o modelos de requisitos particulares, puede ampliar los elementos de la lista de comprobación para que sean más exhaustivos. Los requisitos de negocio, como un documento de visión y alcance, podrían justificar su propia lista de verificación. En el artículo de Hoffman y Burgess (2009) proporcionan varias listas detalladas de revisión, incluyendo una para validar los requisitos de software en función de los requisitos del negocio.

Figura 33

Lista de verificación para revisión de requisitos

Completitud

- ¿Cumplen los requisitos todas las necesidades conocidas del cliente o del sistema?
- ¿Falta alguna información necesaria? Si es así, ¿se identifica como TBD (por determinar)?
- ¿Se han definido los algoritmos intrínsecos a los requisitos funcionales?
- ¿Se definen todas las interfaces de hardware, software y comunicación externas?
- ¿Se documenta el comportamiento esperado para todas las condiciones de error anticipadas?
- ¿Proporcionan los requisitos una base adecuada para el diseño y la prueba?
- ¿Se incluye la prioridad de implementación de cada requisito?
- ¿Está cada requisito dentro del alcance del proyecto, la versión o la iteración?"

Corrección

- ¿Hay requisitos que entran en conflicto o dupliquen otros requisitos?
- ¿Cada requisito está escrito en lenguaje claro, conciso, sin ambigüedades y gramaticalmente correcto?
- ¿Cada requisito es verificable mediante pruebas, demostración, revisión o análisis?
- ¿Son claros y significativos todos los mensajes de error especificados?
- ¿Son todos los requisitos realmente requisitos y no soluciones o restricciones?
- ¿Son los requisitos técnicamente factibles e implementables dentro de las restricciones conocidas?

Atributos de calidad

- ¿Se especifican correctamente todos los objetivos de usabilidad, rendimiento, seguridad y seguridad?
- ¿Se documentan y cuantifican otros atributos de calidad, con los compromisos aceptables especificados?
- ¿Se identifican las funciones críticas en el tiempo y se especifican los criterios de tiempo para ellas?
- ¿Se han abordado adecuadamente los problemas de internacionalización y localización?
- ¿Son medibles todos los requisitos de calidad?

Organización y Trazabilidad

- ¿Están los requisitos organizados de manera lógica y accesible?
- ¿Son correctas todas las referencias cruzadas a otros requisitos y documentos?
- ¿Están todos los requisitos escritos a un nivel de detalle coherente y apropiado?
- ¿Cada requisito está etiquetado de manera única y correcta?
- ¿Cada requisito funcional se rastrea hasta su origen (por ejemplo, requisito del sistema, regla empresarial)?

Otras cuestiones

- ¿Faltan algunos casos de uso o flujos de proceso?
- ¿Faltan algunos flujos alternativos, excepciones u otra información en los casos de uso?
- ¿Se han identificado todas las reglas de negocio?
- ¿Hay algún modelo visual faltante que proporcione claridad o completitud?
- ¿Están presentes y completas todas las especificaciones necesarias de informes?

Nota. Adaptado de *Software Requirements* (p. 339), por Wiegers, K. y Beatty, J., 2013. Microsoft.

Nadie puede recordar todos los elementos de una larga lista. Si hay más de seis u ocho elementos en la lista, un revisor probablemente tendrá que hacer varias pasadas a través del material para buscar todo en la lista; la mayoría de los críticos no se molestan. Pare las listas para satisfacer las necesidades de su organización y modifique los elementos para que reflejen los problemas que la gente encuentra con más frecuencia con sus propios requisitos. Algunos estudios han demostrado que dar a los revisores responsabilidades específicas de detección de defectos —proveer procesos de pensamiento estructurados o escenarios para ayudarlos a buscar tipos particulares de errores— es más efectivo que simplemente entregar a todos los revisores la misma lista de verificación y esperar lo mejor.

En el [anexo 7 Lista de verificación](#), se muestra una lista de verificación que permite validar la especificación de requisitos de software propuesto por Gottesdiener (2005).

6.2.3. Consejos para la revisión de requisitos

Los siguientes consejos se aplican tanto si está realizando revisiones formales o informales de sus proyectos, como si está almacenando sus requisitos en documentos tradicionales, en una herramienta de gestión de requisitos o en cualquier otra forma tangible.

- **Planifique el examen**

Cuando alguien le pide que revise un documento, la tentación es comenzar en la parte superior de la página uno y leerlo de principio a fin. Pero no necesitas hacer eso. Los consumidores de la especificación de requisitos no la leerán de principio a fin como un libro; los revisores tampoco tienen que hacerlo. Invite a determinados revisores a centrarse en secciones específicas de los documentos.

- **Comenzar temprano**

Comience a revisar los requisitos cuando tal vez solo estén completos en un 10 por ciento, no cuando crea que están “terminados”. La detección temprana de defectos importantes y la detección de problemas sistemáticos en la forma en que se escriben los requisitos es una forma poderosa de prevenir, no solo encontrar, defectos.

- **Asignar tiempo suficiente**

Dé a los revisores tiempo suficiente para realizar las revisiones, tanto en términos de horas reales para revisar (esfuerzo) como de tiempo calendario. Tienen otras tareas importantes en las que la revisión tiene que encajar.

- **Proporcione contexto**

Proporcione a los revisores contexto para el documento y tal vez para el proyecto si no están todos trabajando en el mismo proyecto. Busque revisores que puedan proporcionar una perspectiva útil basada en su conocimiento. Por ejemplo, es posible que conozca a un compañero de trabajo de otro proyecto que tenga buen ojo para encontrar las principales lagunas en los requisitos, incluso sin estar íntimamente familiarizado con el proyecto.

- **Establezca el alcance de la revisión**

Indique a los revisores qué material examinar, dónde centrar su atención y qué problemas buscar. Sugiera que usen una lista de verificación de defectos como la descrita en la sección anterior. Es posible que desee maximizar la disponibilidad y las habilidades solicitando a diferentes revisores que revisen diferentes secciones o que utilicen diferentes partes de las listas de verificación.

- **Límite las revisiones**

No le pida a nadie que revise el mismo material más de tres veces. Estará cansado de mirarlo y no detectará problemas importantes después de un tercer ciclo debido a la “fatiga del revisor”. Si necesita que alguien lo revise varias veces, resalte los cambios para que pueda concentrarse en ellos.

- **Priorizar las áreas de revisión**

Priorice la revisión de aquellas partes de los requisitos que son de alto riesgo o tienen una funcionalidad que se usará con frecuencia. Además, busque áreas de los requisitos que ya tengan pocos problemas registrados. Puede ser que esas secciones aún no hayan sido revisadas, no que estén libres de problemas.

Una vez revisados los contenidos sobre la revisión de los requisitos, estimado estudiante, es necesario continuar con el estudio de los siguientes temas de: prototipos, pruebas y validaciones de los requisitos.

6.3. Prototipos

Los prototipos son herramientas de validación que hacen que los requisitos sean reales. Permiten al usuario experimentar algunos aspectos de lo que sería un sistema basado en los requisitos. Los prototipos pueden ayudar a los interesados a juzgar si un producto construido está de acuerdo con los requisitos y si satisfacen sus necesidades, además si los requisitos son completos, factibles y claramente comunicados.

Todo tipo de prototipos le permiten encontrar los requisitos que faltan antes de realizar actividades más costosas como el desarrollo y las pruebas. Algo tan simple como una maqueta de papel se puede utilizar para caminar a través de casos de uso, procesos o funciones para detectar cualquier requisito omitido o erróneo. Los prototipos también ayudan a que las partes interesadas tengan una comprensión compartida de los requisitos. Alguien podría implementar un prototipo basado en su comprensión de los requisitos, solo para saber que un requisito no estaba claro cuando los evaluadores de prototipo no están de acuerdo con su interpretación.

Los prototipos de prueba de concepto pueden demostrar que los requisitos son factibles. Prototipos evolutivos permiten a los usuarios ver cómo funcionan los requisitos cuando se implementan, para validar que el resultado es lo que esperan. Niveles adicionales de sofisticación en prototipos, como simulaciones, permiten una validación más precisa de los requisitos; sin embargo, la construcción de prototipos más sofisticados también tomará más tiempo. De acuerdo a Gottesdiener (2005), existen tres tipos de prototipos:

1. **Prototipo de la interfaz del usuario:**

Es un modelo evolutivo de cómo va quedando la interfaz del usuario con base en los requisitos o necesidades planteadas. Puede ser desarrollado en papel, ejecutable del sistema o proyecto en desarrollo o software específico que se encuentran libremente en el mercado.

2. Prototipos de rendimiento:

Estos no incluyen la interfaz del usuario. Va orientado a los desarrolladores, con el fin de comprobar la funcionalidad de los componentes de la solución que se está desarrollando. No se aplica a la Ingeniería de Requisitos.

3. Prototipo funcional:

Es una versión limitada de la solución desarrollada. No se aplica a la Ingeniería de Requisitos.

El proceso para desarrollar las pruebas mediante prototipos, consiste en:

1. Determine qué requisitos validar utilizando prototipos.

- Elija los requisitos que representan mayor riesgo (como es: requisitos que no pueden ser factibles, reglas de negocio complejas, o algoritmos) o necesidades que representan requisitos de usabilidad crítica. Escoja casos de uso o escenarios de prioridad alta que permitan validar la usabilidad de los requisitos.
- Asegúrese que los requisitos elegidos son bien entendidos.

2. Desarrolle el prototipo

- Decida entre desarrollar un prototipo desecharable o evolutivo (Un prototipo desecharable es fácil y rápido de construir, pero al final se desecha. Un prototipo evolutivo requiere habilidades de desarrollo y más esfuerzo, pero podría involucrar un entregable del producto).
- Usar datos reales y escenarios definidos por el usuario.
- Construir el prototipo de forma iterativa (por ejemplo, desarrollar un pequeño prototipo para empezar a evaluarlo con el usuario, revisarlo como sea necesario antes de seguir adicionando funcionalidades).

3. Evalúe el prototipo

- Identificar los usuarios para la prueba del prototipo. Asegurar con claridad el propósito del prototipo con el usuario antes de empezar.
- Dirigir una demostración. Para prototipos de interfaz de usuario que los usuarios intenten usar el prototipo. Pídale que usen escenarios o tareas para guiar sus pruebas.
- Haga preguntas preparadas tales como: “¿Tiene sentido este siguiente paso?”, “¿el tiempo de respuesta es suficiente?”, “¿qué te detiene?”, “¿los mensajes tienen sentido?”, etc.
- Ejecute una prueba de la parte de construcción del sistema para prototipos que impliquen un intercambio de señales o datos entre componentes de *software* o *hardware*, para probar el rendimiento o un algoritmo complejo.
- Registre los problemas que surjan durante la evaluación. Revise la documentación de requisitos y prototipos para realizar los cambios necesarios.

6.4. Probando los requisitos

Las pruebas basadas en los requisitos funcionales o derivados de los requisitos del usuario ayudan a que los comportamientos esperados del sistema sean tangibles para los participantes del proyecto. El simple hecho de diseñar pruebas revelará muchos problemas con los requisitos mucho antes de que pueda ejecutar esas pruebas en el *software* en ejecución. La escritura de pruebas funcionales cristaliza su visión de cómo el sistema debe comportarse bajo ciertas condiciones. Los requisitos imprecisos y ambiguos saltarán porque no podrás describir la respuesta esperada del sistema. Cuando los analistas de negocio, los desarrolladores y los clientes pasen por las pruebas, lograrán una visión compartida de cómo funcionará el producto y aumentarán su confianza en que los requisitos son correctos. Las pruebas son una herramienta poderosa para validar y verificar los requisitos.

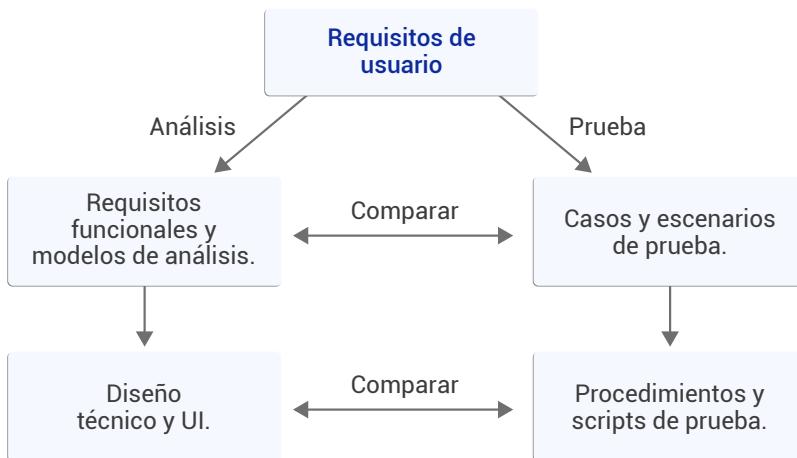
Puede comenzar a obtener pruebas conceptuales de los requisitos del usuario a principios del proceso de desarrollo. Utilice las pruebas para evaluar requisitos funcionales, modelos de análisis y prototipos. Las pruebas deben cubrir el flujo normal de cada caso de uso, flujos alternativos y las excepciones que identificó durante la obtención y el análisis. Del mismo modo, si identificó flujos de procesos de negocio, las pruebas deberían cubrir los pasos del proceso de negocio y todas las rutas de decisión posibles.

Estas pruebas conceptuales son independientes de la implementación. Por ejemplo, considere un caso de uso llamado “Visualizar una orden almacenada” para el Sistema de Seguimiento Químico. Algunas pruebas conceptuales son:

- El usuario introduce el número de pedido para ver, el pedido existe, el usuario ha realizado el pedido. Resultado esperado: mostrar los detalles de la orden.
- El usuario introduce el número de pedido para ver, el pedido no existe. Resultado esperado: Mostrar mensaje “Lo siento, no puedo encontrar esa orden.”
- El usuario introduce el número de pedido para ver, el pedido existe, el usuario no ha realizado el pedido. Resultado esperado: Mostrar mensaje “Lo siento, no es tu pedido”.

Figura 34

Desarrollo y pruebas de los productos de trabajo



Nota. Adaptado de *Software Requirements* (p. 344), por Wiegers, K. y Beatty, J., 2013. Microsoft.

Idealmente, un analista de negocio escribirá los requisitos funcionales y un probador escribirá las pruebas desde un punto de partida común: los requisitos del usuario, como se muestra en la figura 34. Las ambigüedades en los requisitos de los usuarios y las diferencias de interpretación darán lugar a inconsistencias entre las opiniones representadas por los requisitos funcionales, los modelos y las pruebas. A medida que los desarrolladores traducen los requisitos en interfaz de usuario y diseños técnicos, los probadores pueden elaborar las pruebas conceptuales en procedimientos de prueba detallados.

6.5. Validación de requisitos con criterios de aceptación

Los desarrolladores de software pueden creer que han construido el producto perfecto, pero el cliente es el árbitro final. Los clientes deben evaluar si un sistema satisface sus criterios de aceptación preestablecidos. Los criterios de aceptación —y, por lo tanto, las pruebas de aceptación— deben evaluar si el producto satisface sus requisitos documentados y si se debe utilizar en el entorno operativo previsto. Tener usuarios que diseñen pruebas de aceptación es una valiosa contribución para el desarrollo de requisitos eficaces. Cuanto antes se escriban las pruebas de aceptación, más pronto podrán ayudar al equipo a detectar defectos en los requisitos y, en última instancia, en el software implementado.

6.5.1. Criterios de aceptación

Trabajar con clientes para desarrollar criterios de aceptación proporciona una manera de validar tanto los requisitos como la propia solución. Si un cliente no puede expresar cómo evaluaría la satisfacción del sistema de un requisito particular, ese requisito no es suficientemente claro. Los criterios de aceptación definen las condiciones mínimas para que una solicitud sea considerada lista para la empresa.

Pensar en los criterios de aceptación ofrece un cambio de perspectiva desde la pregunta “¿Qué necesita hacer con el sistema?” a “¿Cómo juzgaría si la solución satisface sus necesidades?” Anime a los usuarios a usar el SMART mnemónico –Específico, Mensurables, Alcanzables, Relevantes y Sensibles al tiempo– al definir criterios de aceptación. Los criterios deben especificarse de tal manera que múltiples observadores objetivos lleguen a la misma conclusión acerca de si están satisfechos.

Los criterios de aceptación mantienen el enfoque en los objetivos empresariales de los interesados y las condiciones que permitirían al patrocinador del proyecto declarar la victoria. Esto es más importante que simplemente cumplir con una especificación de requisitos que podría no resolver realmente los problemas de los negocios de los interesados.

Definir criterios de aceptación es algo más que decir que todos los requisitos se implementan o que todas las pruebas pasan. Las pruebas de aceptación constituyen solo un subconjunto de criterios de aceptación. Los criterios de aceptación también podrían abarcar dimensiones como las siguientes:

- Funcionalidad específica de alta prioridad que debe estar presente y operar correctamente antes de que el producto pueda ser aceptado y utilizado. (Otra funcionalidad planificada podría ser entregada más tarde, o las capacidades que no están funcionando bien podrían ser arregladas sin retrasar una versión inicial).
- Criterios esenciales, no funcionales o métricas de calidad que deben cumplirse. (Algunos atributos de calidad deben ser por lo menos mínimamente satisfechos, aunque se podrían diferir las mejoras de la usabilidad y el ajuste de rendimiento. El producto podría tener que cumplir con métricas de calidad tales como una duración mínima de uso operativo sin experimentar un fallo).

- Problemas abiertos restantes y defectos. (Podría estipular que ningún defecto que exceda un determinado nivel de severidad permanezca abierto en contra de los requisitos de alta prioridad, aunque todavía podrían estar presentes errores menores).
- Condiciones legales, reglamentarias o contractuales específicas. (Estos deben estar completamente satisfechos antes de que el producto se considere aceptable).
- Apoyo a la transición, la infraestructura u otros requisitos del proyecto (no del producto). Tal vez los materiales de capacitación deben estar disponibles y las conversiones de datos deben completarse antes de que se pueda liberar la solución.

También puede ser valioso pensar en “criterios de rechazo”, condiciones o resultados de evaluación que llevaría a un interesado a considerar que el sistema aún no está listo para ser entregado. Tenga cuidado con los criterios de aceptación de acuerdo, de tal manera que encontrar uno podría bloquear la satisfacción de otro, de hecho, la búsqueda temprana de criterios de aceptación es una forma de descubrir requisitos contradictorios.

Los proyectos ágiles crean criterios de aceptación basados en historias de usuarios. Los criterios de aceptación no son funcionales o pruebas unitarias; Más bien, son las condiciones de satisfacción que se colocan en el sistema. Las pruebas funcionales y de unidad van mucho más profundamente al probar todos los flujos funcionales, flujos de excepción, condiciones de contorno y funcionalidad relacionada asociada con la historia.

En principio, si se cumplen todos los criterios de aceptación de una historia de usuario, el propietario del producto aceptará la historia de usuario como completada. Por lo tanto, los clientes deben ser muy específicos en la escritura de los criterios de aceptación que son importantes para ellos.

En Gottesdiener (2005), se indica los pasos a seguir para hacer las pruebas, las mismas que constan de:

- Defina los criterios de aceptación para el sistema.
 - a. Identifique la funcionalidad, atributos de calidad y datos correctos necesarios para que los clientes acepten el sistema.
 - b. Pregunte a los usuarios “¿Cómo juzgará si el sistema satisface sus necesidades?”

- Defina los casos de prueba de aceptación.

oLos casos de prueba son los datos de entrada y los resultados esperados. Cada criterio de aceptación podría tener uno o más casos de prueba. Los escenarios creados durante el modelado de análisis son buenas fuentes para los casos de prueba.

- Determine los métodos de prueba de aceptación.

oConsidere utilizar métodos de pruebas comunes, como los que se indica en la siguiente tabla.

Tabla 11
Métodos de prueba de aceptación

Método	Explicación
Pruebas manuales	Casos de prueba escritos en papel y recorrer manualmente a través de los pasos utilizando los modelos de análisis.
Herramientas de pruebas con Interfaz Gráfica de Usuario	Herramientas que ejecutan el sistema mientras graban las acciones del usuario y el sistema responde.
Código y prueba	El código escrito por los desarrolladores para ejecutar una prueba, a menudo ayudado con un marco de pruebas que ayudan a manejar la ejecución de una o más pruebas.
Secuencia de comandos	Forma simplificada de código escrito por desarrolladores o usuarios, que utilizan una notación específica.
Hoja de cálculo	Hoja de cálculo creada con el valor de datos en columnas, con una columna adicional para resultados esperados.
Plantilla	Una combinación de secuencia, comandos y hoja de cálculo.

Nota. Adaptado de *Métodos de pruebas de aceptación* (p. 271), por E. Gottesdiener, 2005, GOAL/QPC

- Valide los modelos de análisis utilizando las pruebas de aceptación de usuario.
 - Los fallos en las pruebas de aceptación pueden tener diferentes niveles de severidad, tal como se indica en el ejemplo que se indica a continuación.

Tabla 12*Criterios de severidad*

Nivel de severidad	Definición
1	Crítico: es imposible continuar con las pruebas o aceptar el sistema a causa de este error.
2	Alto: las pruebas pueden continuar, pero el sistema no se puede implementar con este problema.
3	Mediano: las pruebas pueden continuar y el sistema es probable que se siga desarrollado con algunas salidas de funcionalidad del negocio acordado.
4	Mínimo: las pruebas y el despliegue pueden progresar. El problema debería ser corregido, pero no afectará la funcionalidad de negocio.
5	Visual: los errores como colores, fuentes, y las demostraciones de interfaz que son menos que deseables pueden ser corregidos en algún futuro tiempo.

Nota. Adaptado de *Ejemplo de niveles de severidad para las pruebas de aceptación* (p. 272), por E. Gottesdiener, 2005, GOAL/QPC

6.5.2. Pruebas de aceptación

Las pruebas de aceptación constituyen la mayor parte de los criterios de aceptación. Los creadores de pruebas de aceptación deben considerar los escenarios de uso más comúnmente realizados y los más importantes al decidir cómo evaluar la aceptabilidad del software. Centrarse en probar los flujos normales de los casos de uso y sus correspondientes excepciones, dedicando menos atención a los flujos alternativos menos utilizados.

Los enfoques de desarrollo ágil suelen crear pruebas de aceptación en lugar de escribir requisitos funcionales precisos. Cada prueba describe cómo debe funcionar una historia de usuario en el software ejecutable. Debido a que están reemplazando en gran medida los requisitos detallados, las pruebas de aceptación en un proyecto ágil deben cubrir todos los escenarios de éxito y fracaso (Leffingwell, 2010).

El valor en las pruebas de aceptación escritas es que guía a los usuarios a pensar en cómo se comportará el sistema después de su implementación. El problema de escribir solo pruebas de aceptación es que los requisitos solo existen en la mente de la gente.

Al no documentar y comparar vistas alternativas de requisitos —requisitos de usuario, requisitos funcionales, modelos de análisis y pruebas—, puede perder la oportunidad de identificar errores, inconsistencias y lagunas.

¿Qué le pareció la temática abordada? Interesante verdad, ahora le invito a que realice las siguientes actividades de aprendizaje.



Actividades de aprendizaje recomendadas

Estimado estudiante finalizada la presente unidad, le recomiendo realizar las siguientes actividades con el objeto de profundizar los conocimientos.

1. Elabore una infografía indicando los pasos para realizar el proceso de validación de requisitos.
2. Desarrolle un plan para desarrollar inspección mediante prototipos.

Para realizar estas actividades deberá:

- Revise los temas relacionados con los pasos que se deben desarrollar para la validación en la unidad 7.
- Con base en el ERS del [anexo 4 Descripción de la plantilla ERS](#), elabore un prototipo utilizando la herramienta Figma que le permita validar los requisitos.



3. Finalmente, realice la autoevaluación 6.



Autoevaluación 6

1. El proceso de validación de requisitos consiste en:
 - a. Planificar y ejecutar prototipos.
 - b. Seleccionar e integrar técnicas, participación del usuario, validar los requisitos y revisar la documentación.
 - c. Revisar la documentación.
2. La participación adecuada del usuario en la etapa de validación de requerimientos requiere que:
 - a. Se definan los interesados.
 - b. Se clasifique los interesados.
 - c. Los interesados deben revisar la documentación para asegurarse que los requisitos están completos y son de calidad.
3. El costo de corregir un error en las etapas finales es de:
 - a. 1 a 5 veces.
 - b. 10 a 100 veces.
 - c. 95 a 100 veces.
4. Cuando se requiere revisar y documentar los requerimientos se debe realizar:
 - a. Revisión de pares.
 - b. Crear pruebas de validación.
 - c. Mostrar partes del sistema.
5. Cuando se requiere validar los modelos, se debe:
 - a. Realizar pruebas sobre modelos de requerimientos.
 - b. Revisión de pares.
 - c. Mostrar partes del sistema.

6. Cuando se requiere realizar pruebas de aceptación por parte del usuario, se debe:
 - a. Realizar pruebas sobre modelos de requerimientos.
 - b. Crear pruebas de validación.
 - c. Mostrar partes del sistema.
7. La revisión de requisitos lo realiza:
 - a. Un grupo de personas que lee, analiza y discute el documento de especificación de requisitos.
 - b. Un proceso automatizado con la ayuda del analista.
 - c. El equipo de desarrolladores (programadores).
8. Se utilizan simulaciones de pruebas en lugar de casos de prueba real para pasar por múltiples modelos de análisis. Esto se denomina:
 - a. Modelos de validación.
 - b. Prototipos operacionales.
 - c. Prototipos visuales.
9. Permite demostrar si los modelos son compatibles entre ellos:
 - a. Prototipos operacionales.
 - b. Modelos de validación.
 - c. Pruebas de aceptación.
10. El proceso de prototipado consiste en:
 - a. Desarrollar el prototipo.
 - b. Validar el prototipo.
 - c. Determinar los requisitos, desarrollar y evaluar el prototipo.

[Ir al solucionario](#)

Resultado de aprendizaje 5

- Utiliza estrategias y herramientas apropiadas para llevar una gestión apropiada de las actividades de gestión de requisitos.

Este resultado de aprendizaje está orientado conocer y aplicar adecuadamente las estrategias de gestión del cambio de los requisitos, cuando por cualquier motivo se requiera hacer un ajuste luego de haber definido formalizado el documento de especificación de requisitos de software. Esto se lo realiza definiendo la línea base y el desarrollo de las actividades de: control de cambios, seguimiento y trazabilidad de los requisitos.

Contenidos, recursos y actividades de aprendizaje



Semana 15

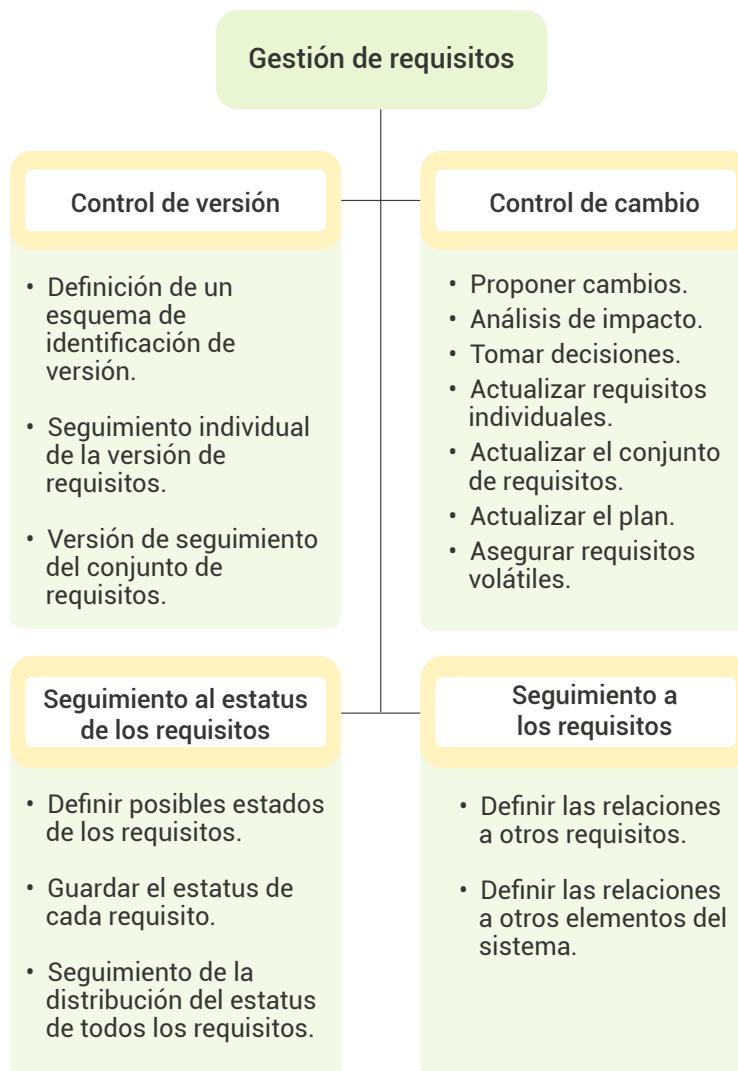
Unidad 7. Gestión de requisitos

7.1. Proceso de gestión de requisitos

La gestión de requisitos es el proceso de monitorear el estado de los requisitos y controlar los cambios a la línea base de los requisitos, incluye todas las actividades que permiten mantener la integridad, exactitud y coherencia de los acuerdos de requisitos a lo largo del proyecto.

Figura 35

Principales actividades de la gestión de requisitos



Nota. Adaptado de *Software Requirements* (p. 278). por Wiegers, K. y Beatty, J., 2013. Microsoft Press.

En la figura 35 se muestra las actividades principales de la gestión de requisitos en cuatro categorías principales: control de versiones, control de cambios, seguimiento de estado de requerimientos y rastreo de requerimientos.

La organización debe definir las actividades que los equipos de proyecto deben realizar para administrar sus necesidades. Documentar estas

actividades y capacitar a los profesionales en su desempeño permite a los miembros de la organización llevarlos a cabo de manera consistente y efectiva. Considere la posibilidad de abordar los siguientes temas:

- Herramientas, técnicas y convenciones para distinguir versiones de requisitos individuales y conjuntos de requisitos.
- La forma en que los conjuntos de requisitos se aprueban y se define su línea base.
- Las formas en que se proponen, se evalúan, negocian y se comunican nuevos requisitos y cambios a los ya existentes.
- Cómo evaluar el impacto de un cambio propuesto.
- Atributos de requisitos y procedimientos de seguimiento de estado de los requisitos, incluidos los estados de requisito que va a utilizar y quién puede cambiarlos.
- ¿Quién es responsable de actualizar la información de seguimiento de los requisitos y cuándo?
- Cómo rastrear y resolver problemas de requisitos.
- Cómo los planes y compromisos del proyecto reflejarán cambios en los requisitos.
- Cómo utilizar eficazmente la herramienta de gestión de requisitos.

Puede incluir toda esta información en una única descripción del proceso de gestión de requisitos. Como alternativa, es posible que prefiera escribir controles de versiones, controles de cambios, análisis de impacto y procedimientos de seguimiento del estado. Estos procedimientos deben aplicarse en toda la organización porque representan funciones comunes que cada equipo del proyecto debe realizar. Las descripciones de proceso deben identificar el rol del equipo que posee cada una de las actividades de gestión de requisitos.

El analista de negocios del proyecto normalmente tiene la responsabilidad principal de la gestión de requisitos. El analista de negocio establecerá los mecanismos de almacenamiento de requisitos, definirá los atributos de requisito, coordinará el estado de los requisitos y rastreará las

actualizaciones de datos y monitoreará la actividad de cambio según sea necesario. La descripción del proceso también debe indicar quién tiene autoridad para modificar el proceso de administración de requisitos, cómo se deben manejar las excepciones y la ruta de escalamiento para los impedimentos encontrados. (Wiegers y Beatty, 2013)

7.2. La línea base de los requisitos

El desarrollo de requisitos incluye actividades para obtener, analizar, especificar y validar los requisitos de un proyecto de software. Los entregables del desarrollo de requisitos incluyen requisitos de negocio, requisitos de usuario, requisitos funcionales y no funcionales, un diccionario de datos y varios modelos de análisis. Una vez revisados y aprobados, cualquier subconjunto definido de estos ítems constituye una línea de base de los requisitos.

Una base de referencia de los requisitos es un conjunto de requisitos que los interesados han acordado, a menudo definiendo el contenido de una liberación planificada específica o una iteración de desarrollo. El proyecto podría tener acuerdos adicionales en cuanto a entregas, restricciones, calendarios, presupuestos, requisitos de transición y contratos.

En el momento en que se establece un conjunto de requisitos, generalmente después de la revisión y aprobación, los requisitos se colocan bajo la administración de la configuración (o cambio). Los cambios subsiguientes solo pueden realizarse a través del procedimiento de control de cambios definido en el proyecto. Antes de la línea de base, los requisitos siguen evolucionando, por lo que no tiene sentido imponer costes indirectos innecesarios sobre esas modificaciones.

Una línea base podría consistir en algunos o todos los requisitos de un ERS en particular (ya sea para un producto completo o una sola versión), o un conjunto de requisitos designados almacenados en una herramienta de gestión de requisitos o un conjunto acordado de historias de usuarios para una única iteración en un proyecto ágil.

Si el alcance de una versión cambia, actualice la línea de base de los requisitos como corresponde. Distinguir los requisitos de una base de referencia particular de otros que fueron propuestos, pero no aceptados, se asignan a una línea de base diferente o permanecen sin asignar en la cartera

de productos. Si los requisitos se especifican en forma de un documento como un ERS, identifíquelo claramente como una versión de referencia para distinguirlo de los borradores anteriores.

El almacenamiento de requisitos en una herramienta de gestión de requisitos facilita la identificación de aquellos que pertenecen a una línea de base específica y el manejo de los cambios a esa línea de base.

Un equipo de desarrollo que acepta cambios o adiciones de requisitos propuestos podría no ser capaz de cumplir con su calendario y compromisos de calidad existentes. El gerente del proyecto debe negociar los cambios con los gerentes, clientes y otros interesados afectados. El proyecto puede acomodar requisitos nuevos o modificados de varias maneras:

- Difiriendo los requisitos de menor prioridad en iteraciones posteriores o cortarlos completamente.
- Obteniendo personal adicional o subcontratando parte del trabajo.
- Ampliando el calendario de entrega o añadiendo iteraciones a un proyecto ágil.
- Sacrificando la calidad para enviar en la fecha original.

Ningún enfoque individual es universalmente correcto, porque los proyectos difieren en su flexibilidad de características, personal, presupuesto, programación y calidad (Wiegert & Beatty, 2006).

La elección debe basarse en los objetivos de negocio del proyecto y en las prioridades establecidas por los principales interesados durante la iniciación del proyecto. No importa cómo responda a las necesidades cambiantes, acepte la realidad de ajustar las expectativas y compromisos cuando sea necesario. Esto es mejor que imaginar que de alguna manera todas las nuevas características serán incorporadas por la fecha de entrega original, sin excesos presupuestarios, miembros del equipo apagados o compromisos de calidad.

7.3. Control de versiones de requisitos

El control de versiones, que identifica de forma única las diferentes versiones de un ítem, se aplica tanto a los requisitos individuales como a los conjuntos de requisitos, representados más comúnmente en forma de

documentos. Comience el control de versión tan pronto como redacte un requisito o un documento para poder conservar un historial de cambios realizados.

Cada versión de los requisitos debe ser identificada de manera única. Cada miembro del equipo debe poder acceder a la versión actual de los requisitos. Los cambios deben ser claramente documentados y comunicados a todos los afectados. Para minimizar la confusión y la falta de comunicación, permita que solo personas designadas actualicen los requisitos y asegúrese de que el identificador de versión cambia cada vez que se realiza una actualización. Cada versión puesta en circulación de un documento de requisitos o cada requisito de una herramienta debe incluir un historial de revisiones que identifique los cambios realizados, la fecha de cada cambio, la persona que hizo el cambio y la razón de cada cambio.

El enfoque más sólido para el control de versiones es almacenar los requisitos en una herramienta de administración de requisitos. Las herramientas de gestión de requisitos rastrean el historial de cambios realizados en cada requisito, lo cual es valioso cuando se necesita volver a una versión anterior. Tal herramienta permite comentarios que describen la razón detrás de una decisión de agregar, modificar o eliminar un requisito. Estos comentarios son útiles si el requisito se convierte en un tema de discusión nuevamente en el futuro.

Si almacena los requisitos en los documentos, puede realizar un seguimiento de los cambios utilizando la función de marcas de revisión del procesador de textos. Esta característica resalta visualmente los cambios realizados en el texto con anotaciones como el tachado de resaltado para las supresiones y subrayados para las adiciones. Al basar un documento, primero archive una versión marcada, luego acepte todas las revisiones y guarde la versión ahora limpia como la nueva línea de base, lista para la siguiente ronda de cambios. Guarde los documentos de requisitos en una herramienta de control de versiones, como la que utiliza su organización para controlar el código fuente a través de procedimientos de *check-out* y *check-in*. Esto le permitirá volver a versiones anteriores si es necesario y saber quién cambió cada documento, cuándo y por qué.

El mecanismo de control de versiones más sencillo consiste en etiquetar manualmente cada revisión de un documento de acuerdo con una convención estándar. Los esquemas que tratan de diferenciar versiones de documentos basadas en fechas son propensos a la confusión. Utilizo

una convención que etiqueta la primera versión de cualquier nuevo documento con su título y “versión 1.0 del borrador 1”. El siguiente borrador conserva el mismo título, pero se identifica como “Versión 1.0 borrador 2.” El autor incrementa el número del borrador con cada iteración hasta que el documento sea aprobado y alineado. En ese momento, el identificador de versión se cambia a “Versión 1.0 aprobada”, manteniendo de nuevo el mismo título del documento. La siguiente versión es “Versión 1.1 borrador 1” para una revisión menor o “Versión 2.0 borrador 1” para un cambio importante. (Por supuesto, “mayor” y “menor” son subjetivos y dependen del contexto). Este esquema claramente distingue entre borrador y versiones de línea base del documento, pero requiere disciplina manual por parte de los que modifican los documentos.

7.4. Atributos de requisito

Piense en cada requisito como un objeto con propiedades que lo distinguen de otros requisitos. Además de su descripción textual, cada requisito debe tener piezas de apoyo de información o atributos asociados con ella. Estos atributos establecen un contexto y antecedentes para cada requisito. Puede almacenar valores de atributo en un documento, una hoja de cálculo, una base de datos o, lo más efectivamente posible, una herramienta de gestión de requisitos. Es engorroso usar más de un par de atributos de requisitos con documentos.

Las herramientas de gestión de requisitos normalmente proporcionan varios atributos generados por el sistema, además de permitirle definir otros, algunos de los cuales se pueden llenar automáticamente. Las herramientas permiten consultar en la base de datos para ver subconjuntos de requisitos seleccionados en función de sus valores de atributo. Por ejemplo, podría enumerar todos los requisitos de alta prioridad asignados a Peter (Programador) para su implementación en la versión 2.3 y tener un estado de aprobado. A continuación, se presenta una lista de atributos de requisitos potenciales a considerar:

- Fecha en que se creó el requisito.
- Número de versión actual del requisito.
- Autor que escribió el requisito.
- Prioridad.
- Estado.

- Origen o fuente del requisito.
- Fundamento del requisito.
- Número de versión o iteración a la que se asigna el requisito.
- Interesados que pueden ponerse en contacto con las preguntas o tomar decisiones sobre los cambios propuestos.
- Método de validación a utilizar o criterios de aceptación.

Los requisitos planificados para una liberación cambiarán a medida que se agreguen nuevos requisitos y se eliminen o aplacen los ya existentes. El equipo puede estar manipulando documentos de requisitos separados para varias versiones o iteraciones. Dejar los requisitos obsoletos en el ERS puede confundir a los lectores en cuanto a si esos requisitos, si están incluidos en esa línea base. Una solución es almacenar los requisitos en una herramienta de gestión de requisitos y definir un atributo “Número de publicación”. Deferir un requisito significa cambiar su lanzamiento planificado, de modo que simplemente actualizar el número de liberación cambia el requisito en una línea base diferente. Maneje los requisitos eliminados y rechazados usando un atributo de estado. Definir y actualizar estos valores de atributos es parte del costo de la gestión de requisitos, pero esa inversión que puede producir un retorno significativo.

7.5. Seguimiento al estado de los requisitos

El seguimiento de estado significa comparar donde realmente se encuentra en un momento dado contra la expectativa de lo que significa “completo” para este ciclo de desarrollo. Es posible que haya planeado implementar solo ciertos flujos de un caso de uso en la versión actual, dejando la implementación completa para una versión futura. Supervisar el estado de solo los requisitos funcionales que se comprometieron para la versión actual, porque ese es el conjunto que se supone que es 100 por ciento hecho antes de declarar el éxito y enviar el lanzamiento.

La tabla 13 enumera varios estados de requisitos posibles. Algunos profesionales añaden otros, como diseñado y entregados. Es valioso mantener un registro de los requisitos rechazados y las razones por las que fueron rechazados. Los requisitos rechazados tienen una forma de resurgir más tarde durante el desarrollo o en un proyecto futuro. El estado Rechazado le permite mantener un requisito propuesto disponible

para posible referencia futura sin desordenar el conjunto de requisitos específicos de una versión específica.

Tabla 13

Estado sugeridos para los requisitos

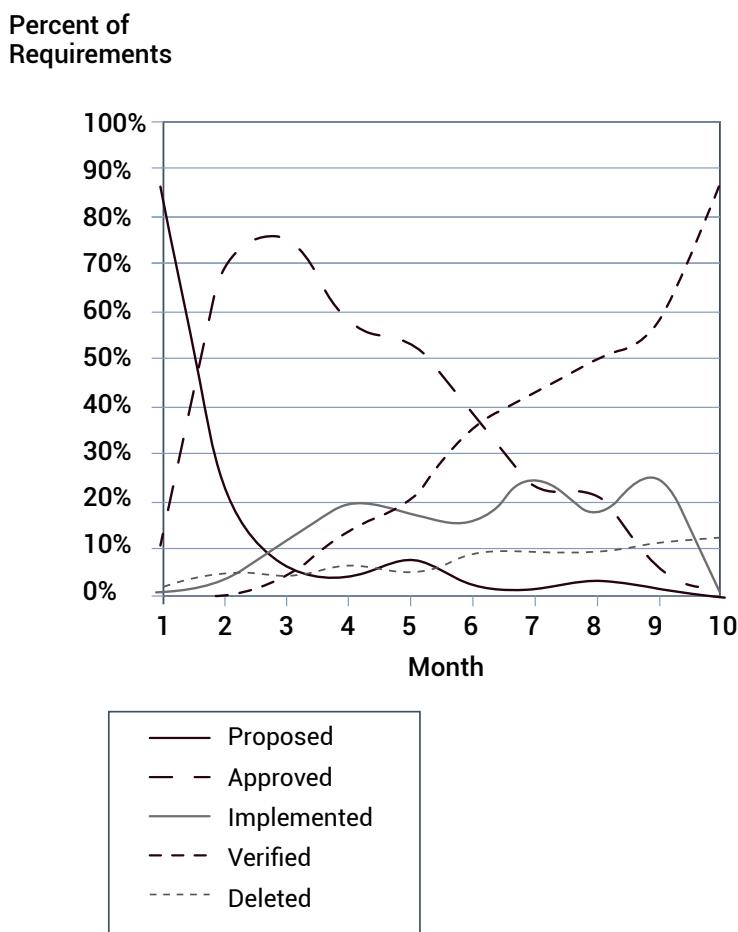
Status	Definition
Proposed	The requirement has been requested by an authorized source.
In Progress	A business analyst is actively working on crafting the requirement.
Drafted	The initial version of the requirement has been written.
Approved	The requirement has been analyzed, its impact on the project has been estimated, and it has been allocated to the baseline for a specific release. The key stakeholders have agreed to incorporate the requirement, and the software development group has committed to implement it.
Implemented	The code that implements the requirement has been designed, written, and unit tested. The requirement has been traced to the pertinent design and code elements. The software that implements the requirement is now ready for testing, review, or other verification.
Verified	The requirement has satisfied its acceptance criteria, meaning that the correct functioning of the implemented requirement has been confirmed. The requirement has been traced to pertinent tests. It is now considered complete.
Deferred	An approved requirement is now planned for implementation in a later release.
Deleted	An approved requirement has been removed from the baseline. Include an explanation of why and by whom the decision was made to delete it.
Rejected	The requirement was proposed but was never approved and is not planned for implementation in any upcoming release. Include an explanation of why and by whom the decision was made to reject it.

Nota. Adaptado de *Estados sugeridos que pueden adoptar los requisitos*. (p 221). K. Wiegers, 2013, Microsoft Press.

La clasificación de los requisitos en varias categorías de estado es más significativa que tratar de supervisar el porcentaje de finalización de cada requisito o de la línea base de liberación completa. Actualizar el estado de un requisito solo cuando se cumplen las condiciones de transición especificadas. Ciertos cambios de estado también requieren actualizaciones de los datos de rastreo de requisitos para indicar qué diseño, código y elementos de prueba trataron el requisito.

Figura 36

Seguimiento de la distribución del estado de los requisitos a lo largo del ciclo de desarrollo de un proyecto



Nota. Adaptado de *Seguimiento de la distribución del estado de los requisitos a lo largo del ciclo de desarrollo de un proyecto* (p. 466), por K. Wiegers y J. Beatty, 2013, Microsoft Press.

La figura 36 ilustra cómo puede monitorear visualmente el estado de un conjunto de requisitos a lo largo de un proyecto hipotético de 10 meses. Muestra el porcentaje de todos los requisitos del sistema que tienen cada valor de estado al final de cada mes. El seguimiento de la distribución por porcentajes no muestra si el número de requisitos en la línea base está cambiando con el tiempo. El número de requisitos aumenta a medida que se agrega el ámbito y disminuye cuando se elimina la funcionalidad de la línea base. Las curvas ilustran cómo el proyecto se está acercando a su objetivo

de verificación completa de todos los requisitos aprobados. Un cuerpo de trabajo se realiza cuando todos los requisitos que se le asignan tienen un estado verificado, eliminado o diferido.



Actividades de aprendizaje recomendadas

Estimado estudiante finalizada la presente unidad, le recomiendo realizar las siguientes actividades con el objeto de profundizar los conocimientos.

1. Utilice una herramienta para gestión de requisitos de acceso abierto y registre los requisitos funcionales y no funcionales, además de los casos de uso.
2. Indique ¿Qué actividades de gestión se puede realizar con la herramienta?

Estrategia de desarrollo:

- Busque herramientas para gestión de requisitos. Puede ser herramientas en la nube, para que no realice instalaciones locales.
- Considere la documentación del caso de uso y los requisitos funcionales como los no funcionales desarrollados en las unidades anteriores para registrarlos en la herramienta.
- Consulte en manuales y foros sobre el uso de la herramienta escogida.
- Realice una comparativa con la herramienta utilizada en Enterprise Architect utilizada en las unidades anteriores.



3. Finalmente, para afianzar sus conocimientos realice la autoevaluación 7.



Autoevaluación 7

1. Al proceso de seguimiento de la situación y control de cambios de los requisitos basados en una línea base, se conoce como:
 - a. Desarrollo de requisitos.
 - b. Mantenimiento de sistema.
 - c. Gestión de requisitos.
2. La correcta gestión de los requisitos permite:
 - a. Planificar el cronograma.
 - b. Gestionar los costos.
 - c. Minimizar los errores en las etapas posteriores a la especificación de requisitos.
3. Elegir una técnica de gestión de requerimientos depende de:
 - a. Tipo de proyecto.
 - b. Software para utilizar.
 - c. Interesados.
4. Contrastar y dar opiniones respecto a los requisitos que se han obtenido, se refiere a la característica de gestión de:
 - a. Análisis.
 - b. Negociación.
 - c. Especificación.
5. A la tarea de realizar un seguimiento de los requisitos, conociendo el ciclo de vida de los mismos, se conoce como:
 - a. Control de cambios.
 - b. Definir requisitos.
 - c. Trazabilidad.

6. Cuando se requiere identificar información suplementaria de los requisitos, se deben establecer:
 - a. Matrices de trazabilidad de requisitos.
 - b. Atributos de requisitos.
 - c. Gestión de cambios.
7. Realizar un análisis de impacto, como parte del proceso de cambios, ayuda a:
 - a. Identificar el proceso de control de cambios.
 - b. Implementar el proceso.
 - c. Crear la línea base.
8. La gestión de requisitos se la puede considerar como una actividad transversal del:
 - a. Ciclo de vida.
 - b. Implementación.
 - c. Prototipos.
9. Una de las características de las herramientas de gestión es:
 - a. Claridad de requisitos.
 - b. Número de requisitos.
 - c. Redundancia.
10. Los atributos de los requerimientos a los nuevos miembros les permite:
 - a. Conocer la organización.
 - b. Educarse acerca de los requisitos.
 - c. Desarrollar el proyecto.

[Ir al solucionario](#)



Semana 16



Actividades finales del bimestre

Hemos completado el estudio de los temas propuestos en el presente curso, que tienen que ver con el desarrollo y gestión de requisitos, esto con el objeto de especificar requisitos en el suficiente detalle y que cumplan con ciertas características que hacen que estos sean de calidad. Que pueda comprender que la importancia de la especificación de los requisitos radica en la comprensión de las actividades y luego su definición dependerá de lo que se pretende desarrollar. Que pueda entender que las plantillas y documentos son una guía para documentar el resultado de todo un proceso donde intervienen diferentes actores llamados Interesados y que su aporte es fundamental, que la labor del BA es la de analizar y canalizar la especificación acorde a las necesidades del negocio.

El desarrollo del caso y las actividades prácticas le van a permitir complementar los conceptos de cada tema y sobre todo luego cuando tenga situaciones reales las pueda canalizar de forma correcta. Aparte de los conocimientos que tenga como BA, otro de los elementos claves para el desarrollo de los proyectos es la experiencia, que irá adquiriendo cada vez que desarrolle los proyectos.

Lo invito a realizar un estudio de los temas relacionados con el análisis, especificación y validación de requisitos, lo que le permitirá disponer de una especificación debidamente documentada, como preparación para la evaluación del segundo bimestre, así como la participación en las actividades de planificadas en la plataforma educativa.



Actividad de aprendizaje recomendada

Analice detenidamente los casos de especificación de requisitos que se indican en la plataforma educativa. Compare los contenidos y determine cuál podría ser la mejor opción que podría usted adoptar en calidad de BA al a hora de desarrollar un proyecto de software.



4. Solucionario

Autoevaluación 1		
Pregunta	Respuesta	Retroalimentación
1	a	El usuario es el que define lo que el sistema debe realizar definiendo los requisitos, para apoyar las actividades en las organizaciones.
2	c	Para determinar la manera en que el sistema se comunicará con todas las entidades externas que interactúan para el desarrollo de las actividades, se requiere definir los requisitos de interfaz externa.
3	c	Inicialmente, se dispone de necesidades que son generales, pero que permite definir los objetivos del negocio, estas definiciones en la especificación se las conoce como requisitos de negocio.
4	b	El analista de negocio es el encargado del desarrollo de los requisitos, por lo tanto, empezará identificando las necesidades para posteriormente traducirlas a requisitos.
5	a	El analista es el responsable de las actividades del desarrollo del software, pero en el equipo se adoptan diferentes roles dependiendo de la actividad específica que desempeñan, por este motivo en el diseño e implementación el responsable es el analista de sistemas.
6	a	En analista de negocio es el responsable del desarrollo de los requisitos, por lo tanto, una de las actividades en este ámbito es la de Identificar a los Interesados.
7	c	Al momento que el cliente opta por un producto software, invierte en algo que será de beneficio y sobre todo solucione sus problemas, por ese motivo el producto le debe ser valioso, útil y cumpla con sus expectativas.
8	b	El primer acercamiento que tiene el analista de negocio es con el patrocinador, por lo tanto, los primeros requisitos (de negocio) se obtienen de esta persona.
9	a	En la primera fase del desarrollo de requisitos se debe identificar quienes son fuente de información, para que colaboren con el desarrollo de los requisitos. Los interesados.

Autoevaluación 1

Pregunta	Respuesta	Retroalimentación
10	b	En el análisis se identifican los requisitos al detalle, por lo tanto, la información que se obtiene debe ser debidamente analizada a un nivel de detalle aceptable para definir los requisitos.

[Ir a la
autoevaluación](#)

Autoevaluación 2		
Pregunta	Respuesta	Retroalimentación
1	a	El experto en seguridad participa directamente en el proyecto, por lo tanto, es un interesado. Aporta especialmente definiendo los requisitos no funcionales para diseñar la arquitectura de la aplicación.
2	b	Un proyecto de desarrollo de software no puede desarrollarse si no existe el propietario, porque es quien, como dueño, decide especialmente el alcance.
3	c	De manera general, a los interesados se los puede catalogar como internos, aquellos que son parte de las actividades del proceso y externos aquellos que aportan o se benefician de las actividades de la organización.
4	c	Una de las características de los Interesados es que al ser parte del desarrollo de las actividades del proceso, influyen ya sea directa o indirectamente en el proceso.
5	a	El sistema de nómina en cualquier ámbito opera sobre los empleados de la empresa, por lo tanto, la participación de este en el sistema es fundamental.
6	c	Entendiendo como equipo de proyecto, aquellos que están a cargo del desarrollo del sistema. El experto en usabilidad es un Interesado que aporta para el desarrollo del sistema, pero no es parte del equipo de desarrollo.
7	a	El software comercial requiere de indicadores a nivel de quienes serían los potenciales clientes, por ende es importante disponer de los requisitos de esos potenciales clientes.
8	c	Los requisitos de mercado necesitan conocer los potenciales clientes, por lo tanto, en el entorno es importante identificar esos segmentos y sobre todo que es lo que esperan. Por ejemplo, que esperan las farmacias con respecto a la facturación.
9	b	En el diagrama de contexto el sistema se lo representa como un círculo que va en el centro del diagrama.
10	b	El visionamiento se orienta exclusivamente a determinar las necesidades y por ende las características del producto.

Ir a la
autoevaluación

Autoevaluación 3		
Pregunta	Respuesta	Retroalimentación
1	c	Las actividades principales de la fase de obtención es la de recolectar, descubrir y extraer la suficiente información mediante estrategias desde fuentes específicas para definir apropiadamente los requisitos.
2	c	Para aplicar las estrategias de obtención es fundamental prepararse mediante la definición del alcance, contar con los recursos para hacerlo y sobre todo que es lo que queremos lograr en cada modelo.
3	b	Para que una entrevista sea productiva y considerando los pasos que recomienda esta técnica, el tiempo de duración está entre 45 a 60 minutos. Más de este tiempo se vuelve pesado y ya no es productiva.
4	a	A un taller asisten personas que conocen de un tema específico y sobre el cual se analizarán y llegarán a acuerdos que permitirán especificar requisitos puntuales.
5	a	Al empezar un proyecto de software inicialmente se requiere de necesidades de alto nivel, por lo tanto, las fuentes de información son interesados específicos, dónde una de las estrategias para obtener información son las entrevistas.
6	b	Una de las primeras actividades que debe ser parte de preparación de una entrevista es identificar a quien entrevistar, con base en el objetivo que se persigue o que requiere conocer.
7	b	En la observación, el enfoque pasivo revisa aquellos recursos que existen y a los cuales puede acceder, como son las notas que se generan de las actividades, los videos que permitan conocer el detalle de las actividades o cualquier otro recurso que aporte en el descubrimiento de información.
8	c	En toda estrategia de obtención, una de las primeras actividades es determinar el objetivo y cuáles serían esas fuentes para lograrlo, por lo tanto, en un taller es necesario determinar cuál es el propósito de desarrollar esta actividad, que es lo que se pretende lograr y luego elegir los participantes más idóneos que pueden participar.
9	a	En una entrevista estructurada, una vez identificado el propósito y a quien se va a entrevistar, se debe
10	b	El SRI al ser una entidad externa aporta con las disposiciones a las que debe ajustarse un producto software que incorpore la facturación. Sería una potencial fuente para identificar requisitos, reglas de negocio o restricciones.

**Ir a la
autoevaluación**

Autoevaluación 4		
Pregunta	Respuesta	Retroalimentación
1	a	El resultado del análisis de requisito es el desarrollo de un modelo de requisitos que puede ser representado mediante diagramas y especificaciones textuales, como es el caso del modelo de requisitos basados en casos de uso.
2	c	La mejor forma de representación de un modelo de análisis es a través de diagramas combinados con especificaciones textuales, de manera que tanto los interesados como el equipo de desarrollo lo puedan entender.
3	b	El diagrama de contexto es aquella representación gráfica (diagrama de flujo de datos), que permite representar al sistema como elemento central, las entidades externas y los flujos de información que indican la información de entrada o salida desde las entidades al sistema o desde el sistema respectivamente.
4	a	El que interactúa con el sistema a través de la interfaz en un modelo de casos de uso, se lo conoce como actor, por lo tanto, una tabla de actores permite identificar aquellos que serán los que utilicen el sistema y ejecuten las funcionalidades del mismo.
5	b	Para entender el negocio, se debe conocer las actividades que se desarrollan para cada uno de los procesos, por lo tanto, un mapa de procesos o un mapa de relaciones permite representar cada una de estas actividades y la forma en que se desarrollan.
6	a	El modelo de casos de uso considera el diagrama y la especificación de cada caso de uso. La especificación permite establecer al detalle la interacción entre el actor y el sistema.
7	b	Un proceso de negocio se lo puede representar mediante un mapa de procesos en el que se definen todas las actividades que intervienen en el proceso con sus respectivos responsables de cada actividad.
8	a	Las restricciones son las reglas de negocio que limitan ciertas acciones que se podrían dar, pero que son parte esencial del negocio, y que se den de respetar.
9	b	Los hechos son reglas de negocio que afirman cuestiones que se debe de realizar como parte de los procesos .
10	c	La generalización es una relación que amplía la funcionalidad de un caso de uso o refina su funcionalidad original mediante el agregado de nuevas operaciones, atributos y secuencias de acciones.

**Ir a la
autoevaluación**

Autoevaluación 5		
Pregunta	Respuesta	Retroalimentación
1	c	El documento de Especificación de requisitos de software, es la fuente principal para el diseño, implementación, pruebas, documentación del sistema, etc.
2	a	El Analista de negocio, Analista de requisitos o simplemente Analista es el responsable directo de aplicar las estrategias adecuadas de obtención y análisis para especificar los requisitos de software.
3	a	Como resultado del proceso de desarrollo de requisitos de la Ingeniería de Requisitos es el documento, de especificación de requisitos, debidamente organizados en funcionales y no funcionales.
4	c	El documento de visionamiento del producto, considera ciertas características del sistema que se intenta desarrollar, por lo tanto, este documento se convierte en fuente de datos para realizar la especificación.
5	b	El proceso de especificación consiste en inicialmente especificar los requisitos de negocio, luego se derivan los requisitos de negocio, para finalmente definir los requisitos de software.
6	a	Tanto para la validación como para la verificación se requiere de requisitos debidamente definidos que permitan desarrollar los escenarios de prueba, por lo tanto, el ERS, es fundamental convirtiéndose en determinado momento en línea base del proyecto.
7	a	Los documentos en los que se describen los requisitos son una recopilación de información que debe estar sustentada, por lo tanto, es preciso determinar dónde está la información, a esto se lo conoce como Identificar las fuentes u origen de la información.
8	a	Cada requisito funcional se asocia a cierto criterio de calidad, por lo tanto, cada definición de un requisito no funcional se debe clasificar con base en los criterios de calidad.
9	b	En el ERS se indica de forma general las limitaciones de diseño e implementación, luego en las siguientes fases se deberá ser más específico en estas limitaciones.
10	c	Se dice que un requisito es trazable, cuando se puede determinar de dónde proviene y bajo qué circunstancias se define.

Ir a la
autoevaluación

Autoevaluación 6		
Pregunta	Respuesta	Retroalimentación
1	b	El proceso de validación de requisitos requiere del desarrollo de varias actividades, y para ello es necesario seleccionar la técnica apropiada, elegir a los usuarios que podría aportar, revisar detenidamente los requisitos en los documentos que se haya elaborado.
2	c	Quienes poseen la información y conocen del desarrollo de las actividades son los interesados, por lo tanto, son los llamados a revisar detenidamente los documentos generados en la especificación, para garantizar que los requisitos están completos y cumplen con las características de calidad.
3	b	Con base en estudios realizados por los investigadores, han determinado que corregir un error luego de haber pasado ya por etapas importantes y básicamente se descubre al final del proyecto, resulta ser muy costoso, se promedió que el costo está en el orden de 10 a 100 veces, esta escala depende del momento en que se descubra.
4	a	Una de las estrategias para revisar el documento de requisitos es la revisión por pares, donde cada individuo revisa el ERS.
5	a	Los escenarios de prueba se elaboran con base en los modelos de requisitos establecidos, por ejemplo en el modelo de casos de uso, la validación se realiza con base en cada componente del modelo.
6	b	Las pruebas se diseñan para diferentes documentos que se generan, ya sea en la especificación, diseño, implementación y pruebas del software. Por lo tanto, para las pruebas de aceptación se debe crear las pruebas de validación.
7	a	La revisión de los requisitos los realiza los interesados que conocen del proceso, aquellos que desarrollan cada una de las actividades, y que están en capacidad de entender y discutir el documento de ERS.
8	a	Una de las estrategias de los modelos de validación, es utilizar y herramientas que permitan simular una situación real para validar los modelos desarrollados.
9	b	Otra de las características de los modelos de validación es la de demostrar que los distintos modelos desarrollados, ya sean diagramas o descripciones textuales, se relacionan entre sí.
10	c	Otra de las estrategias de validación son los prototipos, para ellos partiendo de los requisitos definidos, se desarrollan las características del prototipo, para luego proceder a evaluar.

[Ir a la autoevaluación](#)

Autoevaluación 7		
Pregunta	Respuesta	Retroalimentación
1	c	La gestión de requisitos se desarrolla una vez que se ha establecido un alcance con base en los requisitos, es decir, defina la línea base. La gestión consiste en el seguimiento, control de los cambios y trazabilidad de los requisitos.
2	c	Al desarrollar las actividades de gestión lo que se pretende es llevar un control a los requisitos cuando por algún motivo se tienen que actualizar, con ello minimizar la posibilidad de cometer errores en las siguientes etapas del proyecto.
3	a	Existe diferentes tipos de software, por lo tanto, la técnica apropiada para realizar la gestión dependerá de la naturaleza del proyecto. Por lo tanto, el tipo de proyecto de software es el que determina la técnica que se deba utilizar.
4	a	El Análisis de los requisitos permite discutir y emitir criterios sobre los requisitos que se han definido o que se desea ajustar.
5	c	La trazabilidad es una de las características que es parte de la gestión de requisitos, que se encarga de determinar el origen y relación de cada requisito.
6	b	Cuando se requiere incorporar una mayor cantidad de información a los requisitos es necesario establecer ciertos atributos que ayuden con el control de la información. Esto resulta más apropiado utilizando alguna herramienta de gestión de requisitos.
7	a	Cuando se desea realizar un cambio, es necesario identificar todos los elementos que se ven involucrados por dicho cambio. Por lo tanto, es necesario realizar un análisis de impacto del cambio que se va a implementar.
8	a	La gestión de requisitos se desarrolla en cualquier momento, un cambio o ajuste a los requisitos se puede dar en cualquier fase del ciclo de vida de desarrollo de sistema.
9	c	Generalmente, las herramientas de gestión ayudan a realizar diferentes tipos de pruebas a diferentes momentos del desarrollo del proyecto. Una de estas es la redundancia.
10	b	Al definir atributos se está exigiendo a los involucrados a incorporar información relevante que aporta al correcto desarrollo del proyecto.

[Ir a la autoevaluación](#)



5. Glosario

ERS: Especificación de Requisitos de *Software*.

BA: Analista de negocio.

IR: Ingeniería de Requisitos.

DFD: Diagrama de Flujo de Datos.

ER: Diagrama Entidad-Relación.

IS: Ingeniería de *Software*.

Interesado: o Stakeholder, persona o institución que tiene incidencia en el desarrollo del proyecto.



6. Referencias bibliográficas

- Abran, A., Moore, J. W., Bourque, P., Dupuis, R. & Tripp, L. (2004). *Software engineering body of knowledge*. IEEE Computer Society, Angela Burgess, 25.
- Ambler, S. W. (2005). *The elements of UML (TM) 2.0 style*. Cambridge University Press.
- Boehm, B. W., Abts, C., Brown, A. W., Chulani, S., Clark, B. K., Horowitz, E., Madachy, R., Reifer, D. J. & Steece, B. (2009). *Software Cost Estimation with COCOMO II* (1st ed.). Prentice Hall Press.
- Bourque, P. & Fairley, R. (2014). *Guide to the Software Engineering Body of Knowledge (SWEBOK Guide V3.0)*. IEEE Computer Society Staff. www.swebok.org
- Bron, J.-Y. (2020). *System Requirements Engineering: A SysML Supported Requirements Engineering Method*. John Wiley \& Sons.
- Bruegge, B. & Dutoit, A. H. (2009). Object-oriented software engineering. using uml, patterns, and java. *Learning*, 5(6), 7.
- Bruegge, B. & H Dutoit, A. (2012). *Object-Oriented Software Engineering Using UML, Patterns, and JavaTM Third Edition*. by Pearson Education, Inc.,.
- Chen, A. & Beatty, J. (2012). *Visual models for software requirements*. Pearson Education.
- Cohn, M. (2010). *Succeeding with agile: software development using Scrum*. Pearson Education.
- Fagan, M. (2002). Reviews and inspections. *Software Pioneers--Contributions to Software Engineering*, 562-573.
- Gause, D. C., Weinberg, G. M. & others. (1989). *Exploring requirements: quality before design* (Vol. 7). Dorset House New York.

- Gottesdiener, E. (2005). *The Software Requirements Memory Jogger* (Goal/QPC). Goal/QPC.
- Hatley, D., Hruschka, P. & Pirbhai, I. (2013). *Process for system architecture and requirements engineering*. Addison-Wesley.
- IEEE. (1990). IEEE Standard Glossary of Software Engineering Terminology. *IEEE Std 610.12-1990*, 1-84. <https://doi.org/10.1109/IEEESTD.1990.101064>
- IIBA. (2015). *A Guide to the Business Analysis Body of Knowledge (BABOK Guide)*. Version 3.0. International Institute of Business Analysis.
- Lawrence, B. (1997). «Requirements Happens...». American Programmer.
- Leffingwell, D. (2010). *Agile software requirements: lean requirements practices for teams, programs, and the enterprise*. Addison-Wesley Professional.
- Leffingwell, D. & Widrig, D. (2000). *Managing software requirements: a unified approach*. Addison-Wesley Professional.
- Li, Q. & Chen, Y.-L. (2009). Data Flow Diagram. En *Modeling and Analysis of Enterprise and Information Systems: From Requirements to Realization* (pp. 85-97). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-89556-5_4
- Morgan, T. (2002). *Business rules and information systems: aligning IT with business goals*. Addison-Wesley Professional.
- Robertson, S. & Robertson, J. (2013). *Mastering the requirements process - Getting requirements right*. (Terceras). Pearson Education.
- Sommerville, I. & Sawyer, P. (1997). *Requirements Engineering: A Good Practice Guide* (1st ed.). John Wiley & Sons, Inc.
- Unhelkar, B. (2018). *Software engineering with uml*. Auerbach Publications.
- Wiegers, K. & Beatty, J. (2006). *Software requirements*. Pearson Education.
- Wiegers, K. & Beatty, J. (2013). *Software Requirements*. Microsoft Press.
- Young, R. R. (2004). *The requirements engineering handbook*, Norwood, MA: Artech House.



7. Anexos

Anexo 1. Caso de estudio.

Sistema de envío de encomiendas

Parte 1

“Transporta Fácil”, es una empresa privada dedicada al transporte de encomiendas que opera en el territorio ecuatoriano, desde el año 2010, durante los últimos se ha convertido en una empresa referente en este tipo de servicios.

La empresa ofrece los servicios de envío:

- Puerta a puerta.
- Sucursal a sucursal.
- Sucursal a puerta.
- Puerta a sucursal.

La modalidad de entrega es:

- Estándar: tiempo de entrega máximo 48 horas.
- Premium: tiempo de entrega máximo 24 horas.

La empresa dispone de su propia flota de camiones para la entrega de las encomiendas que se encuentran distribuidos en cada una de las sucursales en todo el país. Solo para ciertos casos hace uso de los medios de transportes aéreo y terrestre, dependiendo de la modalidad de entrega y del lugar.

La sede está ubicada en la ciudad de Quito y posee sucursales en todas las provincias del Ecuador. Aproximadamente dispone de 400 empleados distribuidos en las diferentes oficinas en todo el país. En cada oficina, a excepción de la sede en Quito, existen los administrativos, los repartidores y el bodeguero. Los administrativos están conformados por el director, la

secretaria, el asistente y en algunos lugares el contador. La cantidad de repartidores depende de la provincia. La oficina más pequeña dispone de 1 repartidor, mientras que la más grande (Quito) dispone de 30 repartidores. De igual manera, en las oficinas donde existe un gran volumen de encomiendas, el cargo de bodeguero lo desempeñan de 1 o varias personas, pero para el caso de oficinas pequeñas el cargo de bodeguero lo desempeña el asistente.

La sede en Quito estructuralmente consta de tres departamentos: *marketing*, financiero y recursos humanos.

Marketing: está conformado por el director de marketing y 3 asistentes. Las actividades que realiza el departamento son:

- Estudio y análisis de mercado.
- Desarrollo de planes y promociones de acuerdo con los objetivos estratégicos de la empresa.
- Conquista de nuevos clientes.
- Difusión de las campañas en todo el país a través del director de la oficina de cada sucursal.

Financiero: está conformado por el director financiero, 10 asistentes (de los cuales 2 están en la ciudad de Guayaquil) y 2 responsables de TI. El departamento es responsable de:

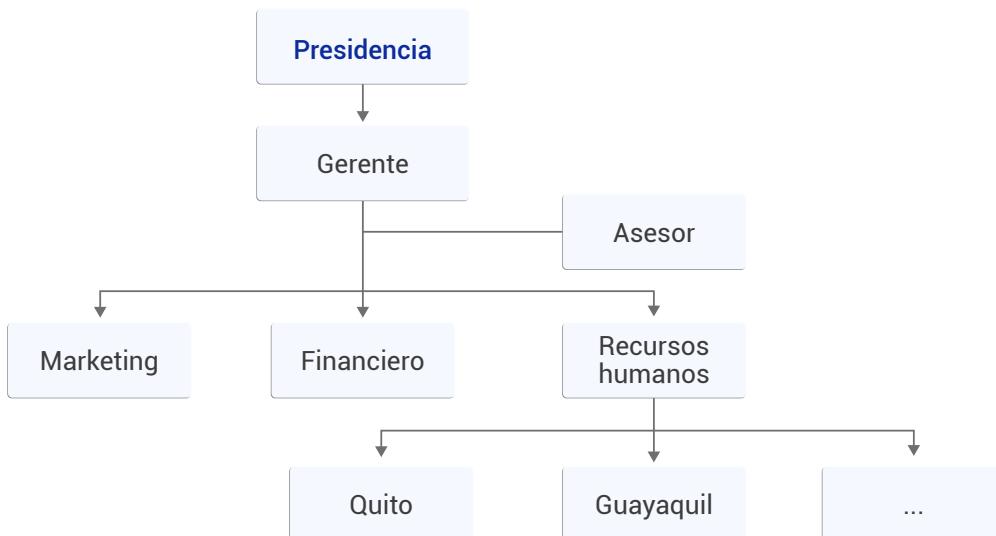
- Controlar y analizar los registros contables realizados por la empresa, de acuerdo con lo exigido por la normativa legal, contable y de procedimientos internos de la empresa.
- Controlar las labores de cobranza y, autorizar y ejecutar pago a proveedores y acreedores de la empresa.
- Determinación exacta, oportuna y confiable de los impuestos mensuales y anuales, dentro de los plazos legales y reglamentarios.
- Declaración oportuna de las obligaciones tributarias ante los organismos estatales.
- Coordinar con otros departamentos, la creación y aplicación de procedimientos y normas que faciliten la gestión financiera.
- Supervisar la custodia de bienes del activo fijo del personal a cargo.

- Presupuesto anual de la empresa.
- Los analistas de TI responden por el normal funcionamiento de las aplicaciones, redes de datos y recursos tecnológicos que posee la institución.

Recursos humanos: está conformado por el director y 5 asistentes (1 en la ciudad de Guayaquil y 1 en la ciudad de Cuenca). El departamento es responsable de:

- Planificación de la plantilla.
- Selección y formación del personal.
- Administración de personal.
- Evaluación del desempeño y control del personal.

En la figura se muestra el organigrama general de la empresa.



Cada departamento coordina con cada oficina para el desarrollo de sus actividades.

Las aplicaciones que actualmente utiliza la empresa son:

- Contabit. Sistema contable adquirido hace 10 años que actualmente ya no dispone de mantenimiento.
- SysFac. Sistema para facturar en cada oficina.

- TracSystem. Sistema para el seguimiento de las encomiendas, requiere de planificación de rutas y actualización de datos. La actualización se puede realizar de forma manual mediante un acceso web o a través de dispositivos.

Problemática

Debido a la gran cantidad de envíos que se deben coordinar, la empresa está teniendo varios problemas en la mayoría de sus operaciones y por ende en cada uno de los departamentos, debido a que cada oficina debe reportar de sus actividades a cada uno de los departamentos que se encuentran en la sede (Quito).

El departamento financiero es el más afectado, especialmente debido a:

- Documentos extraviados.
- Información desactualizada y en muchos casos con errores que provienen de las distintas oficinas.
- Transacciones incorrectas en cada una de las oficinas, lo que genera demasiado tiempo cuadrar.
- Deficiente coordinación con las oficinas por lo que la corrección de los errores requiere de un gran esfuerzo.
- Falta de información ocasionando una gran pérdida para la empresa, ya que se gastan recursos en actividades innecesarias.
- Las disposiciones que se dan desde el departamento no siempre se aplican en las oficinas.
- El sistema Contabit que ayuda con las actividades contables no aporta demasiado en el control y desarrollo de estas, ya que es un sistema exclusivo para el tema contable.
- En cada una de las oficinas se utiliza un sistema de facturación (SysFac) de forma local, lo que ocasiona que las transacciones no se actualicen de forma automática con la oficina central, sino que el consolidado de las transacciones se la realiza con base en reportes.

En el departamento de recursos humanos los principales problemas son:

- No se dispone de un seguimiento adecuado a las actividades de evaluación y desempeño del personal.
- Al no disponer de información de cada empleado no se aplican programas de capacitación acorde a las necesidades de la organización.
- No se dispone de elementos que permitan realizar un control efectivo del cumplimiento de la jornada laboral de los empleados.
- El sistema de comunicación con los empleados no es el adecuado. Las notificaciones se tardan mucho tiempo en llegar a sus destinatarios, por lo que se opta con llamadas telefónicas informales.
- No se realiza el control de la jornada de trabajo de los empleados. En muchas ocasiones, debido a la demanda y urgencia en la entrega de alguna encomienda, los repartidores tienen que laborar horas extras.
- La certificación para realizar los pagos mensuales se la realiza sin los controles necesarios.

En cada una de las oficinas, al gestionar las encomiendas se han identificado los siguientes problemas:

- No se dispone de una base de datos actualizada de clientes, debido a que cada oficina gestiona a su manera la información de sus clientes. Esto ocasiona que no se pueda tener un consolidado de todos los clientes de la empresa, con lo cual realizar los reporte toma demasiado tiempo.
- El registro de las encomiendas es de forma manual, por lo tanto, los datos se registran en papel, ocasionando que los registros no reflejan el estado real del envío.
- En ciertas ocasiones no se calcula de forma apropiada el costo de envío a pesar de disponer de los parámetros adecuados. La forma incorrecta de tomar los datos ocasiona que los envíos se retrasen y en algunos casos se extravíen, en cuyo caso la empresa debe asumir los costos, siendo esto una pérdida para la empresa.
- Los empleados invierten demasiado tiempo en la distribución de los paquetes de forma manual.

- La mala gestión de las encomiendas provoca sobrecarga de envíos, ocasionando que los repartidores tengan que laborar horas extras y al no tener ningún control no se reconoce el trabajo extra a los repartidores.

Con respecto a los clientes:

- Disponen de un sistema web (TracSystem), para consultar el estado de la encomienda, pero requiere de actualización manual por parte de los repartidores y en muchos de los casos no se realiza, por lo tanto, el cliente no cuenta con información de su encomienda y prefiere hacer llamadas telefónicas y en muchas ocasiones ir a la oficina.
- En la oficina es muy complicado obtener un detalle de las entregas de las encomiendas debido a que no se dispone de información oportuna y actualizada, a lo que en la mayoría de los casos la respuesta es "tiene que esperar".
- Malestar en los clientes por paquetes perdidos.
- Malestar en los clientes por el retraso en la entrega de las encomiendas.
- Malestar en los clientes debido a que en ciertas ocasiones les llega un paquete que no le corresponde.
- No dispone de un mecanismo de reclamos ante cualquier inconveniente.
- Para ciertas encomiendas el valor a pagar es demasiado alto.

Todos estos problemas generan inconvenientes con el desarrollo de las actividades que deben realizar los empleados de la empresa, generando insatisfacción en los clientes.

Parte 2

El gerente de la empresa y el responsable de TI han decidido contratarlo a usted, para que, en calidad de analista de negocio, desarrolle el modelo de requisitos para los procesos de:

- Registro de la encomienda.
- Clasificación, traslado y entrega de la encomienda.

- Generación de reportes.

A continuación se describe cada uno de estos procesos:

Proceso de registro de la encomienda

Actualmente, el registro de encomiendas consiste en que el cliente puede acercarse personalmente a la oficina o llamar para que la encomienda sea recogida desde un lugar específico y trasladado a la oficina. El cliente puede ser una persona natural o empresa. El cliente en la oficina es atendido por la secretaria (Asistente), quien realiza el registro de la encomienda. Para calcular el costo del servicio se realiza con base en: tipo, tamaño, peso, destino y tipo de envío. El cliente puede pagar el servicio en efectivo, tarjeta de débito, transferencia bancaria, tarjeta de crédito o cheque. Cuando la encomienda es recogida desde un determinado lugar, el repartidor elabora la guía y entrega al cliente, luego ingresa el paquete a bodega y notifica al asistente, entregando la guía para que sea completada.

El asistente emite la factura y elabora (o completa) la guía, para ello requiere de los datos del cliente, de quien envía y del destinatario, seguidamente el bodeguero registra en su planilla el ingreso del paquete, sella la guía como entregado y devuelve al cliente la guía, con lo cual termina el proceso.

Solamente para aquellos clientes que han sido aprobadas la forma de pago con cheque, el pago lo realizan al final de mes por todas las encomiendas realizadas durante el mes, esto especialmente para empresas.

Clasificación, traslado y envío del paquete

Una vez que el paquete se encuentra en las bodegas de la empresa, el bodeguero procede a definir las zonas de entrega para cada lugar (oficina), en las cuales se deberá definir sitios estratégicos por los que pasará el paquete y que se puedan registrar en el sistema. Las actividades a considerar son:

- Mantenimiento de las zonas, repartidores y camiones de entrega.
- Clasificar los paquetes que se encuentran en bodega de acuerdo con zonas de entrega.
- Asignar los paquetes a los repartidores.
- Actualizar el estado del paquete de acuerdo con los sitios estratégicos que se han definido en cada una de las zonas hasta llegar a su destino.
- Registrar la entrega del paquete.

- Notificar a los implicados sobre la entrega del paquete.

Reportes

Se requiere los reportes de:

- Listado de paquetes para ser entregados de forma diaria y por repartidor.
- Listado de paquetes que se han entregado hasta la fecha indicada.
- Listado de paquetes extraviados.
- Listado de paquetes que no se han podido entregar.

Anexo 2. Documento de visión y alcance

1. Requisitos de negocio

1.1. Antecedentes

Transporta Fácil es una empresa de transporte de encomiendas independiente que ofrece servicios de transporte o entrega de paquetes a diferentes partes del país. Actualmente, la empresa no cuenta con herramientas software que le permita administrar cada uno de los procesos de una manera eficiente, especialmente aquellos relacionados con sus clientes. La empresa cuenta con tres departamentos cuya sede se encuentra en la ciudad de Quito. Cada una de las sucursales distribuidas en todo el país deben reportar el desarrollo de las actividades y las transacciones producto de las encomiendas. Cuando un cliente requiere del servicio, debe trasladarse a la oficina con la encomienda para que sea enviada a un destino específico. La oficina al o contar con un sistema de gestión de encomiendas, el registro lo realiza utilizando una hoja electrónica y para la facturación utiliza el sistema SysFac, que permite emitir la factura, para que el cliente realice el pago por el servicio. La mayoría de las actividades de este proceso son manuales, lo que ocasiona que el cliente invierta un tiempo considerable. Igual ocurre con la secretaria o asistente, ya que al ser un proceso manual, al realizar el reporte para enviar a los distintos departamentos toma demasiado tiempo.

1.2. Oportunidad de negocio

Los directores de oficina han solicitado la implementación de un sistema integrado que le permita:

- Registrar las encomiendas cuando un cliente así lo requiera. Este registro se lo puede realizar en la oficina por parte de la secretaria o asistente, o lo pueda realizar el cliente en línea, en cuyo caso se notifique de manera automática al bodeguero para que la encomienda sea recogida desde un lugar específico.
- Realizar la facturación electrónica por el servicio y a su vez se notifique mediante correo electrónico al cliente.
- Actualizar automáticamente al sistema de seguimiento de encomiendas TracSystem.

- Obtener reportes consolidados y al instante de las transacciones realizadas, así como del estado de las encomiendas.
- Realizar la distribución de las encomiendas de acuerdo con las rutas de distribución.

Este sistema ahorrará tiempo tanto a los clientes como a los empleados y la información se consolidaría de tal manera que se obviaría las actividades manuales.

1.3. Objetivos de negocio

Reducir el tiempo de consolidación de las transacciones que se genera en cada oficina en un 50% a partir del primer lanzamiento.

Reducir el tiempo de registro de la encomienda en un 50% a partir de la implementación del sistema.

Generar reportes con información actualizada de las encomiendas que se han registrado.

Disponer de una base de datos consolidada de clientes de todas las oficinas a nivel nacional.

1.4. Métricas de satisfacción

- El 75% de los empleados que usaron la cafetería al menos 3 veces por semana durante el tercer trimestre de 2013 usan el SPC al menos una vez a la semana dentro de los 6 meses posteriores al lanzamiento inicial.
- La calificación promedio en la encuesta trimestral de satisfacción de la cafetería aumenta en 0.5 en una escala de 1 a 6 desde la calificación del tercer trimestre de 2013 dentro de los 3 meses posteriores a la publicación inicial y en 1.0 dentro de los 12 meses.

1.5. Declaración de la visión

Para los empleados de la empresa **quienes** necesitan registrar las encomiendas para ser enviadas a diferentes destinos, **el** Sistema de Envío de Encomiendas (SEE), **es una** aplicación web y móvil **que** permitirá registrar los datos de los clientes que pueden ser personas naturales o empresas, registrar los datos de envío de la encomienda, registrar el pago

y emitir los comprobantes de la transacción **a diferencia de** los procesos actuales que son manuales, los empleados que usan el **Sistema de Envío de Encomiendas** podrán registrar los envíos, lo que les ahorrará tiempo y aumentará la agilidad en las entregas.

1.6. Riesgos de negocio

- Muy pocos empleados pueden usar el sistema, lo que reduce el retorno de inversión del desarrollo del sistema y los cambios en los procedimientos operativos en cada oficina. (Probabilidad = 0.3; Impacto = 9).

1.7. Supuestos y dependencias de negocio

- Los sistemas con interfaces de usuario apropiadas estarán disponibles para que los empleados de las oficinas gestionen adecuadamente cada envío. (Supuesto)
- Los repartidores y los medios de transporte estarán disponibles para la entrega de todas las encomiendas dentro de los tiempos de entrega requerida. (Supuesto).

2. Alcance y limitaciones

2.1. Características principales

1. Llevar un registro único de clientes que hacen uso del servicio de encomiendas en todas las oficinas de la empresa.
2. Calcular el valor del envío de la encomienda con base en los criterios de tipo de envío, tamaño, peso y destino, para generar la factura de pago.
3. Llevar un control de los tipos y formas de pago que ofrece la empresa por el envío de la encomienda.
4. Gestionar la entrega de las encomiendas en base la gestión de rutas de envío.
5. Actualizar el estado de las encomiendas para que el cliente pueda saber el momento en que se realizará la entrega.

- Las actualizaciones las deberá realizar los repartidores a través del dispositivo móvil.

2.2. Alcance de las versiones iniciales y posteriores

Todas las características van para la versión 1.

2.3. Limitaciones y exclusiones

- Cada oficina deberá utilizar solamente el SEE, por parte de la secretaria o asistente. No se podrá utilizar otros sistemas locales que se han estado utilizando en algunas oficinas.
- Solamente se enviarán a los departamentos los reportes generados a través del SEE.

3. Contexto del negocio

3.1. Perfil de los Interesados

Interesado	Beneficios	Actitudes	Intereses	Criterios de éxito
Secretaria	Mejorar la productividad de los empleados.	Fuerte compromiso a para el desarrollo de la aplicación.	El ahorro de costos y tiempo de los empleados debe superar los costos de desarrollo y uso.	Ninguno identificado.
Bodeguero	Uso más eficiente del tiempo del personal a lo largo del día; mayor satisfacción del cliente.	Preocupación por las relaciones sindicales y la posible reducción de personal; de lo contrario receptivo.	Preservación del trabajo	Necesidad de capacitación para el personal en el uso de Internet; personal de reparto y vehículos necesarios.
Patrocinador	Mejorar la selección de comida; ahorro de tiempo; conveniencia.	Gran entusiasmo, pero es posible que no lo use tanto como se esperaba debido al valor social de comer almuerzos en la cafetería y los restaurantes.	Sencillez de uso; confiabilidad de la entrega; disponibilidad de opciones de alimentos.	Se necesita acceso a la intranet corporativa, acceso a Internet o un dispositivo móvil.

3.2. Prioridades del proyecto

Dimensión	Restricción	Conducción (proyecto)	Grado de libertad
Característica	Todas las funciones programadas para la versión 1.0 deben estar completamente operativas.		
Calidad	El 95% de las pruebas de aceptación de usuarios deben pasar; todas las pruebas de seguridad deben pasar.		
Programación			Se prevé que la versión 1 esté disponible a finales del primer trimestre del próximo año.
Costo			Presupuesto excedido hasta en un 15% aceptable sin revisión del patrocinador.
Personal		El tamaño del equipo es gerente de proyecto a medio tiempo, BA a medio tiempo, 3 desarrolladores y 1 probador; desarrollador adicional y probador de medio tiempo disponible si es necesario.	

3.3. Consideraciones de implementación

El software del servidor web deberá actualizarse a la última versión. Las aplicaciones deberán desarrollarse para teléfonos inteligentes, tabletas iOS y Android. Cualquier cambio de infraestructura debe estar implementado en el momento de la segunda versión. Se deben desarrollar videos de no más de cinco minutos de duración para capacitar a los usuarios en las versiones del sistema basadas en web y en aplicaciones.

Anexo 3. Modelos de análisis

Modelos de análisis que se utilizan para el análisis de información. Se desarrollan con base en la información obtenida a partir de la aplicación de las estrategias de obtención. A continuación se listan cada uno de ellos.

Diagrama de flujo de datos

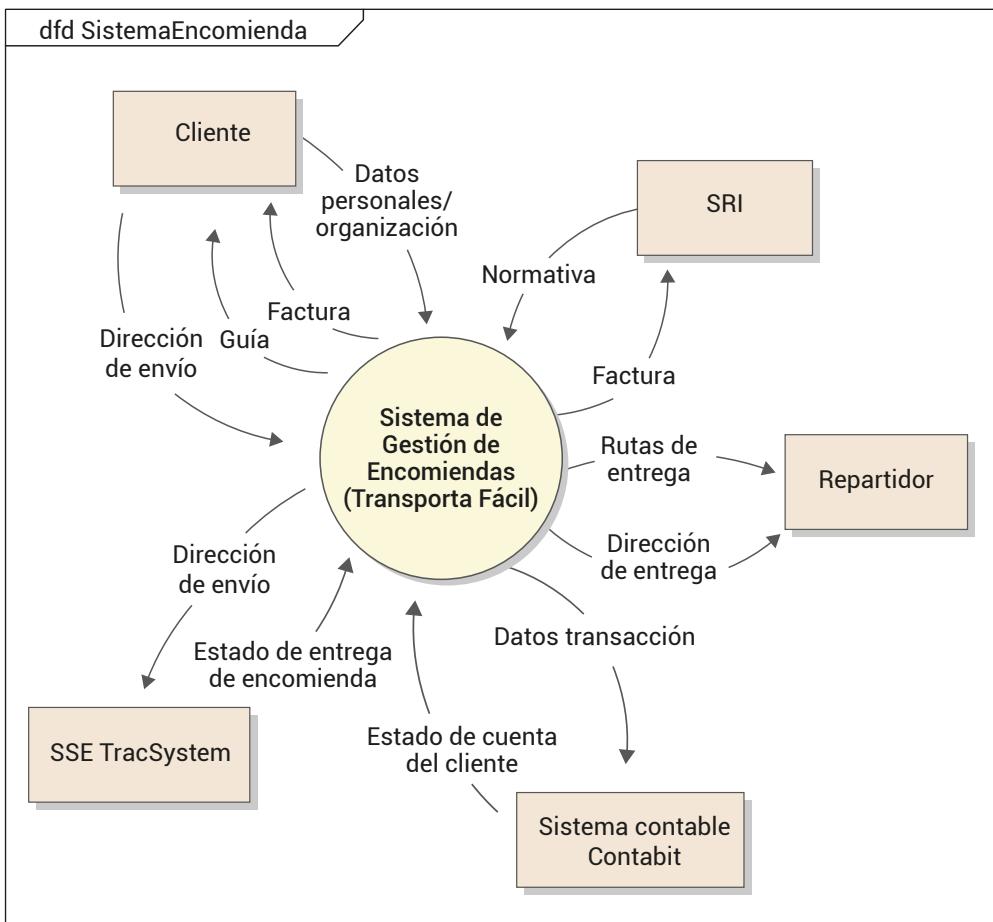
El Diagrama de Flujo de Datos (DFD) es un método estructurado de análisis y diseño. Es una herramienta visual para representar modelos lógicos y expresa la transformación de datos en un sistema. El DFD incluye un mecanismo para modelar el flujo de datos. Admite descomposición para ilustrar detalles de los flujos de datos y funciones. DFD no puede presentar información sobre la secuencia de operaciones (Li & Chen, 2009). Esto funciona bien para los sistemas de procesamiento de transacciones y otras aplicaciones de uso intensivo de funciones. A través de la adición de elementos de control, la técnica DFD se ha extendido para permitir el modelado de sistemas en tiempo real (Hatley et al., 2013).

Los DFD brindan una vista general de cómo se mueven los datos a través de un sistema, que otros modelos no muestran bien. Varias personas y sistemas ejecutan procesos que usan, manipulan y producen datos, por lo que cualquier caso de uso único o diagrama de carriles no puede mostrarle el ciclo de vida completo de un dato. Además, un proceso puede reunir y transformar varios datos (por ejemplo, el contenido del carrito de compras, la información de envío y la información de facturación se transforman en un objeto de pedido). Nuevamente, esto es difícil de mostrar en otros modelos. Sin embargo, los DFD no son suficientes como única técnica de modelado. Los detalles sobre cómo se transforman los datos se muestran mejor mediante los pasos de un proceso mediante casos de uso o diagramas de carriles.

Los DFD se pueden utilizar como una técnica para identificar los requisitos de datos faltantes. Los datos que fluyen entre procesos, almacenes de datos y entidades externas también se deben modelar en ERD y describir en el diccionario de datos. Además, un DFD brinda contexto a los requisitos funcionales con respecto a cómo el usuario realiza tareas específicas (Chen & Beatty, 2012).

Figura 37

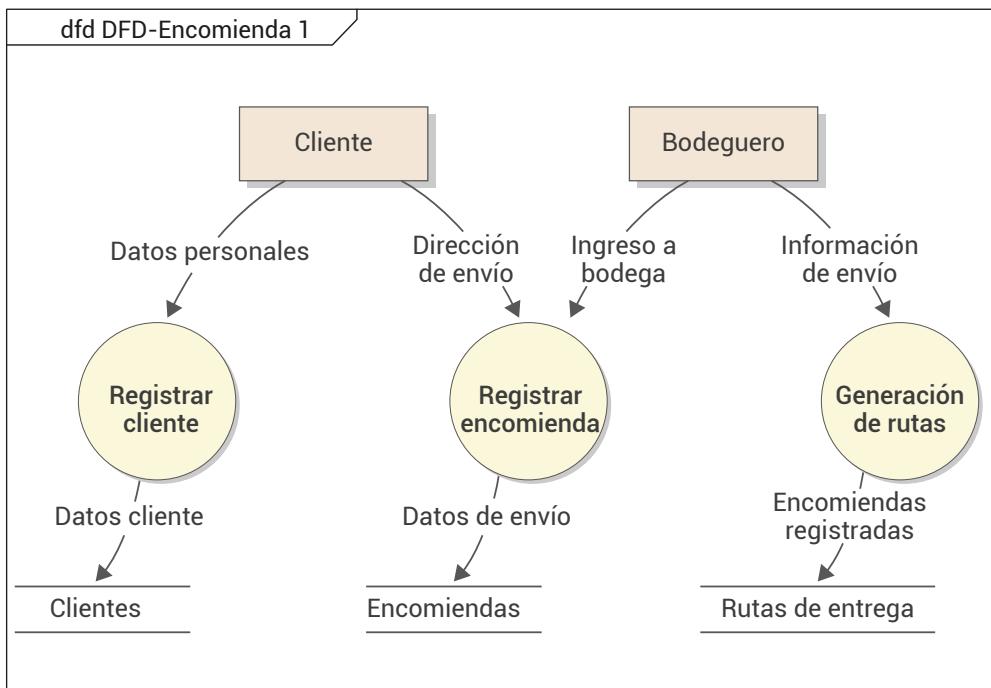
Diagrama de flujo de datos nivel 0 - Diagrama de contexto



El diagrama de contexto de la figura 31 representa el nivel más alto de abstracción del DFD. El diagrama de contexto representa todo el sistema como un único proceso de caja negra, representado como un círculo (una burbuja). También muestra las entidades externas, o terminadores, que se conectan al sistema y los flujos de datos o materiales entre el sistema y las entidades externas. Los flujos en un diagrama de contexto a menudo representan estructuras de datos complejas, que se definen en el diccionario de datos.

Figura 38

Diagrama de flujo de datos - Nivel 1



En la figura 32, se muestra el nivel 1, dónde el único proceso que se representa en el nivel 0 “Sistema de Gestión de encomiendas Transporta Fácil” (figura 21), en este diagrama se divide en tres procesos: registrar cliente, registrar enmienda y generación de rutas. Cada uno de estos procesos, a su vez podrían ser descompuestos en unidades más específicas, como por ejemplo “Registrar cliente”, podría tener los subprocesos: Registrar datos personales, buscar datos del cliente, modificar datos del cliente, etc., todo esto como parte de un nuevo nivel.

Cada proceso que aparece en el círculo en el diagrama puede ampliarse aún más en un DFD para revelar más detalles sobre su funcionamiento. El BA continúa este proceso progresivo hasta que los diagramas de nivel más bajo contienen solo operaciones primitivas del proceso que pueden ser claramente representadas en el texto narrativo, pseudocódigo, un mapa de proceso o un diagrama de actividad.

Los requisitos funcionales definirán precisamente lo que sucede dentro de cada proceso primitivo. Cada nivel del DFD debe ser equilibrado y consistente con el nivel por encima de él para que todos los flujos de

entrada y salida en el diagrama secundario coincidan con los flujos en su matriz. Las estructuras de datos complejas en los diagramas de alto nivel pueden dividirse en sus elementos constitutivos, tal como se definen en el diccionario de datos, en los DFD de nivel inferior.

Mapa de procesos

Los diagramas de carriles, conocidos también como: diagramas Swimlane, mapa de procesos, multifuncional (cross-functional) o modelo de línea de visibilidad (LOVEM), proporcionan una forma de representar los pasos involucrados en un proceso de negocio o las operaciones de un sistema de software propuesto. Son una variación de los diagramas de flujo, subdivididos en subcomponentes visuales llamados carriles. Los carriles pueden representar diferentes sistemas o actores que ejecutan los pasos del proceso. Los diagramas de carriles se usan más comúnmente para mostrar procesos de negocio, flujos de trabajo o interacciones entre el sistema y el usuario. Son similares a los diagramas de actividad UML.

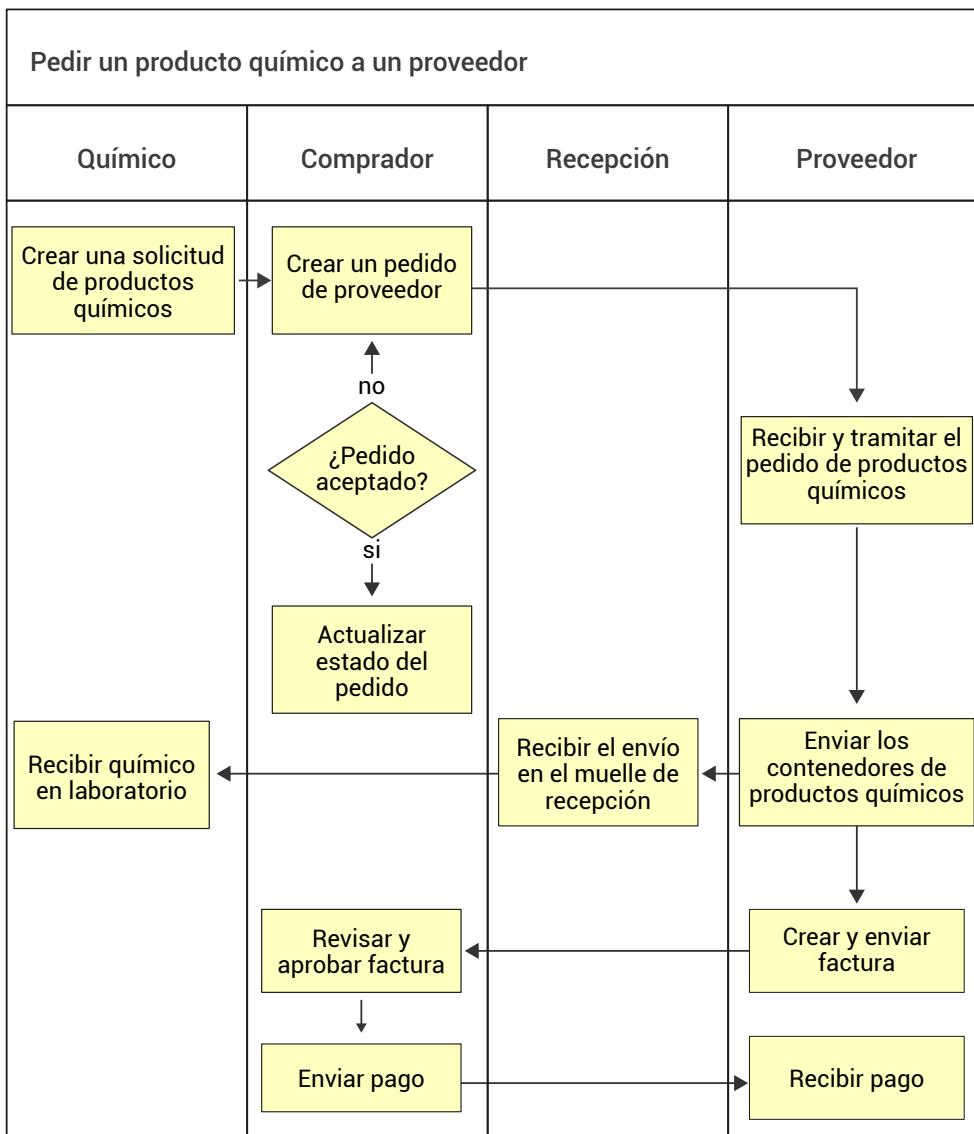
Los diagramas de carriles ayudan a unir los requisitos funcionales que permiten a los usuarios realizar tareas específicas. También se pueden utilizar para realizar un análisis detallado para identificar los requisitos que respaldan cada paso del proceso (Chen & Beatty, 2012).

El diagrama de carriles es uno de los modelos más fáciles de entender por los interesados porque la notación es simple y de uso común. Redactar procesos de negocio en diagramas de carriles puede ser un buen punto de partida para conversaciones de obtención. Los diagramas de carriles pueden contener múltiples formas, pero los elementos más utilizados son:

- Pasos del proceso, mostrados como rectángulos.
- Transiciones entre los pasos del proceso, que se muestran como flechas que conectan pares de rectángulos.
- Decisiones, mostradas como diamantes con múltiples ramas que salen de cada diamante. Las opciones de decisión se muestran como etiquetas de texto en cada flecha dejando un rombo.
- Carriles para subdividir el proceso, que se muestran como líneas horizontales o verticales en la página. Los carriles suelen ser roles, departamentos o sistemas. Muestran quién o qué está ejecutando los pasos en un carril determinado.

Figura 39

Mapa de procesos - Registrar encomienda



Tal como se observa en la figura 33, el diagrama representa el desarrollo de un conjunto de actividades de una forma secuencial, cada carril determina el responsable, además de quien es el que inicia y quien finaliza el proceso. Se observa que en el proceso intervienen tres responsables: Cliente, secretaria y el bodeguero. El proceso lo inicia el Cliente, ya que es quien tiene la necesidad de usar el servicio de envío de la encomienda y debe realizar un conjunto de actividades. Esta representación ayuda mucho a

aclarar las actividades que son parte de un proceso y que fácilmente pueden ser validadas por el Interesado. A partir de esta representación se pueden identificar elementos que permitan crear otros modelos, como por ejemplo determinar escenarios para desarrollar el modelo de requisitos basados en casos de uso.

Diagrama de transición de estado y tabla de estado

Los sistemas de software implican una combinación de comportamiento funcional, manipulación de datos y cambios de estado. Los sistemas en tiempo real y las aplicaciones de control de procesos pueden existir en uno de un número limitado de estados en un momento dado. Un cambio de estado puede tener lugar solo cuando se cumplen criterios bien definidos, como recibir un estímulo de entrada específico bajo ciertas condiciones. Un ejemplo es una intersección de carretera que incorpora sensores de vehículos, carriles de giro protegidos y botones y señales de cruce de peatones. Muchos sistemas de información se ocupan de objetos comerciales (órdenes de venta, facturas, artículos de inventario y similares) con ciclos de vida que involucran una serie de posibles estados o estados.

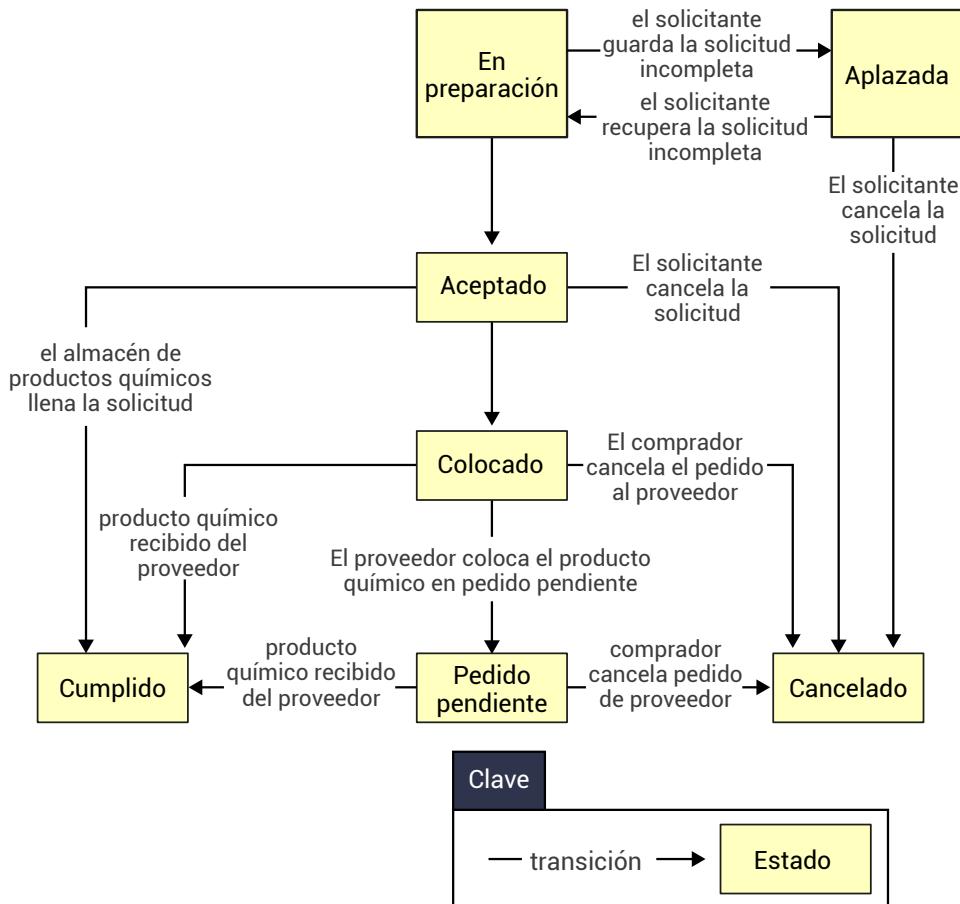
Los diagramas de transición de estado y las tablas de estado son dos modelos de estado que proporcionan una representación concisa, completa e inequívoca de los estados de un objeto o sistema. El diagrama de transición de estado (DTE) muestra visualmente las posibles transiciones entre estados. Una técnica relacionada es el diagrama de máquina de estado incluido en el Lenguaje de modelado unificado (UML), que tiene un conjunto más rico de notaciones y que modela los estados por los que pasa un objeto durante su vida útil (Ambler, 2005). El DTE contiene tres tipos de elementos:

- Posibles estados del sistema, mostrados como rectángulos. Algunas notaciones usan círculos para representar el estado. Los círculos o los rectángulos funcionan bien; solo sea consistente en lo que elija usar.
- Cambios o transiciones de estado permitidos, que se muestran como flechas que conectan pares de rectángulos.
- Eventos o condiciones que provocan que se produzca cada transición, que se muestran como etiquetas de texto en cada flecha de transición. La etiqueta podría identificar tanto el evento como la respuesta del sistema correspondiente.

El DTE de un objeto que pasa por un ciclo de vida definido tendrá uno o más estados de terminación, que representan los valores de estado final que puede tener un objeto. Los estados de terminación tienen flechas de transición que entran, pero ninguna sale. Los clientes pueden aprender a leer un DTE con solo un poco de entrenamiento sobre la notación, estas no son más que cuadros y flechas.

Figura 40

Diagrama de transición de estado



Nota. Adaptado de *Diagrama de transición de estado parcial para una solicitud de producto químico en el sistema de seguimiento de productos químicos*. (p. 233), K. Wiegers, 2013. Microsoft Press.

Este DTE muestra que una solicitud individual puede tomar uno de los siguientes siete estados posibles:

- En preparación. El solicitante está creando una nueva solicitud, habiendo iniciado esa función desde alguna otra parte del sistema.
- Postpuesto. El solicitante guardó una solicitud parcial para completarla en el futuro sin enviar la solicitud al sistema ni cancelar la operación de solicitud.
- Aceptado. El solicitante envió una solicitud de producto químico completa y el sistema la aceptó para su procesamiento.
- Realizado. La solicitud debe ser satisfecha por un proveedor externo y un comprador ha realizado un pedido con el proveedor.
- Cumplido. La solicitud se ha satisfecho, ya sea mediante la entrega de un contenedor de productos químicos desde el almacén de productos químicos al solicitante o mediante la recepción de un producto químico de un proveedor.
- Pedido pendiente. El proveedor no tenía el producto químico disponible y notificó al comprador que tenía un pedido pendiente para entrega futura.
- Cancelado. El solicitante canceló una solicitud aceptada antes de que se cumpliera, o el comprador canceló un pedido de proveedor antes de que se cumpliera o mientras estaba en espera.

Una tabla de estado muestra todas las posibles transiciones entre estados en forma de matriz. Un BA puede usar tablas de estado para garantizar que todas las transiciones se identifiquen mediante el análisis de cada celda de la matriz. Todos los estados se escriben en la primera columna y se repiten en la primera fila de la tabla. Las celdas indican si la transición de un estado a la izquierda a un estado en la parte superior es válida e identifica el evento de transición para moverse entre estados.

Figura 41
Tabla de estados

	En preparación	Pospuesto	Aceptado	Colocado	Pedido pendiente	Cumplido	Cancelado
En preparación	No	el usuario guarda el pedido incompleto	el sistema acepta una solicitud válida	No	No	No	No
Pospuesto	el usuario recupera la solicitud incompleta	No	No	No	No	No	No
Aceptado	No	No	No	el comprador hace el pedido con el proveedor	No	el almacén de productos químicos llena la solicitud	el solicitante cancela la solicitud
Colocado	No	No	No	No	proveedor coloca un producto químico en un pedido pendiente	químico recibido del proveedor	el comprador cancela el pedido del proveedor
Pedido pendiente	No	No	No	No	No	químico recibido del proveedor	el comprador cancela el pedido del proveedor
Cumplido	No	No	No	No	No	No	No
Cancelado	No	No	No	No	No	No	No

Nota. Adaptado de Tabla de estados para una solicitud de productos químicos en el Sistema de Seguimiento de Productos Químicos. (p. 235) K. Wiegers, (2013). Microsoft Press.

La figura 35 muestra una tabla de estado que coincide con el diagrama de transición de estado de la figura 34. Estas dos representaciones muestran exactamente la misma información, pero el formato de tabla ayuda a garantizar que no se pierdan transiciones, y el formato de diagrama ayuda a los Interesados a visualizar las posibles secuencias de transiciones. Es posible que no necesite crear ambos modelos. Sin embargo, si ya ha creado uno, el otro es fácil de crear, si desea analizar los cambios de estado desde dos perspectivas.

El diagrama de transición de estado y la tabla de estado proporcionan un punto de vista de alto nivel que abarca múltiples casos de uso o historias de usuario, cada uno de los cuales puede realizar una transición de un estado a otro. Los modelos de estado no muestran los detalles del procesamiento que realiza el sistema; solo muestran los posibles cambios de estado que resultan de ese procesamiento. Ambos modelos son útiles para garantizar que todos los estados y transiciones requeridos se hayan descrito correcta y completamente en los requisitos funcionales.

Mapas de diálogo

El mapa de diálogo representa un diseño de interfaz de usuario con un alto nivel de abstracción. Muestra los elementos de diálogo en el sistema y los enlaces de navegación entre ellos, pero no muestra los diseños de pantalla detallados. Una interfaz de usuario puede considerarse como una serie de cambios de estado. Solo un elemento de diálogo (como un menú, un espacio de trabajo, un cuadro de diálogo, una indicación de línea o una pantalla táctil) está disponible en un momento dado para la entrada del usuario. El usuario puede navegar a otros elementos de diálogo determinados en función de la acción que realice en la ubicación de entrada activa. El número de rutas de navegación posibles puede ser grande en un sistema complejo, pero el número es finito y las opciones generalmente se conocen. Un mapa de diálogo es realmente solo una interfaz de usuario modelada en forma de diagrama de transición de estado. Una técnica similar, llamada mapa de navegación, incluye un conjunto más rico de notaciones para representar diferentes tipos de elementos de interacción y transiciones de contexto. Un flujo de interfaz de usuario es similar a un mapa de diálogo, pero muestra las rutas de navegación entre las pantallas de la interfaz de usuario en un formato de diagrama de carriles (Ambler, 2005) (Chen & Beatty, 2012).

Un mapa de diálogo le permite explorar conceptos de interfaz de usuario hipotéticos en función de su comprensión de los requisitos. Los usuarios y desarrolladores pueden estudiar un mapa de diálogo para llegar a una visión común de cómo el usuario puede interactuar con el sistema para realizar una tarea. Los mapas de diálogo también son útiles para modelar la arquitectura visual de un sitio web. Los enlaces de navegación que crea en el sitio web aparecen como transiciones en el mapa de diálogo. Por supuesto, el usuario tiene opciones de navegación adicionales a través de los botones Atrás y Adelante del navegador, así como el campo de entrada de URL, pero el mapa de diálogo no los muestra. Los mapas de diálogo están relacionados con los guiones gráficos del sistema, que también incluyen una breve descripción del propósito de cada pantalla (Leffingwell & Widrig, 2000).

Los mapas de diálogo capturan la esencia de las interacciones entre el usuario y el sistema y el flujo de tareas sin atascar al equipo en diseños de pantalla detallados. Los usuarios pueden rastrear a través de un mapa de diálogo para encontrar navegaciones faltantes, incorrectas o innecesarias

y, por lo tanto, requisitos faltantes, incorrectos o innecesarios. El mapa de diálogo conceptual abstracto formulado durante el análisis de requisitos sirve como guía durante el diseño detallado de la interfaz de usuario.

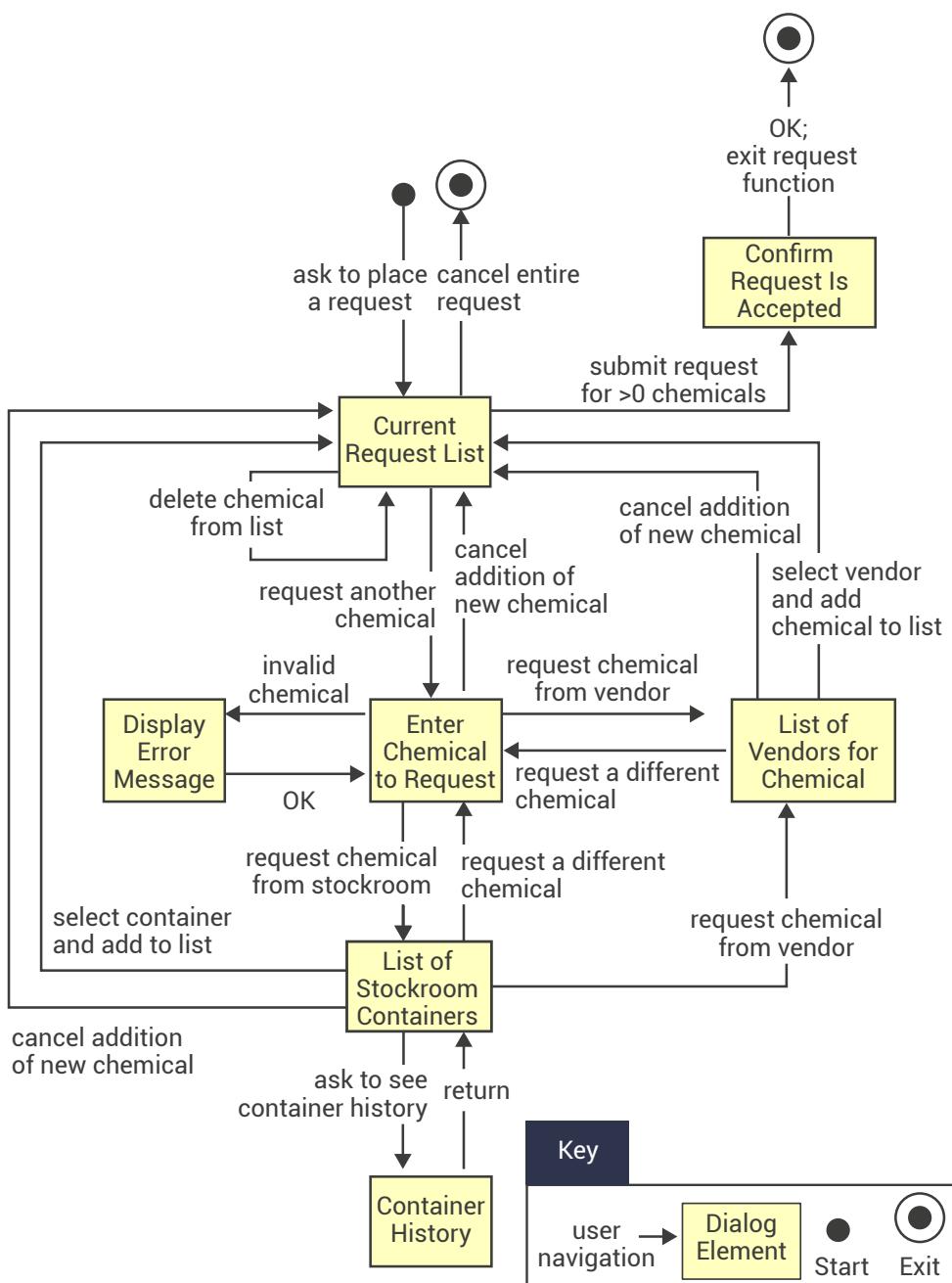
Al igual que en los diagramas de transición de estado ordinarios, el mapa de diálogo muestra cada elemento de diálogo como un estado (rectángulo) y cada opción de navegación permitida como una transición (flecha). La condición que activa la navegación de la interfaz de usuario se muestra como una etiqueta de texto en la flecha de transición. Hay varios tipos de condiciones de activación:

- Una acción del usuario, como presionar una tecla de función, hacer clic en un hipervínculo o hacer un gesto en una pantalla táctil.
- Un valor de datos, como un valor de entrada de usuario no válido, que activa la visualización de un mensaje de error.
- Una condición del sistema, como detectar que una impresora no tiene papel.
- Alguna combinación de estos, como escribir un número de opciones de menú y presionar la tecla Enter.

Los mapas de diálogo se parecen un poco a los diagramas de flujo, pero tienen un propósito diferente. Un diagrama de flujo muestra explícitamente los pasos de procesamiento y los puntos de decisión, pero no muestra la interfaz de usuario. Por el contrario, el mapa de diálogo no muestra el procesamiento que tiene lugar a lo largo de las líneas de transición que conectan un elemento de diálogo con otro, tal como se indica en la figura 26. Las decisiones de bifurcación (generalmente elecciones del usuario) están ocultas detrás de las pantallas de visualización que se muestran como rectángulos en el mapa de diálogo, y las condiciones que llevan a mostrar una pantalla u otra aparecen en las etiquetas de las transiciones. Un mapa de diálogo es una excelente forma de representar las interacciones entre un actor y el sistema que describe un caso de uso.

Figura 42

Ejemplo de un mapa de diálogo



Nota. Adaptado de un mapa de diálogo parcial para el caso de uso “Solicitar un producto químico” del sistema de seguimiento de productos químicos. (p. 237), K. Wiegers, 2013. Microsoft Press.

Tabla de decisión y árbol de decisión

Un sistema de software a menudo se rige por una lógica compleja, con varias combinaciones de condiciones que conducen a diferentes comportamientos del sistema. Los desarrolladores necesitan requisitos funcionales que describan lo que debe hacer el sistema en todas las combinaciones posibles de condiciones. Sin embargo, es fácil pasar por alto una condición, lo que da como resultado que falte un requisito. Estas brechas son difíciles de detectar al revisar una especificación textual.

Las tablas de decisión y los árboles de decisión son dos técnicas alternativas para representar lo que debe hacer el sistema cuando entran en juego decisiones y lógica compleja (Chen & Beatty, 2012). Una tabla de decisiones enumera los diversos valores de todos los factores que influyen en el comportamiento e indica la acción esperada del sistema en respuesta a cada combinación de factores. Los factores pueden mostrarse como declaraciones con posibles condiciones de verdadero y falso, como preguntas con posibles respuestas de sí y no, o como preguntas con más de dos valores posibles.

Figura 43

Ejemplo de una tabla de decisión

Número de requisito					
Condición	1	2	3	4	5
El usuario está autorizado	F	T	T	T	T
El químico está disponible	-	F	T	T	T
El producto químico es peligroso	-	-	F	T	T
El solicitante está capacitado	-	-	-	F	T
Acción					
Aceptar petición			X		X
Rechazar solicitud	X	X		X	

Nota. Adaptado de Ejemplo de tabla de decisiones para el sistema de seguimiento de productos químicos. (p. 239) K. Wiegers. 2013, Microsoft Press.

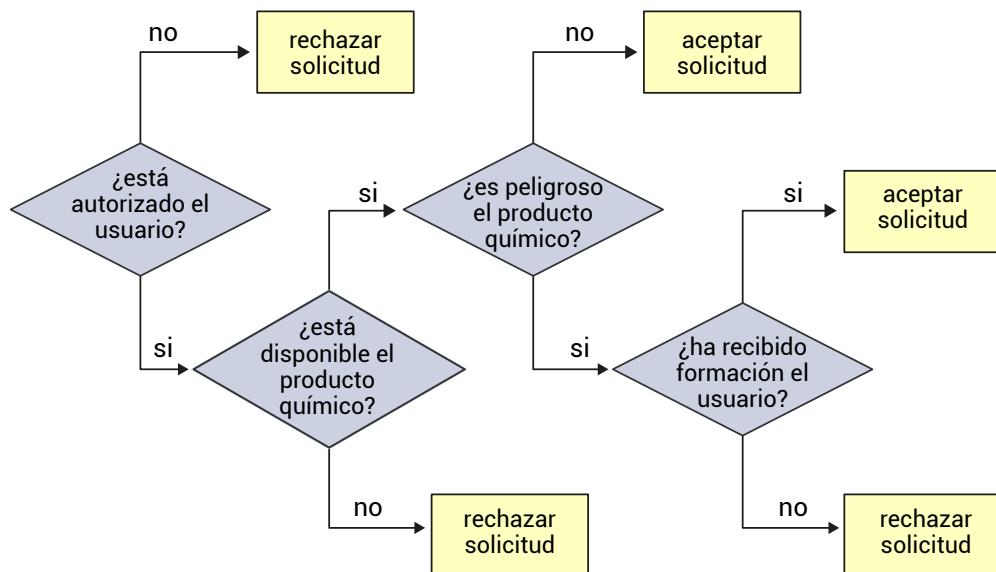
La figura 37 muestra una tabla de decisiones para la lógica que rige si el sistema de seguimiento de productos químicos debe aceptar o rechazar cada solicitud de un nuevo producto químico. Cuatro factores influyen en esta decisión:

- Si el usuario que está creando la solicitud está autorizado para solicitar productos químicos.
- Si el producto químico está disponible en el almacén de productos químicos o de un proveedor.
- Si el producto químico está en la lista de productos químicos peligrosos que requieren capacitación especial en el manejo seguro.
- Si el usuario que está creando la solicitud ha sido capacitado en el manejo de este tipo de sustancias químicas peligrosas.

Cada uno de estos cuatro factores tiene dos posibles condiciones, verdadero o falso. En principio, esto da lugar a 24, o 16, posibles combinaciones verdadero/falso, para un potencial de 16 requisitos funcionales distintos. Sin embargo, en la práctica, muchas de las combinaciones conducen a la misma respuesta del sistema. Si el usuario no está autorizado para solicitar productos químicos, el sistema no aceptará la solicitud, por lo que las demás condiciones son irrelevantes (se muestran como guiones en las celdas de la tabla de decisiones). La tabla muestra que solo surgen cinco requisitos funcionales distintos de las diversas combinaciones.

Figura 44

Ejemplo de un árbol de decisión



Nota. Adaptado de ejemplo de árbol de decisiones para el sistema de seguimiento de productos químicos. (p. 240), K. Wiegers, 2013. Microsoft Press.

La figura 38 muestra un árbol de decisiones que representa esta misma lógica. Los cinco recuadros indican los cinco posibles resultados de aceptar o rechazar la solicitud de producto químico. Tanto las tablas de decisiones como los árboles de decisiones son formas útiles de documentar requisitos (o reglas comerciales) para evitar pasar por alto cualquier combinación de condiciones. Incluso una tabla o árbol de decisión complejo es más fácil de leer que una gran cantidad de requisitos textuales repetitivos.

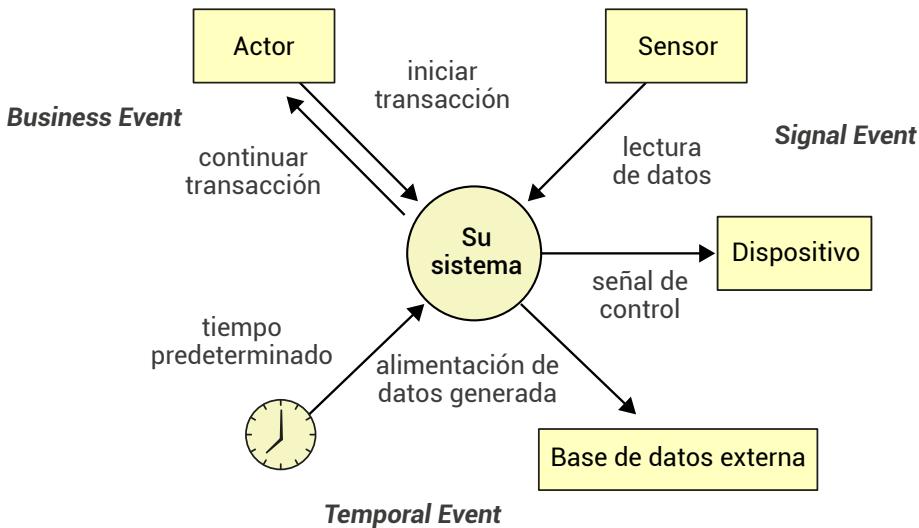
Tabla evento-respuesta

Los casos de uso y las historias de usuarios no siempre son útiles o suficientes para descubrir la funcionalidad que los desarrolladores deben implementar. Esto es particularmente cierto para los sistemas en tiempo real. Considere una intersección de carretera compleja con numerosos semáforos y señales para peatones. No hay muchos casos de uso para un sistema como este. Es posible que un conductor desee pasar el semáforo o girar a la izquierda o a la derecha. Un peatón quiere cruzar la calle. Tal vez un vehículo de emergencia quiera poder cambiar las señales de tráfico a verde en su dirección para poder acelerar su camino hacia las personas que necesitan ayuda. Las fuerzas del orden pueden tener cámaras en la

intersección para fotografiar las placas de los vehículos que infrinjan los semáforos en rojo. Esta información por sí sola no es suficiente para que los desarrolladores creen la funcionalidad correcta.

Otra forma de abordar los requisitos del usuario es identificar los eventos externos a los que debe responder el sistema. Un evento es algún cambio o actividad que tiene lugar en el entorno del usuario que estimula una respuesta del sistema de software. Una tabla de eventos-respuesta (también llamada tabla de eventos o lista de eventos) detalla todos esos eventos y el comportamiento que se espera que muestre el sistema en reacción a cada evento. Hay tres clases de eventos del sistema, como se muestra en la figura 39.

Figura 45
Ejemplo de eventos



Nota. Adaptado de Los sistemas responden a eventos comerciales, de señales y temporales. (p. 241), K. Wiegers. 2013, Microsoft Press.

- **Evento de negocio.** Un evento de negocio es una acción de un usuario humano que estimula un diálogo con el software, como cuando el usuario inicia un caso de uso. Las secuencias de evento-respuesta corresponden a los pasos en un caso de uso o diagrama de carriles.
- **Evento de señal.** Un evento de señal se registra cuando el sistema recibe una señal de control, una lectura de datos o una interrupción de un dispositivo de hardware externo u otro sistema de software, como cuando se cierra un interruptor, cambia el voltaje, otra aplicación

solicita un servicio o un usuario desliza su dedo en la pantalla de una tableta.

- **Evento temporal.** Un evento temporal se activa con el tiempo, como cuando el reloj de la computadora alcanza una hora específica (por ejemplo, para iniciar una operación de exportación automática de datos a medianoche) o cuando ha pasado una duración preestablecida desde un evento anterior (como en un sistema que registra la temperatura). Leído por un sensor cada 10 segundos.

El análisis de eventos funciona especialmente bien para especificar sistemas de control en tiempo real. Para identificar eventos, considere todos los estados asociados con el objeto de que está analizando e identifique cualquier evento que pueda hacer que el objeto pase a esos estados. Revise los diagramas de contexto para cualquier entidad externa que pueda iniciar una acción (desencadenar un evento) o requerir una respuesta automática (necesita que se active un evento temporal). La tabla 37 contiene un ejemplo de tabla de eventos y respuestas que describe parcialmente el comportamiento de los limpiaparabrisas de un automóvil. Aparte del evento 6, que es un evento temporal, todos estos son eventos de señal. Tenga en cuenta que la respuesta esperada depende no solo del evento, sino también del estado del sistema en el momento en que ocurre el evento. Por ejemplo, los eventos 4 y 5 en la tabla 8 dan como resultado comportamientos ligeramente diferentes según si los limpiaparabrisas estaban encendidos en el momento en que el usuario configuró el control del limpiaparabrisas en la configuración intermitente. Una respuesta podría simplemente alterar alguna información interna del sistema o podría resultar en un resultado visible externamente. Otra información que quizás desee agregar a una tabla de respuestas de eventos incluye:

Tabla 14

Ejemplo de una tabla evento-respuesta para un sistema de limpia parabrisas de un automóvil

ID	Evento	Estado del sistema	Respuesta del sistema
1	Ajuste el control del limpiaparabrisas a baja velocidad	Limpiador apagado, en alta velocidad, o en intermitente	Ajuste el motor del limpiaparabrisas a baja velocidad
2	Ajuste el control del limpiaparabrisas a alta velocidad	Limpiador apagado, en baja velocidad, o en intermitente	Ajuste el motor del limpiaparabrisas a alta velocidad

ID	Evento	Estado del sistema	Respuesta del sistema
3	Ajuste el control del limpiaparabrisas a apagado	Limpiado en alta velocidad, en baja velocidad, o en intermitente	<ol style="list-style-type: none"> Ciclo completo de limpieza actual Desactiva el motor del limpiaparabrisas
4	Ajuste el control del limpiaparabrisas a intermitentes	Limpiador apagado	<ol style="list-style-type: none"> Realice un ciclo de limpieza Lea el ajuste intermitente del tiempo de limpieza Inicializar el temporizador del limpiaparabrisas
5	Ajuste el control del limpiaparabrisas a intermitentes	Limpiador a baja velocidad o alta velocidad	<ol style="list-style-type: none"> Completar el ciclo actual de limpieza Leer el ajuste del intervalo de limpieza Inicializar el temporizador de limpieza
6	El intervalo de tiempo de limpieza que ha pasado desde que se completó el último ciclo	Limpiador intermitente	4. Ejecuta un ciclo de limpieza a baja velocidad
7	Cambiar el intervalo intermitente del limpiaparabrisas	Limpiador intermitente	<ol style="list-style-type: none"> Leer el ajuste del intervalo de limpieza Inicializar el temporizador de limpieza
8	Cambiar el intervalo intermitente del limpiaparabrisas	Limpiador apagado, a baja velocidad o alta velocidad	No responde
9	La señal de limpieza recibida	Limpiador apagado	Ejecutar un ciclo a baja velocidad

Nota. Adaptado de *Tabla parcial de eventos-respuesta para un sistema de limpiaparabrisas de automóviles*. K. Wiegers, 2013.

La frecuencia del evento (cuántas veces ocurre el evento en un período de tiempo dado, o un límite de cuántas veces puede ocurrir).

- Elementos de datos que se necesitan para procesar el evento.
- El estado del sistema después de que se ejecutan las respuestas al evento (Gottesdiener, 2005).

Enumerar los eventos que cruzan el límite del sistema es una técnica de alcance útil (Wiegers 2006). Una tabla de eventos y respuestas que defina todas las combinaciones posibles de eventos, estados y respuestas, incluidas las condiciones de excepción, puede servir como

parte de los requisitos funcionales para esa parte del sistema. Puede modelar la tabla de eventos y respuestas en una tabla de decisiones para garantizar que se analicen todas las combinaciones posibles de eventos y estados del sistema. Sin embargo, el BA debe proporcionar requisitos funcionales y no funcionales adicionales. ¿Cuántos ciclos por minuto realiza el limpiaparabrisas en las configuraciones de limpieza lenta y rápida? ¿La configuración intermitente es continuamente variable o tiene configuraciones discretas? ¿Cuáles son los tiempos de retraso mínimo y máximo entre barridos intermitentes? Si omite este tipo de información, el desarrollador tiene que rastrearla o tomar las decisiones él mismo. Recuerde, el objetivo es especificar los requisitos con la precisión suficiente para que un desarrollador sepa qué construir y un probador pueda determinar si se construyó correctamente.

Anexo 4. Descripción de la plantilla para ERS

1. Introducción

La introducción presenta una visión general para ayudar al lector a entender cómo se organiza el ERS y cómo usarlo.

1.1. Propósito

Identificar el producto o la aplicación cuyos requisitos se especifican en este documento, incluyendo el número de revisión o liberación. Si este ERS pertenece solo a una parte de un sistema complejo, identifique esa porción o subsistema. Describir los diferentes tipos de lectores a los que se destina el documento, tales como desarrolladores, gestores de proyectos, personal de marketing, usuarios, probadores y documentadores.

1.2. Convenciones de documentos

Describa todas las normas o convenciones tipográficas utilizadas, incluyendo el significado de estilos de texto específicos, resaltados o anotaciones. Si está etiquetando manualmente los requisitos, puede especificar el formato aquí para cualquier persona que necesite agregar uno más tarde.

1.3. Alcance del proyecto

Proporcione una breve descripción del software especificado y su propósito. Relacionar el software con los objetivos del usuario o de la empresa y con los objetivos y estrategias de negocio. Si se dispone de una visión separada y un alcance o documento similar, consulte el mismo en lugar de duplicar su contenido aquí. Un ERS que especifica una liberación incremental de un producto en evolución debe contener su propia declaración de alcance como un subconjunto de la visión estratégica a largo plazo del producto. Puede proporcionar un resumen de alto nivel de las principales funciones que contiene la versión o las funciones significativas que realiza.

1.4. Referencias

Haga una lista de los documentos u otros recursos a los que se refiere este ERS. Incluya hipervínculos a ellos si están en una ubicación persistente. Estos pueden incluir guías de estilo de interfaz de usuario, contratos, normas, especificaciones de requisitos del sistema, especificaciones

de interfaz o ERS para un producto relacionado. Proporcione suficiente información para que el lector pueda acceder a cada referencia, incluyendo su título, autor, número de versión, fecha, fuente, ubicación de almacenamiento o URL.

2. Descripción general

Esta sección presenta una visión general de alto nivel del producto y el entorno en el que se utilizará, los usuarios anticipados y las restricciones, suposiciones y dependencias conocidas.

2.1. Perspectiva del producto

Describa el contexto y el origen del producto. ¿Es el siguiente miembro de una línea de productos en crecimiento, la próxima versión de un sistema maduro, un reemplazo para una aplicación existente o un producto completamente nuevo? Si este ERS define un componente de un sistema más grande, indique cómo se relaciona este software con el sistema general e identifique las interfaces principales entre los dos. Considere la posibilidad de incluir modelos visuales, como un diagrama de contexto o un mapa del ecosistema, para mostrar la relación del producto con otros sistemas.

2.2. Clases y características de usuario

Identifique las diferentes clases de usuarios que usted anticipa que usará este producto, y describa sus características pertinentes. Algunos requisitos pueden referirse solo a ciertas clases de usuario. Identifique las clases de usuario favorecidas. Las clases de usuario representan un subconjunto de las partes interesadas descritas en el documento de visión y alcance. Las descripciones de clase de usuario son un recurso reutilizable. Si está disponible un catálogo maestro de clases de usuario, puede incorporar descripciones de clase de usuario simplemente señalándolas en el catálogo en lugar de duplicar información.

2.3. Entorno operativo

Describir el entorno en el que operará el software, incluida la plataforma de hardware; Sistemas operativos y versiones; Ubicaciones geográficas de usuarios, servidores y bases de datos; Y las organizaciones que alojan las bases de datos, servidores y sitios web relacionados. Enumere cualquier otro componente o aplicación de software con el que el sistema debe coexistir pacíficamente. Si se requiere un extenso trabajo de infraestructura

técnica en conjunto con el desarrollo del nuevo sistema, considere la posibilidad de crear una especificación de requisitos de infraestructura separada para detallar ese trabajo.

2.4. Restricciones de diseño e implementación

Hay ocasiones en las que se debe utilizar cierto lenguaje de programación, se debe utilizar una biblioteca de códigos en particular que ya ha invertido tiempo para desarrollarla, y así sucesivamente. Describa cualquier factor que restrinja las opciones disponibles para los desarrolladores y la justificación para cada restricción. Los requisitos que incorporan o se escriben en la forma de ideas de solución en lugar de las necesidades están imponiendo limitaciones de diseño, a menudo innecesariamente, así que ten cuidado con ellos.

2.5. Suposiciones y dependencias

Una suposición es una afirmación que se cree que es verdadera en ausencia de pruebas o de conocimiento negativo. Pueden surgir problemas si los supuestos son incorrectos, obsoletos, no se comparten o cambian, por lo que ciertos supuestos se traducirán en riesgos del proyecto. Un lector de ERS podría asumir que el producto se ajustará a una convención de interfaz de usuario particular, mientras que otro podría asumir algo diferente. Un desarrollador podría asumir que un determinado conjunto de funciones se escribirá por encargo para esta aplicación, mientras que el analista de negocios podría asumir que se reutilizarán de un proyecto anterior y el director de proyecto podría esperar obtener una biblioteca de funciones comerciales. Los supuestos a incluir aquí son los relacionados con la funcionalidad del sistema; Los supuestos empresariales aparecen en el documento de visión y alcance.

Identificar las dependencias que el proyecto o sistema que se está construyendo tiene sobre factores externos o componentes fuera de su control. Por ejemplo, si se debe instalar Microsoft .NET Framework 4.5 o una versión más reciente antes de que se pueda ejecutar el producto, se trata de una dependencia.

3. Características del sistema

La plantilla que se indica en la Figura 6.2 muestra los requisitos funcionales organizados por la característica del sistema, que es solo una forma posible

de organizarlos. Otras opciones organizativas incluyen la organización de los requisitos funcionales por área funcional, proceso de uso, caso de uso, modo de operación, clase de usuario, estímulo y respuesta. También son posibles combinaciones jerárquicas de estos elementos, como los casos de uso dentro de las clases de usuario. No hay una sola elección correcta; Seleccione un método de organización que facilite a los lectores la comprensión de las capacidades previstas del producto. Describiremos el esquema de características como ejemplo.

3.x Función del sistema X

Indique el nombre de la función en pocas palabras, como “3.1 Registrar trámite”.

3.x.1 Descripción

Proporcione una breve descripción de la característica e indique si es de alta, media o baja prioridad. Las prioridades suelen ser dinámicas, cambiando a lo largo del proyecto. Si está utilizando una herramienta de gestión de requisitos, defina un atributo de requisito para la prioridad

3.x.2 Requisitos funcionales

Describa los requisitos funcionales específicos asociados con esta característica. Estas son las capacidades de software que deben implementarse para que el usuario realice los servicios de la característica o para realizar un caso de uso. Describa cómo el producto debe responder a las condiciones de error previstas y a las entradas y acciones no válidas. Etiquetar de manera única cada requisito funcional, como se describió anteriormente en este capítulo. Si está utilizando una herramienta de administración de requisitos, puede crear varios atributos para cada requisito funcional, como la justificación, el origen y el estado.

4. Requerimientos de datos

Los sistemas de información proporcionan valor mediante la manipulación de datos. Utilice esta sección de la plantilla para describir varios aspectos de los datos que el sistema consumirá como entradas, procesará de alguna manera o creará como salidas. Algunos autores describen muchos patrones para documentar con precisión los requisitos de datos (también conocidos como información).

4.1. Modelo de datos lógicos

Un modelo de datos es una representación visual de los objetos de datos y colecciones que el sistema procesará y las relaciones entre ellos. Existen numerosas anotaciones para el modelado de datos, incluyendo diagramas entidad-relación y diagramas de clases UML. Puede incluir un modelo de datos para las operaciones de negocio que está siendo abordado por el sistema o una representación lógica de los datos que el sistema manipulará. Esto no es lo mismo que un modelo de datos de implementación que se realizará en forma de diseño de base de datos.

4.2. Diccionario de datos

El diccionario de datos define la composición de las estructuras de datos y el significado, el tipo de datos, la longitud, el formato y los valores permitidos para los elementos de datos que constituyen esas estructuras. Las herramientas de modelado de datos comerciales a menudo incluyen un componente de diccionario de datos. En muchos casos, es mejor guardar el diccionario de datos como un artefacto independiente, en lugar de incrustarlo en el medio de un ERS. Esto también aumenta su potencial de reutilización en otros proyectos.

4.3. Informes

Si su aplicación genera informes, identifíquelos aquí y describa sus características. Si un informe debe ajustarse a un diseño predefinido específico, puede especificarlo aquí como una restricción, tal vez con un ejemplo. De lo contrario, enfóquese en las descripciones lógicas del contenido del reporte, secuencia de clasificación, niveles de total y así sucesivamente, diferir el diseño detallado del informe en la etapa de diseño.

4.4. Adquisición, integridad, retención y eliminación de datos

Si es relevante, describa cómo se obtienen y mantienen los datos. Por ejemplo, al iniciar un inventario de datos, es posible que necesite hacer un volcado inicial de todos los datos de inventario al sistema receptor y luego tener que alimentar solo los cambios. Exponga cualquier requisito relativo a la necesidad de proteger la integridad de los datos del sistema. Identificar cualquier técnica específica que sea necesaria, como copias de seguridad, puntos de verificación, reflejo o verificación de exactitud de datos. Políticas de estado que el sistema debe aplicar para retener o deshacerse de datos,

incluidos datos temporales, metadatos, datos residuales (como registros eliminados), datos almacenados en caché, copias locales, archivos y copias de seguridad provisionales.

5. Requisitos de la interfaz externa

Esta sección proporciona información para garantizar que el sistema se comunique correctamente con los usuarios y con elementos externos de hardware o software. El logro de un acuerdo sobre interfaces de sistemas externos e internos ha sido identificado como una mejor práctica de la industria del software. Un sistema complejo con múltiples subcomponentes debería crear una especificación de interfaz o una especificación de arquitectura de sistema separada. La documentación de la interfaz podría incorporar material de otros documentos por referencia. Por ejemplo, podría apuntar a un manual de dispositivo de hardware que enumera los códigos de error que el dispositivo podría enviar al software.

5.1. Interfaces de usuario

Describir las características lógicas de cada interfaz de usuario que el sistema necesita. Algunas características específicas de las interfaces de usuario podrían aparecer en la sección 6.1 Usabilidad. Algunos puntos para tratar aquí son:

1. Referencias a estándares de interfaz de usuario o guías de estilo de línea de producto que se deben seguir.
2. Estándares para fuentes, iconos, etiquetas de botones, imágenes, esquemas de color, secuencias de tabulación, controles de uso común, gráficos de marca, avisos de copyright y privacidad y similares.
3. Restricciones de tamaño, disposición o resolución de pantalla.
4. Botones, funciones o vínculos de navegación estándar que aparecerán en todas las pantallas, como un botón de ayuda.
5. Teclas de atajo.
6. Convenciones de presentación de mensajes y frases.

7. Directrices de validación de datos (como restricciones de valores de entrada y cuándo validar los contenidos de campo).
8. Estándares de diseño para facilitar la localización de *software*.
9. Alojamiento para usuarios con impedimentos visuales, ciegos de color o con otras limitaciones.

5.2. Interfaces de software

Describir las conexiones entre este producto y otros componentes de *software* (identificados por nombre y versión), incluyendo otras aplicaciones, bases de datos, sistemas operativos, herramientas, bibliotecas, sitios web y componentes comerciales integrados. Indique el propósito, los formatos y el contenido de los mensajes, datos y valores de control intercambiados entre los componentes de *software*. Especifique las asignaciones de datos de entrada y salida entre los sistemas y cualquier traducción que se necesite hacer para que los datos pasen de un sistema a otro. Describir los servicios necesarios para o desde componentes de *software* externos y la naturaleza de las comunicaciones entre componentes. Identifique los datos que se intercambiarán o compartirán entre componentes de *software*. Especifique los requisitos no funcionales que afectan a la interfaz, como los niveles de servicio para los tiempos de respuesta y las frecuencias, o los controles y restricciones de seguridad. Parte de esta información podría especificarse como requisitos de datos en la sección 4 o como requisitos de interoperabilidad en la sección 6, Atributos de calidad.

5.3. Interfaces de hardware

Describir las características de cada interfaz entre los componentes de *software* y los componentes de *hardware*, si los hay, del sistema. Esta descripción puede incluir los tipos de dispositivos soportados, los datos y las interacciones de control entre el *software* y el *hardware* y los protocolos de comunicación que se utilizarán. Enumere las entradas y salidas, sus formatos, sus valores o rangos válidos y cualquier problema de sincronización que los desarrolladores tengan que tener en cuenta. Si esta información es extensa, considere la posibilidad de crear un documento de especificación de interfaz separado.

5.4. Interfaces de comunicaciones

Indique los requisitos para cualquier función de comunicación que el producto utilice, incluido correo electrónico, navegador web, protocolos de red y formularios electrónicos. Defina cualquier formato de mensaje pertinente. Especifique la seguridad de la comunicación y los problemas de cifrado, las tasas de transferencia de datos, la sincronización y los mecanismos de sincronización. Indique las restricciones en torno a estas interfaces, como si ciertos tipos de archivos adjuntos de correo electrónico son aceptables o no.

6. Atributos de calidad

Esta sección especifica los requisitos no funcionales distintos de las restricciones, que se registran en la sección 2.4, y los requisitos de la interfaz externa, que aparecen en la sección 5. Estos requisitos de calidad deben ser específicos, cuantitativos y verificables. Indique las prioridades relativas de varios atributos, como la facilidad de uso sobre la facilidad de aprendizaje, o la seguridad sobre el rendimiento. Una rica notación de especificaciones como Planguage aclara los niveles necesarios de cada calidad mucho mejor que las simples descripciones.

6.1. Usabilidad

Los requisitos de usabilidad se refieren a la facilidad de aprendizaje, la facilidad de uso, la evitación y recuperación de errores, la eficiencia de las interacciones y la accesibilidad. Los requisitos de usabilidad especificados aquí ayudarán al diseñador de interfaz de usuario a crear la experiencia óptima del usuario.

6.2. Rendimiento

Especifique los requisitos de rendimiento específicos para las distintas operaciones del sistema. Si diferentes requerimientos funcionales o características tienen diferentes requisitos de rendimiento, es apropiado especificar esos objetivos de rendimiento correctamente con los requisitos funcionales correspondientes, en lugar de recopilarlos en esta sección.

6.3. Seguridad

Especifique cualquier requisito relacionado con cuestiones de seguridad o privacidad que restrinjan el acceso o uso del producto. Estos pueden

referirse a la seguridad física, de datos o de software. Los requisitos de seguridad a menudo se originan en las reglas de negocio, así que identifique las políticas o regulaciones de seguridad o privacidad a las cuales el producto debe cumplir. Si están documentados en un repositorio de reglas de negocio, solo refiérase a ellos.

6.4. Seguridad externa

Especifique los requisitos que se refieren a posibles pérdidas, daños o daños que pudieran resultar del uso del producto. Definir las salvaguardias o acciones que deben tomarse, así como las acciones potencialmente peligrosas que deben evitarse. Identifique las certificaciones de seguridad, políticas o reglamentos a los cuales el producto debe cumplir.

6.x [Otros]

Cree una sección separada en el ERS para cada atributo adicional de calidad del producto para describir las características que serán importantes para los clientes o para los desarrolladores y mantenedores. Las posibilidades incluyen disponibilidad, eficiencia, instalación, integridad, interoperabilidad, modificabilidad, portabilidad, fiabilidad, reutilización, robustez, escalabilidad y verificabilidad.

7. Requisitos de internacionalización y localización

Los requisitos de internacionalización y localización aseguran que el producto sea adecuado para su uso en naciones, culturas y ubicaciones geográficas distintas de aquellas en las que se creó. Esos requisitos podrían abordar las diferencias de divisas; formato de fechas, números, direcciones y números de teléfono; Idioma, incluyendo las convenciones nacionales de ortografía dentro del mismo idioma (como el inglés americano versus inglés británico), los símbolos utilizados y los conjuntos de caracteres; Nombre y apellido; zonas horarias; Reglamentos y leyes internacionales; Culturales y políticos; Tamaños de papel utilizados; pesos y medidas; Voltajes eléctricos y formas de enchufe, y muchos otros. Los requisitos de internacionalización y localización podrían ser reutilizables en todos los proyectos.

8. [Otros requisitos]

Defina cualquier otro requisito que no esté cubierto en otras partes del ERS. Ejemplos de ello son los requisitos legales, reglamentarios o normativos; Requisitos para la instalación, configuración, puesta en marcha y cierre

del producto, y registros, monitoreo y auditoría. En lugar de simplemente combinar todos estos en “otro”, agregue cualquier nueva sección a la plantilla que sea pertinente.

A su proyecto. Omita esta sección si todos sus requisitos se acomodan en otras secciones. Los requisitos de transición que son necesarios para migrar de un sistema anterior a uno nuevo podrían incluirse aquí si involucran software escrito (como para programas de conversión de datos), o en el plan de gestión de proyecto si no lo hacen.

Apéndice A: glosario

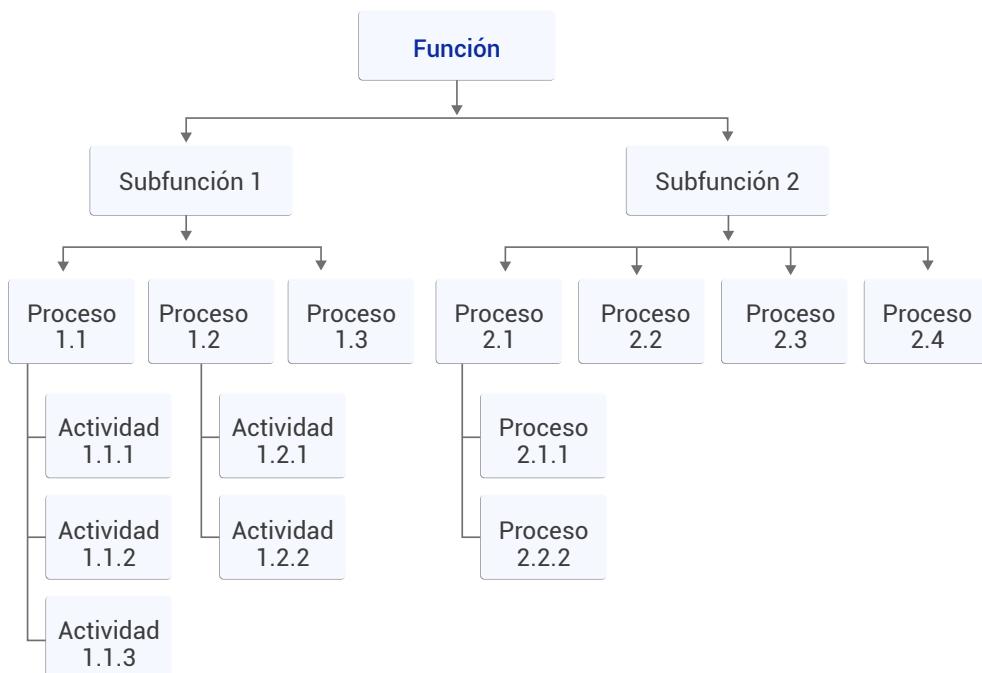
De todos los términos especializados que un lector necesita saber para entender el ERS, incluyendo acrónimos y abreviaturas. Escriba cada acrónimo y proporcione su definición. Considere la posibilidad de construir un glosario reutilizable a nivel de empresa que abarque múltiples proyectos e incorporando por referencia cualquier término que pertenezca a este proyecto. Cada ERS solo definiría los términos específicos de un proyecto individual que no aparecen en el glosario de nivel empresarial. Tenga en cuenta que las definiciones de datos pertenecen al diccionario de datos, no al glosario.

Apéndice B: modelos de análisis

Esta sección opcional incluye o apunta a modelos de análisis pertinentes, tales como diagramas de flujo de datos, árboles de características, diagramas de transición de estado o diagramas entidad-relación. A menudo, es más útil para el lector si incorpora ciertos modelos en las secciones pertinentes de la especificación en lugar de recogerlos al final.

Anexo 5. Descomposición funcional

La descomposición funcional permite descomponer procesos, áreas funcionales, o entregables en sus partes componentes y permitir que cada parte sea analizada independientemente. Implica el desglose de un problema grande en funciones o entregables más pequeños. El objetivo principal de la 'descomposición funcional' es asegurar que el problema sea dividido en subproblemas que sean tan independientes como sea posible, de modo que el trabajo pueda ser asignado a grupos diferentes. Esto proporciona la capacidad de graduar y administrar proyectos más grandes.



Elementos

La 'descomposición funcional' identifica las funciones de alto nivel de una organización o solución y luego separa esas funciones en piezas pequeñas, tales como subprocessos y actividades, características, y así sucesivamente.

Cuando se descompone una función de la organización, los modelos comienzan con una función de alto nivel, generalmente correspondiente a una unidad de la organización y continúa cayendo en cascada en subfunciones, representando los procesos que se llevan a cabo por esa unidad, y debajo de esos subprocessos y actividades individuales (los nombres para cada nivel son solamente convencionales y no implican que la

descomposición deba detenerse después que es alcanzado el cuarto nivel). Pueden ser representados por un diagrama jerárquico, un diagrama de árbol, o numerando cada subfunción. Cada función está totalmente compuesta por las subfunciones por debajo de ella. El proceso de descomposición funcional sigue hasta que una subfunción no puede ser dividida en dos o más funciones de nivel inferior.

Un proceso similar puede ser realizado para el trabajo implicado en un proyecto. Esta descomposición (conocida como una Estructura de Descomposición de Trabajo, EDT) descompone el alcance del proyecto en fases, paquetes de trabajo y entregables. La descomposición también puede ser realizada para describir un producto o proceso.

Ventajas

- Crea un modelo conceptual del trabajo que necesita ser llevado a cabo para entregar la solución nueva de negocio.
- Provee a todos los Interesados una visión congruente del alcance del esfuerzo. Ayuda a la estimación, ya que se pueden hacer las estimaciones para subconjuntos del todo y, por lo tanto, más fácilmente comprensibles.

Desventajas

- No hay modo alguno de estar seguro de que han sido capturados todos los componentes.

La descomposición de un problema sin entender totalmente la relación entre las piezas del problema puede crear una estructura inadecuada que impida el análisis.

Anexo 6. Especificación de requisitos de software

1. Introducción

1.1. Propósito

El presente documento de Especificación de Requisitos de Software (ERS) describe los requisitos funcionales y no funcionales para la versión de software 1.0 del Sistema de envío de encomiendas (SEE). Este documento está destinado a ser utilizado por los miembros del equipo de proyectos que implementarán y verificarán el correcto funcionamiento del sistema. Los requisitos definidos en este documento son considerados para la versión 1.0.

1.2. Convención de documento

En este documento de ERS no se utilizan convenciones tipográficas especiales.

1.3. Alcance del proyecto

El sistema SEE permitirá a los empleados llevar un control coordinado de las actividades relacionadas con la gestión de encomiendas:

- Llevar el registro de encomiendas que pertenezcan a los clientes de la empresa.
- Registrar cada encomienda junto a la generación de un comprobante de pago.
- Clasificar las encomiendas para poder trasladarlas mediante los trabajadores de la empresa y realizar la entrega de cada encomienda.
- Los clientes podrán realizar un seguimiento de sus encomiendas, además de que la información generada en el proceso de envío de la encomienda se envía en forma de comprobante al correo del cliente.
- Actualizar el estado del paquete durante el traslado y entrega de la encomienda mediante el móvil.
- Generar los reportes necesarios de los casos que puedan presentarse.

1.4. Referencias

- Documento de visión y alcance.
- Descripción del caso de estudio.

2. Descripción general

2.1. Perspectiva del producto

El sistema de envío de encomiendas se desarrollará en un ambiente web para actividades de oficina y a su vez estará disponible también para móvil, el objetivo del sistema es mejorar la manera en que se desarrollan las actividades de la empresa con el fin de mejorar los servicios brindados, para ello el sistema va a trabajar bajo un motor de *workflow* que ayude a llevar las diferentes etapas en el envío de cada encomienda de manera automática.

2.2. Clases y características de usuario

Nº	Clase de usuario	Descripción
1	Cliente	Es el consumidor principal de los servicios que ofrece la empresa, es quien solicitará el servicio de envío de encomiendas, el cual en primera instancia deberá ser registrado en el sistema por la secretaria (asistente).
2	Secretaria (Asistente)	Es la encargada de registrar a los clientes nuevos (persona natural y empresa) y paquetes en el sistema, además es quien obtiene los reportes de los paquetes y clientes.
3	Bodeguero	El bodeguero está encargado de clasificar las zonas de reparto a cada paquete para luego registrar en el sistema, además es quien se encarga de actualizar el estado del paquete.

2.3. Ambiente de operación

- OE-1: El SEE deberá operar correctamente con los siguientes navegadores web: Windows Internet Explorer versiones 7, 8 y 9; Firefox versiones 12 a 26; Google Chrome (todas las versiones); y Apple Safari versiones 4.0 a 8.0.
- OE-2: El SEE funcionará en un servidor que ejecute las versiones actuales aprobadas por la empresa de Red Hat Linux y Apache HTTP Server.

- OE-3: El SEE deberá permitir el acceso de los usuarios desde la intranet corporativa; desde una conexión a Internet VPN; y por teléfonos inteligentes y tabletas con Android, iOS y Windows.

2.4. Restricciones de diseño e Implementación

- CO-1: El diseño del sistema, el código y la documentación de mantenimiento deben cumplir con el Estándar de desarrollo de intranet de impacto en el proceso, versión 1.3 [2].
- CO-2: El sistema utilizará el motor de base de datos MYSQL estándar corporativo actual. CO-3: Todo el código HTML se ajustará al estándar HTML 5.0.

2.5. Suposiciones y dependencias

- SU-1: Las oficinas funcionan en horario de 08:00 a 19:00 interrumpidamente todos los días hábiles de la empresa.

3. Características del sistema

3.1. Manejo de clientes

3.1.1. Descripción

Permite registrar los datos del cliente, que pueden ser personas naturales o empresa. Al ser personas naturales se podrá gestionar la información personal, para poder realizar el envío de las encomiendas. Al ser una empresa los datos que identifiquen a la empresa.

3.1.2. Requisitos funcionales

Cliente. Gestionar.	Registrar cliente
.Nuevo:	El SEE permitirá registrar los datos esenciales de un nuevo cliente.
.No:	Si el cliente es una empresa deberá registrar los datos que corresponden a una empresa (razón social, RUC, representante legal)
.Buscar	El SEE permitirá buscar un cliente por diferentes criterios de búsqueda (identificación, apellidos/razón social)
.Modificar	El SEE permitirá modificar los datos personales o de empresa de un determinado cliente.
.Inhabilitar	El SEE permitirá inhabilitar a un cliente, y por ende no se podrá realizar ninguna operación con este cliente.

3.2. Gestión de encomiendas

(Desarrolle)

3.3. Gestión de pagos

(Desarrolle)

3.4. Reportes

(Desarrolle)

4. Requisitos de datos

No se define

5. Requisitos de Interfaz externa

5.1. Interfaz de usuario

- UI-1: Las pantallas del Sistema de Envío de Encomiendas deben cumplir con el Estándar de Interfaz de Usuario de la Aplicación de Internet de Impacto en el Proceso, Versión 2.0 [3].
- UI-2: El sistema proporcionará un enlace de ayuda de cada página web mostrada para explicar cómo usar esa página.
- UI-3: Las páginas web deberán permitir la navegación completa y la selección de alimentos usando solo el teclado, además de usar combinaciones de mouse y teclado.

5.2. Interfaz de software

- SI-1: Sistema de registro de encomiendas
- SI-1.1: El SEE deberá transmitir las cantidades de alimentos ordenados al sistema de inventario de cafetería a través de una interfaz programática.
- SI-1.2: El COS consultará el sistema de inventario de la cafetería para determinar si un alimento solicitado está disponible.

- SI-1.3: Cuando el sistema de inventario de la cafetería notifique al COS que un alimento específico ya no está disponible, el COS eliminará ese alimento del menú para la fecha actual.

5.3. Interfaces de comunicaciones

- CI-1: El SEE enviará un correo electrónico o mensaje de texto (según la configuración de la cuenta de usuario) al usuario para confirmar el registro de la encomienda, el precio y las instrucciones de entrega.
- CI-2: El SEE deberá enviar un correo electrónico o mensaje de texto (según la configuración de la cuenta de usuario) al usuario para informar cualquier problema con un pedido o entrega de la encomienda.

6. Atributos de calidad

6.1. Requisitos de usabilidad

- USA-1: El SEE permitirá a un usuario recuperar los datos del registro de la encomienda con una sola interacción.
- USA-2: el 95 % de los usuarios podrán registrar la encomienda sin errores en su primer intento.

6.2. Requisitos de desempeño

- PER-1: El sistema debe acomodar un total de 400 usuarios y un máximo de 100 usuarios simultáneos durante la ventana de tiempo de uso pico de 9:00 a. m. a 10:00 a. m. hora local, con una duración media estimada de la sesión de 8 minutos.
- PER-2: El 95% de las páginas web generadas por el SEE se descargarán por completo en 4 segundos desde el momento en que el usuario solicita la página a través de una conexión a Internet de 20 Mbps o más rápida.
- PER-3: El sistema deberá mostrar mensajes de confirmación a los usuarios en un promedio de 3 segundos y un máximo de 6 segundos después de que el usuario envíe la información al sistema.

6.3. Requisitos de seguridad

- SEC-1: todas las transacciones de la red que involucren información financiera o información de identificación personal se cifrarán según BR-33.
- SEC-2: Se requerirá que los usuarios inicien sesión en el COS para todas las operaciones, excepto para ver un menú.
- SEC-3: Solo los administradores de menú autorizados podrán trabajar con menús, según BR-24.
- SEC-4: El sistema permitirá a los Usuarios ver solo los pedidos que hayan realizado.

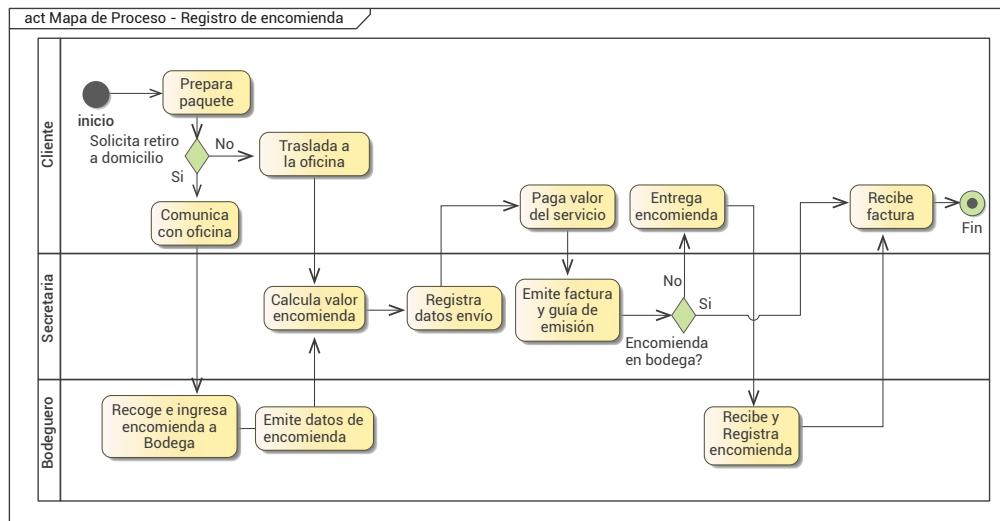
6.4. Requisitos de disponibilidad

- AVL-1: El SEE deberá estar disponible al menos el 98% del tiempo entre las 5:00 a.m. y la medianoche, hora local, y al menos el 90 % del tiempo entre la medianoche y las 5:00 a. m. hora local, excluyendo las ventanas de mantenimiento programadas.

6.5. Requisitos de robustez

- ROB-1: Si la conexión entre el usuario y el SEE se interrumpe antes de que se confirme o finalice un nuevo pedido, el SEE permitirá al usuario recuperar un pedido incompleto y continuar trabajando en él.

7. Anexo A. Mapa de procesos



8. Anexo B. Reglas de negocio

ID	Regla	Tipo de regla	Estática o dinámica	Fuente
RN-01	Cada paquete para entregar como encomienda debe poseer un código de seguimiento único.	Hecho	Estática	N/A
RN-02	El bodeguero debe definir las rutas correctas solo cuando ya se tiene toda la información y documentos requeridos de las mismas.	Hecho	Estática	N/A
RN-03	El sistema debe cumplir con las regulaciones gubernamentales para el uso de personas con discapacidades	Restricción	Dinámica	Reglas del gobierno actual
RN-04	Toda la información generada en el proceso, así como los comprobantes deben ser enviados al correo del cliente.	Restricción	Estática	N/A
RN-05	Cada paquete para entregar debe estar en un grupo de paquetes con un destino de una zona cercana en común.	Restricción	Estática	N/A
RN-06	Todo valor por pagar por envío de encomienda debe realizarse en tiempo real y de manera automática.	Restricción	Estática	N/A
RN-07	El costo del envío se calcula con base en el tipo, tamaño, peso y contenido.	Hecho	Estática	Director

Anexo 7. Lista de verificación para inspección de la especificación de requisitos de software

Identificador del documento	
Autor	
Nombre del proyecto	
Nombre de los revisores	
Fecha de inspección	

Exactitud

- ¿Están los requisitos establecidos de una manera que sea independiente de la solución?
- ¿Están los requisitos libres de errores de contenido y gramaticales?
- ¿Todas las referencias cruzadas internas a otros requisitos son correctas?
- ¿Los requisitos pueden ser utilizados como base para aceptar el sistema?

Claridad

- ¿Puede interpretarse cada requisito de una sola manera?
- ¿Cada requisito está identificado de manera única?
- ¿Están todos los requisitos escritos en un nivel de detalle consistente y apropiado?
- ¿Los requisitos son lo suficientemente claros como para ser entregados a un grupo independiente para el diseño y la implementación y todavía deben ser entendidos con una explicación mínima?
- ¿Están los requisitos escritos de forma concisa (es decir, lo más corto posible sin perder el sentido)?
- ¿Cada requisito es único y no está duplicado por ningún otro requisito?

Completo

- ¿Todo el *hardware* e interfaces externas están definidas?

- ¿Están especificadas todas las entradas y salidas del sistema, incluyendo su fuente, exactitud, rango de valores y frecuencia?
- ¿Están especificadas todas las tareas que el usuario necesita realizar?
- ¿Cada tarea indica los datos utilizados en la tarea y los datos resultantes de la tarea?
- ¿Están documentadas todas las reglas de negocio para las tareas del usuario?
- ¿Falta alguna información necesaria de un requisito? Si es así, ¿se identifica como "por determinar (TBD)"?
- ¿Están especificados todos los atributos de calidad operacional necesarios (por ejemplo, rendimiento, usabilidad, fiabilidad)? ¿Cada una de ellas precisa las escalas de medida?
- ¿Se especifican todos los atributos necesarios de calidad de despliegue (por ejemplo, escalabilidad, disponibilidad y flexibilidad)? ¿Cada una de ellas precisa las escalas de medida?
- ¿Se han definido atributos importantes (por ejemplo, estado, propietario de la fuente, liberación, etc.) para los requisitos?
- ¿Los requisitos proporcionan una base adecuada para el diseño?
- ¿Los requisitos han sido firmados por el aprobador y se ha creado formalmente la línea base?

Consistencia

- ¿Están todos los requisitos de acuerdo (es decir, desprovistos de conflicto o de contradicción)?
- ¿Se especifica el equilibrio aceptable entre los atributos especificados (es decir, entre el tiempo de respuesta y el valor de los datos)?
- ¿Se han escrito los requisitos en un formato estándar?

Relevancia

- ¿Es necesario cada requisito para lograr la visión del producto?

- ¿Se han identificado los límites, alcance y contexto de cada característica o conjunto de requisitos?
- ¿Se incluye la prioridad de implementación de cada requisito?
- ¿Se relaciona cada requisito funcional con su origen en el entorno del problema o con una justificación que explique su propósito?
- ¿Están documentados los requisitos para establecer una relación entre cada requisito y su posterior diseño, implementación y resultados de prueba?
- ¿Pueden utilizarse los requisitos como base para aceptar el sistema?

Factibilidad

- ¿Es posible desarrollar los requisitos con las tecnologías existentes?
- ¿Se pueden satisfacer los requisitos prioritarios dentro de los recursos aprobados?
- ¿Hay al menos una solución de diseño e implementación que puede implementar correctamente cada requisito?
- ¿Pueden implementarse los requisitos dentro de las limitaciones de conocimiento?

Verificabilidad

- ¿Se verifican los requisitos mediante pruebas, demostraciones, revisiones o análisis?
- ¿Se establece cada requisito de una manera que permita que los criterios de prueba se desarrolle y se realicen para determinar si los criterios se han cumplido?
- ¿Pueden usarse los requisitos para crear planes de prueba, procedimientos y casos?