



Instituto Nacional  
de Tecnologías  
de la Comunicación

# GUÍA AVANZADA DE GESTIÓN DE REQUISITOS

**LNCS**

## AVISO LEGAL

- CMMI® es una marca registrada en la Oficina de Marcas y Patentes de EEUU por la Universidad Carnegie Mellon.
- Las distintas normas ISO mencionadas han sido desarrolladas por la International Organization for Standardization.
- SWEBOK® (Software Engineering Body of Knowledge) es una marca registrada por el IEEE (Institute of Electrical and Electronics Engineers).

Todas las demás marcas registradas que se mencionan, usan o citan en la presente guía son propiedad de los respectivos titulares.

INTECO cita estas marcas porque se consideran referentes en los temas que se tratan, buscando únicamente fines puramente divulgativos. En ningún momento INTECO busca con su mención el uso interesado de estas marcas ni manifestar cualquier participación y/o autoría de las mismas.

Nada de lo contenido en este documento debe ser entendido como concesión, por implicación o de otra forma, y cualquier licencia o derecho para las Marcas Registradas deben tener una autorización escrita de los terceros propietarios de la marca.

Por otro lado, INTECO renuncia expresamente a asumir cualquier responsabilidad relacionada con la publicación de las Marcas Registradas en este documento en cuanto al uso de ninguna en particular y se eximen de la responsabilidad de la utilización de dichas Marcas por terceros.

El carácter de todas las guías editadas por INTECO es únicamente formativo, buscando en todo momento facilitar a los lectores la comprensión, adaptación y divulgación de las disciplinas, metodologías, estándares y normas presentes en el ámbito de la calidad del software.

## ÍNDICE

<b>1.</b>	<b>INTRODUCCIÓN</b>	<b>7</b>
1.1.	¿Por qué necesitamos requisitos?	8
1.2.	¿Qué es un requisito?	8
1.3.	Tipos de requisitos	9
1.3.1.	Requisitos funcionales	10
1.3.2.	Requisitos no funcionales	10
1.4.	Roles y responsabilidades	12
<b>2.</b>	<b>DESARROLLO DE REQUISITOS</b>	<b>14</b>
2.1.	Obtención de requisitos	14
2.1.1.	Roles y responsabilidades	17
2.1.2.	Buenas prácticas	18
2.1.3.	Técnicas de recogida de requisitos	18
2.2.	Definición de requisitos	22
2.2.1.	Documentar los requisitos	23
2.3.	Verificación de requisitos	24
2.4.	Revisión de especificación	25
2.4.1.	Priorización	27
2.4.2.	Evolución de los requisitos	29
<b>3.</b>	<b>GESTIÓN DE LOS REQUISITOS</b>	<b>32</b>
3.1.	Gestión de cambios	33
3.1.1.	Evaluar el impacto	35
3.1.2.	Aceptación del cambio	39
3.1.3.	Implementación del cambio	40
<b>4.</b>	<b>MEJORES PRÁCTICAS</b>	<b>41</b>
4.1.	Mejores prácticas en el desarrollo de requisitos	41
4.1.1.	Documentar el alcance y visión del proyecto	41
4.1.2.	Mantener un glosario del proyecto	41
4.1.3.	Técnicas de obtención de requisitos de usuario	42
4.1.4.	Involucrar a toda la gente implicada	42
4.1.5.	Desarrollo incremental de requisitos	42
4.1.6.	Entender por quién y para quién es cada requisito	42
4.1.7.	Captura de requisitos usando casos de uso	42



Instituto Nacional  
de Tecnologías  
de la Comunicación

4.1.8.	Validar requisitos	42
4.1.9.	Verificar requisitos	43
4.2.	Mejores prácticas en la gestión de requisitos	43
4.2.1.	Priorizar requisitos	43
4.2.2.	Establecer líneas base de los requisitos	43
4.2.3.	Comunicación abierta	43
4.2.4.	Gestión de cambios de los requisitos	44
4.2.5.	Uso de herramientas para la gestión de requisitos	44
4.2.6.	Mantener trazabilidad de requisitos	45
4.2.7.	Establecer un plan de mejora de procesos para la ingeniería de requisitos	45
4.2.8.	Formar a los analistas de requisitos	45
<b>5.</b>	<b>ENFOQUE DE ALGUNOS MODELOS (CMMI® vs SPICE)</b>	<b>46</b>
5.1.	CMMI®	46
5.2.	SPICE	48
<b>6.</b>	<b>ARTEFACTOS RELACIONADOS CON GESTIÓN DE REQUISITOS</b>	<b>50</b>
<b>7.</b>	<b>ANEXO I: PROTOTIPOS</b>	<b>52</b>
7.1.	Evaluar las necesidades de Prototipado	52
7.2.	Definir el Tipo de Prototipo	53
7.2.1.	Prototipo de concepto	53
7.2.2.	Prototipos de viabilidad	53
7.2.3.	Prototipo horizontal	53
7.2.4.	Prototipo vertical	54
7.2.5.	“Storyboard” funcional	54
7.3.	Desarrollar del Prototipo	55
7.4.	Refinar el Prototipo	55
7.5.	Herramientas de Prototipado	56
7.6.	Beneficios y Dificultades	56
<b>8.</b>	<b>GLOSARIO</b>	<b>59</b>
<b>9.</b>	<b>REFERENCIAS</b>	<b>63</b>

## ÍNDICE DE TABLAS

Tabla 1	Roles y responsabilidades en el desarrollo y gestión de requisitos.....	13
Tabla 2	Factores relacionados con la obtención de requisitos .....	15
Tabla 3	Categorías para clasificar la información que proviene del cliente.....	16
Tabla 4	Matriz conflicto entre requisitos .....	26
Tabla 5	Categorizaciones de prioridades de requisitos .....	28
Tabla 6	Matriz hacia atrás / hacia delante .....	38
Tabla 7	Matriz de dependencias.....	39
Tabla 8	Tabla resumen prototipos .....	55



Instituto Nacional  
de Tecnologías  
de la Comunicación

## ÍNDICE DE FIGURAS

Figura 1	Desarrollo de requisitos .....	14
Figura 2	Evolución de los requisitos .....	30
Figura 3	Modelo espiral del proceso de la ingeniería de requisitos .....	31
Figura 4	Proceso de gestión de cambios .....	35

## 1. INTRODUCCIÓN

---

Los mejores productos, desde el punto de vista del usuario, son aquellos creados por desarrolladores que tienen muy claro lo que se pretende conseguir con el producto y cómo obtenerlo. Para llegar a este punto, se debe entender el trabajo del usuario, cómo afectará el producto a su trabajo y cómo se adecuará a los objetivos de la organización.

Lo que hace el producto y las condiciones que debe satisfacer en este contexto son los **requisitos del producto**.

Excepto en unos cuantos casos fortuitos, ningún producto tendrá éxito sin un claro entendimiento previo de sus requisitos. No importa el tipo de trabajo del usuario, ni tampoco qué lenguaje de programación o herramientas de desarrollo serán usadas para construir el producto. Si no se entiende cómo se quiere que sea y funcione el producto, no se podrá construir correctamente. En este caso, el proceso de desarrollo es irrelevante.

Los requisitos del producto deben ser entendidos por todas las partes (cliente y desarrollador) antes de que comience su construcción, o el proyecto fracasará. Sólo cuando se conocen los requisitos correctos se puede diseñar y construir un producto que permita a los usuarios hacer su trabajo de forma que satisfaga las necesidades del negocio.

Por desgracia, los requisitos no son siempre entendidos correctamente. Existen muchos estudios que lo demuestran, por ejemplo, de acuerdo con el Instituto Nacional de Estándares y Tecnología de Estados Unidos, los **requisitos incompletos, imprecisos y conflictivos normalmente causan un 70% de los defectos de una aplicación**.

Aunque los desarrolladores tienen la oportunidad de subsanar la mayoría de los errores en la definición de requisitos, muchas veces se precipitan o hacen suposiciones que, como consecuencia, dan lugar a un producto erróneo. Esto puede ser debido a varios motivos como falta de tiempo o de presupuesto, y como resultado, el coste del producto se multiplica, cosa que no ocurriría si se cumpliera con un análisis correcto de los requisitos.

El primer paso a llevar a cabo será el proceso de búsqueda de requisitos. Este proceso es una exploración minuciosa del producto con la intención de descubrir su funcionalidad y comportamiento. El resultado de este proceso es una descripción, preferiblemente escrita, de los requisitos que serán usados como entradas del diseño del producto.

Los requisitos normalmente se expresan de una manera tecnológicamente neutra para evitar influenciar el diseño de la solución. Los requisitos son un puro comunicado de las necesidades del negocio sin ninguna predisposición sobre la implementación.

El papel del diseño del producto es traducir los requisitos en un plan en el que se detalle cómo será construido el producto final. Para que el producto tenga éxito es importante que la decisión final sobre el diseño no sea tomada hasta que los requisitos más relevantes estén claros y definidos.

Una vez que el producto se ha construido y se empieza a usar, empieza a evolucionar. Los usuarios demandan más funcionalidad y el producto debe ser capaz de crecer alojando las nuevas demandas. La evolución de un producto y de sus requisitos es un proceso que no se

puede negar y que hay que aceptar. Los requisitos de un producto no se congelan en el momento que se construye sino que evolucionan durante un periodo de tiempo.

La ingeniería de requisitos se refiere a la rama de ingeniería de software que está relacionada con los objetivos reales y las funciones de los sistemas de software. La ingeniería de requisitos también está relacionada con las relaciones de estos factores reales con las especificaciones exactas del comportamiento del software, y la evolución de los requisitos en el tiempo y durante el ciclo de vida de desarrollo del software.

### 1.1. ¿POR QUÉ NECESITAMOS REQUISITOS?

El producto es lo que se quiere construir, y sus requisitos deben ser entendidos antes de que comience a construirse. Los requisitos correctos son aquellos que proceden del buen entendimiento del trabajo que el producto soportará. Sólo conociendo los requisitos correctos se podrá diseñar y construir el producto correcto, y así permitirá a los usuarios del producto hacer su trabajo de forma que satisfaga sus necesidades de negocio.

A pesar de todo esto, los requisitos no son siempre entendidos. Existen muchas estadísticas y estudios que demuestran que la mayoría de los errores se originan en la etapa de requisitos, y que aunque los desarrolladores tienen la oportunidad de corregirlos eligen construir el producto de forma precipitada. Como consecuencia, ellos pagan por el producto mucho más que su precio real. Esto se evitaría si el desarrollo y análisis de requisitos se hubiese realizado de la forma correcta desde el primer momento.

El coste de una buena recogida de requisitos y análisis del sistema a desarrollar es menor comparado con el coste resultante de tener requisitos pobres, es decir, el coste de reparar productos deficientes o de poca calidad, el coste de los proyectos cancelados y el coste de haber perdido la oportunidad de tener el producto correcto en el momento correcto.

El fundamento básico de cualquier software recae sobre su proceso de ingeniería de requisitos. El éxito o fallo del software depende casi siempre en cómo de bien se hayan capturado, entendido y usado los requisitos como base para el desarrollo. La ingeniería de requisitos es la fase de la ingeniería del software donde se definen las propiedades y la estructura del software.

La ingeniería de requisitos debe tener un entendimiento adecuado del dominio para alcanzar los requisitos esenciales y documentarlos de forma apropiada. Debido a la diversidad de dominios y situaciones es importante conocer diferentes tipos de métodos para llevar a cabo la ingeniería de requisitos.

### 1.2. ¿QUÉ ES UN REQUISITO?

Hasta el momento hemos usado el término requisito de forma muy general pero ¿qué definición se podría dar y qué tipos de requisitos existen?

Un **requisito** es algo que el producto debe hacer o una característica que debe tener. Un requisito existe por el tipo de demanda que tiene el producto o porque el cliente quiere que el requisito sea parte del producto entregado. La tarea de todo analista de requisitos es

hablar con la gente, entenderla, escuchar lo que dicen y también lo que no dicen, para entender lo que necesitan.

El trabajo de recopilar los requisitos no es sólo del analista de requisitos. Tanto los usuarios como los demás agentes que intervienen en el negocio colaboran con él. El analista tiene que entender lo que se dice, traducir este conocimiento en requisitos del producto y hacer que el trabajo sea más fácil.

Por lo tanto, los pasos a seguir para identificar y definir los requisitos de un producto, de forma genérica, se podrían enumerar de la siguiente manera:

1. **Observar y entender el trabajo desde el punto de vista del usuario.** Para ello, es necesario trabajar con el usuario, estudiar su trabajo y preguntarle acerca de lo que hace y por qué lo hace...
2. **Interpretar el trabajo del usuario** y la forma en la que él lo describe ya que él es el experto del mismo.
3. **Inventar mejores formas de hacer el trabajo.** El analista de requisitos interpreta lo que el producto debe hacer para satisfacer el trabajo.
4. **Plasmar estos resultados en forma de especificación de requisitos** de tal forma que sean fáciles de entender para todos los implicados en el proyecto. El analista debe asegurarse de que él y el resto del equipo de trabajo tienen el mismo concepto del producto que se va a crear.

### 1.3. TIPOS DE REQUISITOS

Existen distintos tipos de requisitos, desde los requisitos a más alto nivel, referentes al propio negocio (**requisitos de negocio**) hasta los requisitos que se recogen directamente del usuario (**requisitos de usuario**), o los **requisitos del sistema/software**.

Los **requisitos de negocio** dan una descripción a alto nivel de lo que el sistema debe hacer. Representan los objetivos, la base del negocio, estrategias, visión, alcance y el valor esperado del desarrollo del software, es decir, proporcionan la dirección que ha de seguir el proyecto y forman la base de los requisitos de usuario. Los **requisitos de usuario** entran en más detalle. Son una descripción de las tareas que el sistema ha de ejecutar cuando el usuario opera con él. Estos requisitos describen la funcionalidad necesaria para satisfacer tareas específicas, necesidades operacionales y grupos de usuarios. Otro tipo de requisitos son los **requisitos de sistema**, que definen las funcionalidades y características que debe tener el sistema para satisfacer tanto los requisitos de negocio como los de usuario. Este tipo de requisitos van a servir como base para llevar a cabo la arquitectura, diseño y planes de pruebas del sistema.

Por último tenemos las **restricciones**. Las restricciones son condiciones que limitan las elecciones disponibles al diseñador o programador. Representan otro tipo de requisitos no funcionales que se deberían documentar en la especificación de requisitos. Hay que intentar prevenir al cliente de que imponga restricciones innecesarias, ya que pueden inhibir la creación de la solución mejor.

En este apartado se van a entrar más en detalle sobre los **requisitos de sistema/software**. Los requisitos de sistema/software contienen requisitos funcionales y no funcionales del sistema a un alto nivel de arquitectura.

### 1.3.1. Requisitos funcionales

Los requisitos funcionales especifican qué debe hacer el producto. Describen las acciones que el producto debe llevar a cabo. Hay que pensar en estos requisitos como aquello que el producto debe hacer desde el punto de vista del negocio. Los agentes del negocio describirán estas acciones que el producto deberá realizar para completar alguna parte de su trabajo. Los requisitos funcionales también pueden verse como requisitos independientes a cualquier tecnología. Son la esencia del trabajo.

Cuando llega el momento de diseñar una solución para los requisitos funcionales, el diseñador añade requisitos técnicos que son necesarios para la tecnología usada en la solución. Los requisitos técnicos algunas veces se unen a los **requisitos de negocio**, y nos podemos referir a ambos como requisitos funcionales, ya que se refieren a funciones del diseño o de la solución. Sin embargo, es más preciso y menos confuso separar los requisitos técnicos de los requisitos funcionales de negocio.

La especificación de requisitos sirve de contrato a la hora de construir el producto. Por lo tanto, los requisitos funcionales deben contener suficiente detalle para que el desarrollador construya el producto correcto con una mínima explicación de los analistas de requisitos y demás agentes del negocio.

Existen distintos métodos que pueden ayudar a identificar los requisitos funcionales del producto. Uno de los más utilizados son los escenarios de los que se hablará más adelante.

### 1.3.2. Requisitos no funcionales

Los requisitos no funcionales son las cualidades que debe tener el producto. Estos requisitos hacen que el producto sea atractivo, útil, rápido, fiable o seguro. Estas propiedades no son requeridas porque no son actividades funcionales del producto, pero son deseables, ya que el cliente espera que las actividades sean ejecutadas de cierta manera y con un específico grado de calidad.

Los requisitos no funcionales no alteran la funcionalidad esencial del producto pero pueden añadir más. Es más fácil pensar en los requisitos funcionales como aquellos que hacen que el producto realice el trabajo y los no funcionales como aquellos que hacen que el producto dé cierto carácter al trabajo.

Los requisitos no funcionales forman una parte significativa de la especificación de requisitos. Además, las propiedades no funcionales como la usabilidad o la conveniencia, pueden ser la diferencia entre un producto aceptado y que guste al usuario y un producto que no se utilice. La usabilidad de un producto crea una importante diferencia en cuanto a la aceptación del cliente. El conocido 'look and feel' es importante para un gran número de usuarios, y la mantenibilidad o falta de la misma elimina muchas frustraciones a los responsables del producto.

Como se puede observar, los requisitos no funcionales son tan importantes como los funcionales. Cada producto tiene un estilo que lo diferencia del resto. Alguien puede comprar una televisión porque le transmita una sensación distinta, porque sea estéticamente mejor o porque sea más fácil de usar que otra. Todo esto forma el carácter del producto. Los requisitos no funcionales describen este carácter.

Los requisitos no funcionales se pueden clasificar de diferentes formas, por ejemplo:

- **Look and feel:** Engloba aspectos referentes a la apariencia del producto
- **Usabilidad y humanidad:** Engloba aspectos referentes a la usabilidad del producto y cualquier consideración especial necesaria para una mejor experiencia del usuario.
- **Ejecución:** Engloba aspectos referentes a cómo de rápido, seguro, disponible y exacto debe ser el producto.
- **Operacional:** Engloba aspectos referentes al entorno operativo del producto y cualquier consideración que deba tenerse en cuenta para este entorno.
- **Mantenibilidad y soporte:** Engloba aspectos referentes a los cambios esperados y el tiempo necesario para hacerlos; también la especificación del soporte que se dará al producto.
- **Seguridad:** Engloba aspectos referentes a la seguridad, confidencialidad, y facilidad de recuperación del producto.
- **Cultural y político:** Engloba aspectos referentes a requisitos especiales que se deben a la cultura y costumbres de la gente involucrada con el producto.
- **Legal:** Engloba aspectos referentes a las leyes y estándares que se aplican al producto.

Si todavía queremos clasificar aún más estas categorías, podríamos dividir los requisitos no funcionales en tres grandes grupos:

- Requisitos de producto (**usabilidad, eficiencia, fiabilidad, portabilidad, seguridad y escalabilidad**).
- Requisitos organizacionales (**entrega, implementación, estándares, recursos...**)
- Requisitos externos (**interoperabilidad, legislación, privacidad, seguridad...**)

Una vez más, estos son sólo algunos ejemplos de clasificaciones que se pueden dar a los requisitos. Lo más importante es tener claro que los requisitos no funcionales son las propiedades, o cualidades que el producto debe tener. En algunos casos, los requisitos no funcionales son críticos para el éxito del producto.

Los requisitos no funcionales son normalmente definidos después de la funcionalidad del producto. Una vez que conocemos lo que el producto va a hacer, podemos determinar cómo se va a comportar, qué cualidades va a tener, cómo de rápido, útil, legible y seguro será.

## 1.4. ROLES Y RESPONSABILIDADES

Los roles y responsabilidades que toman parte en el proceso de desarrollo y gestión de los requisitos vienen reflejados en la siguiente matriz RACI.

### **Roles:**

RM Gerente

PM Jefe de proyecto

SQA Responsable de Aseguramiento de la Calidad del Software

CCB Equipo de Control de Configuración

Actividad	Responsabilidad				Salida
	Preparación	Revisión	Aprobación	Responsabilidad	
Identificar los proveedores de requisitos y las autoridades firmantes	PM			PM	
Documentar los requisitos de usuario y de negocio	Equipo de proyecto	Cliente /RM	Cliente /RM	PM	Requisitos de usuario y de negocio
Documentar los requisitos de software/ sistema	Equipo de proyecto	Cliente /RM	Cliente /RM	PM	Especificación de requisitos software y especificación de casos de uso
Preparar y actualizar la matriz de trazabilidad de requisitos	Equipo de proyecto	SQA	PM/SQA	PM	Matriz de trazabilidad de requisitos
Analizar los requisitos				Equipo de proyecto/ PM	Matriz de trazabilidad de requisitos
Verificar y validar los requisitos. Obtener acuerdo		Cliente	Cliente	PM	Requisitos de usuario y de negocio, especificación de requisitos software, especificación de casos de uso
Línea base de los requisitos		SQA		PM	Línea base de los requisitos de usuario y de negocio, de la especificación de requisitos software y de la especificación de casos de uso
Gestionar cambios a los requisitos	PM	SQA	CCB	PM	Registro de peticiones de cambio, matriz de trazabilidad requisitos.

**Tabla 1 Roles y responsabilidades en el desarrollo y gestión de requisitos**

## 2. DESARROLLO DE REQUISITOS

Antes de entrar a describir el proceso de desarrollo de requisitos y las actividades que tiene relacionadas, vamos a situar el desarrollo de requisitos dentro de la gran área de ingeniería de requisitos.

La ingeniería de requisitos comprende el desarrollo y gestión de requisitos.

- El **desarrollo de requisitos** implica entender los requisitos de negocio, identificar los requisitos de usuario y trasladar los requisitos de usuario y de negocio a requisitos de sistema/software.
- La **gestión de requisitos** implica gestionar los cambios de requisitos y mantener la consistencia entre los requisitos y otros productos de trabajo del proyecto.

Las principales actividades realizadas durante el proceso de desarrollo de requisitos son:

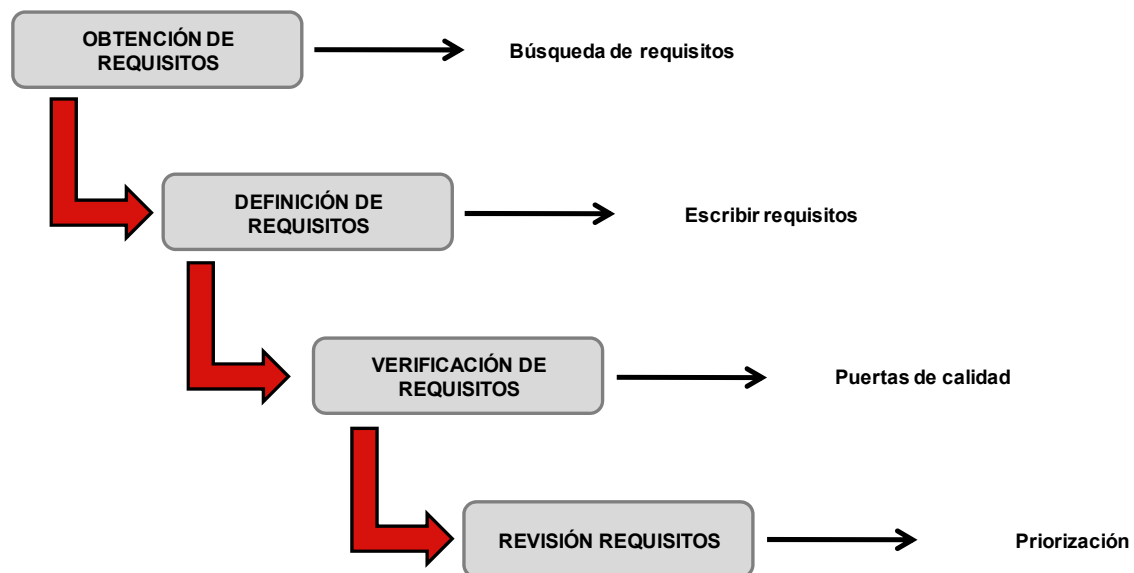


Figura 1 Desarrollo de requisitos

### 2.1. OBTENCIÓN DE REQUISITOS

La obtención de requisitos se define como el proceso de identificar las necesidades del negocio, solucionando las posibles disparidades entre las personas involucradas en el mismo, con el propósito de definir y destilar los requisitos para cumplir las restricciones impuestas por las distintas partes.

Un buen proceso de obtención de requisitos soporta el desarrollo de la especificación de los requisitos, de tal forma que tengan los siguientes **atributos**:

- Los requisitos han de ser completos, consistentes y han de estar dentro del alcance del proyecto

- Los requisitos son identificados de forma única y han de priorizarse
- Cumplen con los objetivos de los clientes
- Son viables y apropiados para el desarrollo
- Están indicados de forma clara y no ambigua
- Los requisitos han de ser “testeables”, es decir, es necesario que sean comprobables para poder validarlos y verificarlos en etapas posteriores. Deben tener capacidad de prueba.

En la obtención de requisitos se necesita tener un entendimiento de la organización en la que se va a utilizar la aplicación que se va a desarrollar, así como un entendimiento de la misión del sistema dentro del contexto de la organización. La siguiente tabla muestra algunos de los **factores a tener en cuenta** durante este proceso. Estos factores se han clasificado en 3 niveles: organizacional, factores de entorno y de proyecto.

Factores organizacionales	Factores de entorno	Factores de proyecto
Personas que presentan las entradas del sistema objetivo	Restricciones de hardware y software impuestos en el sistema objetivo	Los atributos de las distintas personas involucradas en el proyecto (usuarios finales, patrocinadores, desarrolladores y analistas de requisitos). Ejemplos de estos factores son estilos de gestión, experiencia en el dominio...
Usuarios de las salidas del sistema objetivo	La madurez del dominio del sistema objetivo	Las restricciones impuestas por las personas involucradas en el proceso de obtención de requisitos
Formas en las que el sistema objetivo cambiará la forma de hacer negocio de la organización	El papel del sistema objetivo dentro de un sistema mayor	

**Tabla 2 Factores relacionados con la obtención de requisitos**

La obtención de requisitos es un proceso complejo. Hay que tener claro que no hay que esperar a que los clientes presenten los requisitos con una lista de necesidades concisa, completa y bien organizada.

Los analistas de requisitos deben clasificar la información que surge de la voz del cliente en varias categorías, de tal forma que se pueda documentar de una forma apropiada y se pueda usar de la forma más sensata posible. A continuación se propone una serie de categorías en las que se puede clasificar esta información:

Categorías	Definición
Requisitos de negocio	Cualquier cosa que describa los beneficios financieros, de mercado u otros beneficios de negocio que los clientes o la organización puedan obtener de un producto.
Casos de uso o escenarios	Las declaraciones generales de metas de usuario o tareas de negocio que necesitan realizar con el sistema pueden ser probablemente casos de uso, mientras que las descripciones de tareas específicas representan escenarios de uso
Reglas de negocio	Cuando un cliente dice que algunas actividades sólo las pueden realizar ciertos individuos o roles, bajo unas condiciones concretas, pueden estar describiendo una regla de negocio.  Son principios operativos sobre un proceso de negocio.
Requisitos funcionales	Los requisitos funcionales describen los comportamientos observables que el sistema debe exhibir como respuesta del sistema
Atributos de calidad	Las afirmaciones que indican cómo de bien realiza el sistema alguna tarea o cómo de bien deja al usuario realizar alguna acción, un tipo de requisito no funcional.
Requisitos de la interfaz externa	Describen las conexiones entre el sistema y el resto del universo
Restricciones	Son condiciones que limitan las opciones disponibles para el diseñador o programador.  Representan otro tipo de requisitos no funcionales que se deberían documentar en la especificación de requisitos
Definiciones de datos	Siempre que los usuarios describan el formato, los valores permitidos, o los valores por defecto para cada ítem de datos o la composición de una estructura de datos de negocio compleja, están presentando una definición de datos.  Agrupar todo esto en un diccionario de datos proporciona una referencia que los participantes del proyecto pueden utilizar durante el desarrollo y mantenimiento del proyecto.
Ideas de solución	Si un cliente describe una forma específica de interactuar con el sistema para llevar a cabo alguna acción se trata de una solución sugerida, no un requisito.

**Tabla 3** *Categorías para clasificar la información que proviene del cliente*

La información que no encaje en uno de los grupos anteriores puede representar un requisito de proyecto que no es de software, como puede ser un requisito de dar formación a los usuarios del nuevo sistema.

### 2.1.1. Roles y responsabilidades

Al principio de todo proyecto, el analista de requisitos se centra en entender el negocio para crear un producto que solvete las necesidades del mismo. Además se debe llegar a un acuerdo sobre las condiciones que ha de tener el producto para que éste sea óptimo.

Los requisitos provienen de las personas. El **analista de requisitos** será el encargado de hablar con la gente, escuchar lo que dicen, y entender lo que necesitan. Por lo tanto el encargado de llevar a cabo la búsqueda de requisitos será el analista de requisitos. Pero el analista no trabaja solo, sino que los usuarios y otros involucrados en el negocio colaborarán en la recogida de los mismos.

El analista es un traductor. Entiende lo que los usuarios y los involucrados en el negocio están diciendo sobre el trabajo, y después traduce este conocimiento en requisitos para el producto. Las tareas asignadas al analista de requisitos son:

- **Observar y aprender el trabajo** que realizan los usuarios, y entenderlo desde su punto de vista. Para ello será necesario realizarles preguntas sobre lo que están haciendo y por qué lo están haciendo.
- **Interpretar el trabajo.** El usuario es el experto en el trabajo que lleva a cabo. Sin embargo, el analista debe filtrar la descripción para obtener la esencia real del trabajo.
- **Inventar mejores maneras de hacer el trabajo.** Una vez que el analista de requisitos captura la esencia, interpreta lo que el producto debe hacer para satisfacer esa parte del trabajo. Al mismo tiempo, y con la ayuda de los involucrados en el negocio, crea un producto para mejorar el trabajo.
- **Registrar los resultados** en la especificación de requisitos y en modelos de análisis. El analista de requisitos debe asegurarse de que los involucrados en el negocio y él tienen el mismo entendimiento del producto, y de que los involucrados en el negocio estén de acuerdo y entiendan que ese es el producto necesario.

Existen distintas técnicas para ayudar en la tarea de búsqueda de requisitos. No existe una única técnica que funcione en todas las situaciones, por lo tanto, otra de las tareas del analista de requisitos será seleccionar la técnica que mejor se ajuste a cada situación.

Durante esta actividad también deberá tener en cuenta a los **involucrados en el negocio**. Ellos son conscientes de ciertos requisitos pero no de todos. Existen ciertos requisitos que están tan arraigados a su trabajo, que se han olvidado de que existen. La razón por la que existen requisitos de los que los involucrados en el negocio no son conscientes, es la falta de conocimiento acerca de ciertas tecnologías o porque nunca han visto su trabajo de la manera en que el analista de requisitos se lo mostrará. Parte de la responsabilidad del analista de requisitos será sacar a la luz todos estos requisitos.

Es importante tener claro que es más barato y efectivo descubrir y capturar todos los requisitos durante esta primera fase de búsqueda de requisitos. Aquellos requisitos que no sean descubiertos durante esta fase aparecerán cuando el usuario empiece a operar con el

producto, y por lo tanto será mucho más costoso hacer los cambios necesarios para acomodar los nuevos requisitos encontrados.

### 2.1.2. Buenas prácticas

A continuación se proponen algunas buenas prácticas a tener en cuenta durante el proceso de recogida de requisitos:

1. Durante la obtención de requisitos se debe determinar si el **alcance** del producto está mal definido.
  - Si el alcance es muy grande, se recogen más requisitos de los que realmente se necesitan entregar y el proceso se puede alargar.
  - Si el alcance del proyecto es demasiado pequeño, los clientes presentarán necesidades claramente importantes que no estarán dentro del alcance actual establecido para el producto.

2. Se dice con frecuencia que los requisitos expresan lo que el sistema tiene que hacer, mientras que cómo se ha de implementar la solución es parte del diseño.

La obtención de requisitos debería centrarse en el qué, pero hay un **área dudosa entre el análisis y el diseño**. Se pueden usar “cómos” hipotéticos para clarificar y refinar el entendimiento de lo que los usuarios necesitan.

3. Los **modelos de análisis, escenarios y prototipos** ayudan a hacer más tangibles los conceptos que se expresan durante el proceso de obtención de requisitos y proporcionan una forma de encontrar errores u omisiones.
4. El proceso de obtención de requisitos en el que están implicados **demasiados participantes** se pueden volver demasiado lentos.

Pueden tener lugar discusiones extendidas de detalles innecesarios y puede ser difícil ponerse de acuerdo en cómo ha de funcionar cada caso de uso.

Reducir el número de participantes a las personas que representan los roles principales del proyecto puede ayudar a acelerar el proceso.

5. En cambio, recoger información de **demasiada poca gente** puede ser también un problema.

Puede conducir a pasar por alto requisitos que son importantes para algún tipo de usuario o a especificar requisitos que no representan las necesidades de una mayoría de los usuarios.

### 2.1.3. Técnicas de recogida de requisitos

Existen múltiples técnicas que pueden ayudar a la hora de recoger los requisitos de un producto.

Algunas de ellas son bastante conocidas como por ejemplo: realizar entrevistas, reuniones, cuestionarios.... y otras como la utilización de escenarios o prototipos, que quizás no son tan comunes.

#### **2.1.3.1. Entrevistas**

Esta es una técnica simple y directa. Preguntas libres de contexto pueden ayudar a conseguir entrevistas libres. El objetivo es prevenir condicionar la respuesta del usuario a las preguntas. Entonces, puede ser apropiado buscar requisitos no descubiertos explorando soluciones. La convergencia en algunas necesidades comunes iniciará un repositorio de requisitos para usar durante el proyecto.

Se puede usar esta técnica para:

- reunir información sobre un sistema existente
- determinar requisitos de un sistema nuevo
- clarificar especificaciones funcionales
- obtener información de entorno sobre la organización del cliente
- obtener realimentación respecto a la usabilidad

#### **2.1.3.2. Reunión**

Las reuniones se usan para comunicar y promover la solución de un problema en grupo. Esta técnica se puede usar para cualquier tipo de reunión, desde reuniones pequeñas de dos o tres participantes hasta reuniones a gran escala. Cuanto más grande sea mayor deberá ser el nivel de formalidad. La efectividad de la reunión se puede evaluar mediante una encuesta a los participantes.

#### **2.1.3.3. Formulario de recogida de observaciones**

El formulario de recogida de observaciones puede usarse para empezar a analizar un proceso de negocio reuniendo hechos que prueben una teoría u opinión y para empezar a detectar patrones en un proceso.

El formulario de recogida de observaciones más efectivo:

- Contiene información homogénea
- Muestra datos de forma visual en un formato que revela patrones subyacentes
- Es fácil de entender y de usar

#### **2.1.3.4. Cuestionarios y Encuestas**

Esta técnica se usa cuando la información proviene directamente de la gente, como actitudes, valores, hábitos o preferencias personales:

- Para determinar la funcionalidad de una aplicación existente

- Para determinar el uso de una tecnología existente

Usar cuestionarios por mail, de administración personal cuando:

- La gente está alejada geográficamente
- Haya mucha gente para entrevistar
- La información a reunir sea simple
- El anonimato es importante

Usar cuestionarios en entrevista cuando:

- Se requiera información en profundidad
- Se quiere comprobar los distintos puntos de vista de las personas
- Se requiere una tasa de respuesta mayor que la generada por cuestionarios vía mail

### **2.1.3.5. Brainstorming**

El brainstorming o lluvia de ideas implica tanto la generación como la reducción de ideas. Las ideas más creativas e innovadoras resultan con frecuencia de la combinación de ideas aparentemente sin relación. Se pueden utilizar algunas técnicas de votación para priorizar las ideas creadas. Aunque se prefieren los brainstorming en vivo, en algunas situaciones puede ser viable el brainstorming basado en web.

### **2.1.3.6. Casos de Uso**

Los casos de uso identifican el quién, el qué y el cómo del comportamiento del sistema. Los casos de uso describen las interacciones entre el usuario y el sistema, centrándose en qué hace el sistema para el usuario. El modelo de caso de uso describe la totalidad del comportamiento del sistema.

Los pasos que hay que seguir para llevar a cabo un caso de uso serían:

1. **Identificar actores:** los actores son entidades externas que interactúan con el sistema.

Identificar actores permite a los desarrolladores entender diferentes perspectivas del sistema. Los actores incluyen usuarios primarios del sistema, usuarios secundarios y sistemas hardware/software externos que interactúan con el sistema que se encuentra bajo desarrollo.

2. **Identificar escenarios:** un escenario es una descripción narrativa de lo que hacen y experimentan las personas cuando usan el sistema y las aplicaciones.

Un escenario incluye tareas primarias del sistema, datos que crean los actores, almacenamientos, eliminaciones, modificaciones en el sistema, cambios externos de lo que el sistema necesita saber, cambios o eventos sobre los que el actor debe estar informado.

3. **Identificar casos de uso:** cuando el esbozo de los actores está completo, el siguiente paso es buscar los casos de uso del sistema. Un caso de uso es una generalización de varios escenarios.

Representa un flujo completo de eventos. Los primeros casos de uso son muy preliminares, y es indudable que cambiarán muchas veces hasta que sean estables. Si la visión o los requisitos del sistema son deficientes, o si el análisis del sistema es impreciso, la funcionalidad del sistema no estará clara. Por lo tanto, hay que preguntarse constantemente si se han encontrado los casos de uso correctos. Es muy común tener que añadir, eliminar, combinar y dividir los casos de uso antes de llegar a una versión final.

La mejor forma de encontrar casos de uso es considerar qué espera cada actor del sistema. Hay que recordar que el sistema sólo existe para sus usuarios, y debería basarse en las necesidades de los usuarios. Hay que reconocer muchas de las necesidades de los usuarios a través de los requisitos funcionales del sistema. Para cada actor hay que hacer las siguientes preguntas:

- ¿Cuáles son las tareas primarias que el actor quiere que realice el sistema?
- ¿El actor creará, almacenará, cambiará, eliminará o leerá datos en el sistema?
- ¿El actor necesita informar al sistema de cambios externos?
- ¿El actor necesita ser informado sobre ciertas ocurrencias en el sistema?

Las respuestas a estas preguntas representan el flujo de eventos que identifican casos de uso candidatos. No todos constituyen casos de uso separados; algunos pueden ser modelados como variantes del mismo caso de uso.

### 2.1.3.7. Prototipos y escenarios

Durante esta etapa, los analistas se ayudan de escenarios, prototipos y otros modelos que les puedan servir de ayuda en la identificación y análisis de los requisitos.

#### Prototipos

Algunas veces los analistas de requisitos no pueden continuar su trabajo porque les faltan datos. Esto puede ser debido a:

- el usuario no ha dado suficientes detalles
- el producto es tan innovador que nadie conoce realmente sus requisitos

En esos casos, el analista de requisitos o el resto de personas involucradas necesitan trabajar con algo más concreto que una lista de requisitos escritos, y para ello utilizan un prototipo.

Un prototipo es un borrador de un producto potencial o de una parte del mismo. Es una simulación de los requisitos. Hay dos aproximaciones para construir prototipos:

- Prototipos de **alta fidelidad**, que usan herramientas de software especializadas.

- Prototipos de **baja fidelidad**, en los que se usa papel y bolígrafo o cualquier otro medio familiar.

Los equipos usan normalmente prototipos de baja fidelidad porque pueden generarse rápidamente y los usuarios valoran la espontaneidad y la inventiva de estos prototipos. Para más información acerca de los prototipos se puede acudir al [Anexo I](#).

### Escenarios

Los escenarios son una descripción paso a paso de la funcionalidad de un caso de uso del producto o del negocio, sin demasiado detalle, con el objetivo de hacer entender cómo funciona este caso de uso.

Los pasos en los escenarios son fácilmente reconocibles por las personas involucradas en el negocio, ya que están escritos en el lenguaje que utilizan dichas personas.

No hay un número fijo de pasos a completar por escenario. Sin embargo, un número demasiado elevado (más de 50) hará que el escenario sea difícil de seguir y entender, con lo que una buena pauta sería entre 3 y 10 pasos.

Los analistas consideran los escenarios como un medio neutral, entendible por todo el mundo, y los utilizan para llegar a acuerdos con los responsables del proyecto sobre el trabajo que se tiene que hacer. Una vez que ellos los aprueban, los escenarios forman los cimientos de los requisitos.

## 2.2. DEFINICIÓN DE REQUISITOS

Conseguir que los requisitos estén claramente definidos puede ser difícil. Para ello es importante tener muy en cuenta la perspectiva del usuario. El problema es que a menudo los usuarios están demasiado ocupados como para definir con precisión lo que necesitan. Algunas organizaciones resuelven el problema del '**usuario ocupado**' creando un perfil – el ya nombrado *analista de negocio* - que represente las necesidades de la aplicación del usuario.

Si no se toma el tiempo necesario para definir lo que se necesita, más que ahorrar tiempo se perderá. Al final del proyecto se tendrán que redefinir los requisitos incompletos para obtener una solución satisfactoria al problema que no hemos podido resolver.

Si falta tiempo, la **reutilización** de los requisitos puede ser una buena solución. Los requisitos de un producto nunca serán completamente únicos. Es aconsejable que antes de empezar cualquier proyecto se revisen proyectos ya finalizados para ver si contienen material potencialmente reutilizable. En ocasiones hay requisitos que se pueden reutilizar sin ni siquiera alterarlos pero lo más habitual es encontrar requisitos que, aunque no son exactamente lo que se quiere, son una buena base para construir los requisitos deseados.

La ventaja de esta reusabilidad es que, una vez que un requisito ha sido especificado satisfactoriamente para un producto y que el producto ha tenido éxito, el requisito no tendrá que volverse a inventar, podrá ser utilizado las veces que se desee.

### 2.2.1. Documentar los requisitos

Un gran problema que hay en el desarrollo de sistemas es el de mal entender los requisitos. Para evitar este dilema, el analista debe definir los requisitos de tal forma que puedan ser probados, y asegurarse de que los involucrados en el negocio estén de acuerdo con los requisitos escritos.

Una forma de comprobarlo es seleccionando un determinado número de requisitos de forma aleatoria, y pidiendo a los agentes del negocio que los interpreten uno cada vez. Si todo el mundo está de acuerdo con su significado para el primer requisito se coge otro y se repite el proceso hasta que quede claro que la especificación es aceptable. Si se ve que hay muchas discordancias entre las interpretaciones de la gente se debería plantear reescribir la especificación.

Aunque escribir los requisitos puede parecer una tarea tediosa, es la única manera de asegurar que la esencia de los requisitos ha sido capturada correctamente, y que esto pueda ser probado.

Los requisitos son escritos para los involucrados en el negocio, es decir, los requisitos deben ser escritos utilizando lenguaje de negocio de tal forma que las personas que no sean técnicos puedan entenderlos y verificar que son correctos. Si por ejemplo, la fuente de los requisitos procede de documentos o de ideas obtenidas en entrevistas, se debería tener conciencia de la ambigüedad y mal entendimiento que procede de estas fuentes. Es importante prestar especial atención al significado de las palabras para reducir su ambigüedad. Algunas **buenas prácticas** que pueden aplicarse a la hora de definir requisitos son:

- Los requisitos son escritos de una forma tecnológicamente neutra, es decir, especifican lo que el producto hace y no qué tecnología se usará para crearlo.
- Su especificación no debe ser ambigua.
- Eliminar todos los pronombres de la especificación de requisitos, sustituyéndolos por los sujetos.
- Tener cuidado con adjetivos y adverbios ya que pueden llevar a confusiones.
- Se debe evitar usar palabras tales como 'debería' al escribir los requisitos ya que da a entender que el requisito es opcional.
- Al escribir los requisitos una buena técnica es leerlos en alto. Y si es posible pedir a alguien que los lea.
- Confirmar que los involucrados en el negocio tienen el mismo entendimiento acerca de los requisitos que la persona que los escribe.

Otra técnica para escribir correctamente los requisitos es utilizar una convención de nombres y definiciones común dentro de la organización. De esta forma se aseguran que en todos los proyectos se está usando el mismo vocabulario. Un mal uso del lenguaje puede llevar a un mal entendimiento, horas de trabajo perdido, una mala comunicación entre miembros del equipo, y en definitiva una especificación de requisitos de poca calidad.

Algunas buenas prácticas relacionadas con la **convención de nombres** son las que se describen a continuación:

- Realizar un **glosario** que defina los **términos** más importantes que van a utilizarse por las personas involucradas en el negocio. Este glosario debería ir evolucionando a lo largo del proyecto. En un primer momento deberían introducirse los términos que el proyecto utiliza y el significado de los mismos, y a medida que el proyecto avanza ir refinando el glosario.
- Los nombres que se utilizan en la especificación de requisitos deberían ser nombres comunes y habituales para los involucrados en el negocio, es decir, vocabulario que utilicen en su día a día en el trabajo.
- Si dichos nombres “**de negocio**” son erróneos o pueden llevar a confusión, sería interesante sugerir mejores nombres.
- Los nombres “**buenos**” son fácilmente distinguibles ya que invocan el significado correcto por ellos mismos, evitando invertir tiempo en su explicación.
- Crear un **diccionario** escribiendo cada término y su definición.
- Incluir todas las **abreviaturas y acrónimos** utilizados por los usuarios en dicho diccionario.

### 2.3. VERIFICACIÓN DE REQUISITOS

Los requisitos son el medio de comunicación entre el negocio y el área de TI. Debido a su importancia, se podría incluir la verificación de requisitos como una etapa más dentro del ciclo de vida de calidad del producto. En esta fase, el usuario final añade criterios de aceptación para cada requisito. Además, apoya el hecho de que los requisitos han de ser correctos antes de que sean entregados a los diseñadores y desarrolladores.

La **puerta de calidad** es un punto por el que pasan cada uno de los requisitos antes de formar parte de la especificación. Las puertas de calidad normalmente se establecen de tal forma que una o dos personas, probablemente el responsable del análisis de los requisitos y el técnico de pruebas, sean las únicas personas autorizadas para pasar los requisitos a través de las puertas de calidad. Ambas personas se reúnen y validan una serie de características de cada uno de los requisitos tales como: su relevancia, coherencia o facilidad de seguimiento y de pruebas, entre otras cualidades, antes de permitir que pasen a formar parte de la especificación.

Una de las tareas de las puertas de calidad es asegurarse de que cada requisito cumple con el criterio que tiene asignado. Este criterio es una medida del requisito que le hace entendible y con capacidad para ser probado. Con este criterio de aceptación se pretende:

- Validar que los requisitos están completos y son correctos.
- Asociar los requisitos a las funciones de negocio.
- Eliminar ambigüedades.
- Validar que el resultado de cada requisito sea fácil de probar.

- Identificar las funciones de negocio que se cambiaron durante el proyecto.

La característica de poder entender con facilidad los requisitos es siempre en beneficio del cliente, que en muchas ocasiones no está dispuesto a aceptar requisitos que no es capaz de entender o que no contribuyen a su trabajo. El cliente quiere contribuir en el proceso por lo que es necesario poder medir cada requisito.

El analista de los requisitos tiene una razón diferente para medir y probar los requisitos. Necesita asegurarse de que no hay requisitos ambiguos, es decir, han de tener el mismo significado para los clientes y los desarrolladores. También necesitan medirlos para poder decir que se está construyendo el producto que el cliente realmente necesita.

Otra razón por la que el proyecto tiene puertas de calidad es para prevenir posibles **fugas de requisitos**. Los requisitos, algunas veces, aparecen en las especificaciones sin que nadie realmente sepa de donde vienen o qué valor añaden al producto. Asegurándose de que la única forma de que los requisitos entren a formar parte de las especificaciones sea a través de las puertas de calidad, el equipo del proyecto tiene un total control de los requisitos. La función principal de las puertas de calidad es prohibir el paso de requisitos no adecuados a la especificación. Para ello, las puertas hacen seguimientos de cada requisito.

## 2.4. REVISIÓN DE ESPECIFICACIÓN

Las **puertas de calidad**, vistas en el apartado anterior, y **las revisiones** de la especificación **trabajan de forma conjunta**.

Las puertas de calidad comprueban que cada requisito, de forma individual, tiene el estado correcto, no es ambiguo, están dentro del alcance, es probable y trazable. Cuando un requisito pasa por una puerta de calidad, se puede tener confianza acerca de la corrección y viabilidad de los requisitos. *¿Pero qué ocurre con la especificación en conjunto?* En este punto se sabe que los requisitos son correctos, pero *¿se puede asegurar que todos estos requisitos, en conjunto, describen la 'historia' completa?*

El término **especificación de requisitos** hace referencia a la colección de requisitos especificados y definidos previamente. La especificación no tiene que estar en un determinado formato, puede ser una especificación sobre papel, o un blog, o algo similar.

Una vez que la especificación de los requisitos está completa se tendrá un conocimiento preciso del alcance y funcionalidad del producto. Este es el momento de llevar a cabo la revisión de la especificación. En esta revisión final se valida que no falta ningún requisito.

En la primera validación que se lleva a cabo durante la revisión se determina si todos los tipos de requisitos del producto están presentes en la especificación. El objetivo del proyecto normalmente indica los tipos apropiados de requisitos. Por ejemplo, si se está desarrollando un producto financiero pero no hay requisitos de seguridad, está claro que algo falta. Al igual que si se trata de un producto web, la falta de requisitos de usabilidad o de 'look and feel' puede provocar graves problemas.

Al llevar a cabo esta revisión, podría ser útil hacerse preguntas como las siguientes:

- ¿Los usuarios estarán satisfechos con lo que el producto hace de tal forma que satisface lo que necesitan para su trabajo?
- ¿Se han generado suficientes excepciones y escenarios alternativos que cubran estas eventualidades?
- ¿Se han definido los suficientes requisitos no funcionales necesarios para cada caso de uso?

Otro punto muy importante a tener en cuenta es asegurarse de que los requisitos tengan **consistencia**, y en caso contrario, que cualquier conflicto entre los requisitos ha sido resuelto. Dos requisitos **están en conflicto** si no pueden implementarse juntos, es decir, si la solución a un requisito impide la implementación de otro.

Existen distintas **técnicas** para encontrar conflictos entre requisitos. Una técnica bastante utilizada es el uso de matrices. Para rellenar estas matrices se deberían ordenar los requisitos por tipos. Después, examinar todas las entradas que se tienen por cada tipo, observando las parejas de requisitos cuyos criterios estén en conflicto.

		Requisitos			
		1	2	3	4
Requisitos	4				
	3	X			
	2				
	1			X	

*Tabla 4 Matriz conflicto entre requisitos*

Esta matriz identifica los requisitos que están en conflicto. En el caso del ejemplo anterior, los requisitos 1 y 3 están en conflicto. Es decir, que si implementamos una solución para el requisito 1, tendrá efectos negativos sobre la disponibilidad de implementación de la solución para el requisito 3 y viceversa. Las **hojas de cálculo** es una herramienta muy útil cuando se está llevando este tipo de validaciones.

El jefe de proyecto es la persona responsable de asegurarse que no existen inconsistencias entre los requisitos. Otra de sus funciones es:

- asegurarse de la ausencia de **fugas de los requisitos**
- estimar el coste de la construcción del producto
- evaluar los riesgos asociados a los requisitos
- identificar áreas de mejora

## Buenas prácticas

Una vez finalizada la especificación de requisitos se recomienda realizar una serie de **entrevistas** a los agentes del negocio y sesiones de grupo con los desarrolladores para descubrir si el proceso hasta el momento ha sido bueno o malo y sugerir una acción que lo remedie. La intención es hablar con toda la gente que está involucrada en el proceso y hacerles una serie de cuestiones tales como:

- ¿Qué hicimos bien?
- ¿Qué hicimos mal?
- Si lo tuviéramos que hacer otra vez, ¿qué haríamos de forma diferente?

**Revisar y apuntar** las respuestas a este tipo de preguntas ayudará a mejorar el proceso de tal forma que en el siguiente proyecto se pueden usar estos apuntes como punto de partida. Este proceso puede ser algo informal, una reunión con el grupo del proyecto o responsables del proyecto donde se recopilen mensajes de los distintos participantes. Si la participación es muy grande se podría formalizar de tal forma que una persona ajena al proyecto se comunique con los distintos participantes, tanto de forma individual como en grupo, y publique un informe. Las compañías que regularmente realizan esta técnica de forma consistente consiguen significantes mejoras en sus procesos.

### 2.4.1. Priorización

Cuando las expectativas del cliente son altas, la fecha de entrega y los recursos son limitados, es importante asegurarse de que las funciones más importantes del producto sean entregadas y además tan pronto como sea posible. Otro problema que se puede plantear es que haya demasiados requisitos.

La solución a los problemas anteriores es la priorización de los requisitos. El jefe de proyecto tiene que buscar un equilibrio entre el alcance del proyecto deseado y las restricciones de agenda, presupuesto, recursos y metas de calidad. Una manera de llevar a cabo este balance sería dejar los requisitos de baja prioridad para la última versión del producto, mientras que los requisitos de prioridad alta implementarlos cuanto antes.

Las decisiones que hay que tomar sobre la priorización son complejas porque involucran muchos factores, y estos factores están a menudo en continuo conflicto entre ellos. Además, debido a los diferentes objetivos y metas de las distintas personas involucradas en el negocio, puede ser un gran reto alcanzar un acuerdo acerca de la priorización de los requisitos. Las personas, lógicamente tienen sus propios intereses y necesidades, y no siempre están dispuestos a ceder por el beneficio de otra persona involucrada en el negocio.

Los clientes y los desarrolladores deben proporcionar las entradas para la priorización de los requisitos. Los clientes priorizan los requisitos inicialmente desde la perspectiva de cómo de valiosos son cada uno de los requisitos para ellos. Una vez que el desarrollador especifica el coste, dificultad y riesgos asociados a los requisitos, los clientes deben decidir si son tan esenciales como pensaban. La priorización es un balance entre los beneficios de negocio de cada requisito, su coste y cualquier implicación que pueda tener.

Si por ejemplo el cliente no es capaz de diferenciar sus requisitos y clasificarlos en función de su importancia y urgencia, será el jefe de proyecto el encargado de tomar este tipo de decisiones. En estos casos el cliente puede que no esté de acuerdo con las decisiones tomadas por el jefe de proyecto, por eso es importante que el cliente indique qué requisitos son críticos y cuáles podrían ser aplazados. Deberían establecerse prioridades en el proyecto tan pronto como fuese posible.

#### 2.4.1.1. Técnicas

Para facilitar la priorización de requisitos, éstos se podrían agrupar en categorías. El siguiente paso sería priorizar cada una de estas categorías de requisitos, de tal forma que todos los requisitos pertenecientes a una misma categoría tengan la misma prioridad que la de la categoría a la que pertenecen. Por ejemplo podrían haber 3 escalas de prioridades: “alta”, “media”, “baja”. Si estas escalas son demasiado subjetivas e imprecisas y las personas no llegan a un acuerdo, se podrían identificar etiquetas más definidas como: “comprometido”, “permitido en el tiempo”, “futuras versiones”, o, “esencial”, “condicional”, “opcional”... Se podría crear una tabla como la siguiente que contenga las categorías elegidas por la organización y los aspectos relacionados con cada una de ellas.

Etiquetas	Significados
Alta	Requisito con misión crítica; esencial liberar en la siguiente versión del producto.
Media	Soporta operaciones necesarias para el sistema; necesario eventualmente pero podría esperar hasta una versión posterior.
Baja	Mejora de calidad o funcional que podría ser de ayuda algún día, si los recursos lo permiten

*Tabla 5 Categorizaciones de prioridades de requisitos*

La prioridad es un atributo de cada requisito, y como tal debería incluirse tanto en la especificación de requisitos como en los documentos relacionados con los casos de uso. También sería positivo establecer una convención para la especificación de requisitos. Por ejemplo, definir si la prioridad de un requisito de alto nivel se hereda a sus requisitos subordinados o si por el contrario cada requisito individual debería tener su propia prioridad.

Otra forma de priorizarlos sería en función de los riesgos que tengan asociados los requisitos.

A continuación se enumeran otros factores que pueden ayudar a tomar decisiones acerca de la priorización de los requisitos:

- Coste de la implementación
- Valor para el cliente

- Tiempo necesario para implementar el producto
- Facilidad de implementación tecnológica
- Facilidad de implementación organizacional o de negocio
- Objetivos del negocio
- Restricciones legales

### 2.4.2. Evolución de los requisitos

Una idea comúnmente equivocada es el pensar que se han de especificar todos los requisitos antes de pasar a las siguientes etapas de diseño y construcción. En algunas circunstancias es necesario pero no siempre. Por ejemplo, si el documento de los requisitos es una de las bases del contrato, claramente se necesita tener una especificación completa de los mismos. Una vez especificados, se entregan al diseñador quien añade los requisitos técnicos. En este punto habrá finalizado la especificación final de los requisitos y estará lista para entregarse al desarrollador.

Sin embargo, en ocasiones, se puede empezar a crear el producto antes de que todos los requisitos estén perfectamente especificados. Para ello, los agentes de negocio seleccionan casos de uso de alta prioridad para el negocio. Los analistas de requisitos revisan los requisitos para esos casos de uso, exclusivamente. Es inviable ignorar el resto porque siempre hay una conexión funcional mínima entre ellos. Cuando este primer 'grupo' de requisitos ha sido aprobado, se entrega a los desarrolladores, quienes podrán empezar su trabajo. El objetivo es implementar un pequeño número de casos de uso tan pronto como sea posible para poder detectar posibles errores o deficiencias en los requisitos lo antes posible. De este modo, mientras que los primeros casos de uso están siendo desarrollados y liberados, los analistas están trabajando en los siguientes requisitos. En poco tiempo, se consigue establecer un ritmo de liberación con nuevos casos de uso cada pocas semanas.

En resumen, los requisitos **evolucionan** a la vez que lo hace el producto. El cliente comienza la especificación con vagas ideas; mientras, los analistas y el resto de gente implicada en el proyecto exploran el área de trabajo. Van surgiendo nuevas ideas para el producto y los requisitos, a su vez, se van haciendo más precisos y más fáciles de probar. Estos permanecen tecnológicamente neutros hasta que el diseñador se involucra en el trabajo y añade aquellos requisitos necesarios para hacer que el producto trabaje en el entorno tecnológico.

Esta evolución se ve reflejada en la siguiente figura:





















































- **Llevar a cabo una demostración:** Proporcionar al usuario final una visión general del sistema que al que pertenece el prototipo y la demostración del prototipo.
- **Observar cómo los participantes usan el prototipo:** Observar a los participantes puede cubrir problemas de aprendizaje de los que un usuario final podría no informar en una entrevista. Funciona de forma más efectiva cuando se simulan tareas reales y se involucra a los usuarios finales.

## 7.5. HERRAMIENTAS DE PROTOTIPADO

Una herramienta de prototipado debería:

- Dar soporte de desarrollo basado en componentes, por ejemplo permitir la definición y soporte de objetos de código reutilizables
- Proporcionar un entorno integrado para el desarrollo basado en repositorio que soporta el desarrollo de modelos de análisis y diseño del código del programa actual
- Ser flexible, intuitiva y fácil de usar
- Proporcionar un entorno de programación intuitivo y visual
- Incluir soporte para depuración y pruebas
- Permitir realizar cambios en los prototipos de forma rápida y fácil
- Proporcionar herramientas de diseño de base de datos que soporten el desarrollo interactivo de prototipos de base de datos
- Soportar el desarrollo interactivo para acceso a datos (p.ej.: SQL)
- Soportar el volumen requerido de usuarios y transacciones con tiempos de respuesta razonables
- Concordar con la infraestructura técnica
- Proporcionar soporte para la generación automática para múltiples plataformas hardware, si es necesario

Ejemplos de herramientas de prototipado:

- Sistema de gestión de base de datos relacional (RDBMS)
- Generadores de pantallas
- Generadores de menús
- Programación visual
- Lenguajes de 4ª generación

## 7.6. BENEFICIOS Y DIFICULTADES

Una vez que se ha visto cómo desarrollar un prototipo, a continuación se van a describir cuáles son algunos de los principales beneficios y dificultades en el uso de prototipos.

## Beneficios

El uso de prototipos se debería usar para todos los sistemas interactivos. Hay que considerar que una revisión seria por los usuarios finales puede resultar casi siempre en un cambio. Algunos de los beneficios que se obtienen son:

- Los prototipos fundamentalmente **ayudan en la comunicación** entre el usuario final y las personas encargadas del desarrollo; la mejora en la comunicación resulta en una **exactitud mayor** y en un **descenso de los errores** en las partes posteriores del proyecto, cuando son más caros de resolver. La participación temprana otorga poderes a los usuarios finales y facilita la aceptación y entendimiento del sistema.
- La generación de prototipos es un medio efectivo de comunicación del entendimiento del analista de los requisitos del sistema al usuario final. El usuario final consigue un **mejor entendimiento** del sistema propuesto que el que obtendría de una documentación escrita.
- La generación de prototipos es un medio efectivo de comunicación de los requisitos de sistema y el diseño a los programadores. Es **más fácil** para el programador **construir** un sistema desde un modelo de funcionamiento que desde un documento de diseño.
- Algunas actividades de diseño, como el diseño de la interfaz, se realizan en etapas tempranas del ciclo de vida cuando se desarrolla el prototipo. Los usuarios finales son capaces de probar el diseño y proporcionar sus entradas antes de que el tiempo y el dinero se hayan expandido. Si se usan prototipos horizontales durante el análisis, una primera parte de los componentes externos se definen en ese momento. Si se desarrolla un prototipo vertical, se puede diseñar una parte de la base de datos durante el análisis.
- El uso de prototipos puede **reducir enormemente el alcance y tamaño** de las actividades de diseño. Las actividades de diseño siguen teniendo lugar pero en otras etapas del ciclo de vida del sistema. Dependiendo de la extensión del prototipo, se pueden reducir los esfuerzos de diseño para incluir sólo tareas como preparación del plan de pruebas, diseño de ayudas de usuario, diseño de conversión y optimización de base de datos.
- Los **usuarios** finales se pueden **involucrar más** en el proceso de desarrollo del sistema cuando se usan prototipos. Esto ayuda a asegurar que los requisitos de usuario se cumplen.
- Los prototipos hacen a un **sistema** ser **más intuitivo** y **reduce** la cantidad de **documentación** escrita requerida.
- Los prototipos **animan** y requieren la **participación activa** de los usuarios finales.

## Dificultades

- La estructura de los prototipos se corrompe por cambios constantes. De ahí que sea difícil una **evaluación a largo plazo**.
- Los prototipos **requieren una participación activa** de los usuarios finales.
- Un prototipo **no puede sustituir completamente** una especificación en papel. Los prototipos deberían complementar, no reemplazar, otras metodologías.
- Numerosas cuestiones de diseño no son y no pueden ser documentadas de forma adecuada mediante el uso de prototipos; estas cuestiones se pueden olvidar sin querer.
- Los prototipos con frecuencia conducen a un **acuerdo prematuro del diseño físico**.

## 8. GLOSARIO

---

- **Analista de requisitos:** El analista de requisitos es un rol. Es la persona que se encarga de capturar los requisitos en modelos de caso de uso, casos de uso y especificaciones suplementarias. Facilita la definición del alcance y los detalles de los requisitos, describe las relaciones y dependencias entre requisitos y proporciona un vocabulario común.
- **Atributos de calidad:** Exposiciones que indican cómo de bien realiza el sistema algunos comportamientos. Son un tipo de requisitos no funcionales. Pueden ser características deseables como: rápido, fácil, intuitivo, robusto, fiable, seguro y eficiente. Hay que trabajar con los usuarios para definir de forma precisa que quieren decir con este tipo de términos que suelen ser ambiguos y subjetivos.
- **Brainstorming:** El término ‘brainstorming’ hace referencia a un proceso de pensamiento lateral y es una excelente forma de desarrollar distintas soluciones creativas para un problema. Consisten en centrarse en un problema y presentar todas las soluciones posibles. Se diseña para ayudar a romper con unos patrones de pensamiento y encontrar nuevas formas de mirar a ese problema. Intenta abrir posibilidades y desechar suposiciones erróneas sobre el límite del problema y no contempla críticas a las ideas propuestas. Todas las ideas se evalúan una vez que la sesión de brainstorming ha finalizado y estas soluciones se pueden explorar o analizar usando enfoques convencionales.
- **Capacidad de prueba:** Es un aspecto medible de los requisitos para comprobar si la solución se corresponde con el requisito original.
- **Casos de uso:** Son declaraciones generales de objetivos de usuario o tareas de negocio que son necesarios para realizar el sistema, mientras que las descripciones de tareas específicas representan escenarios de uso.
- **Control de cambios:** El control de cambios es un proceso formal que se utiliza para asegurar que un producto, servicio o proceso sólo se modifica de acuerdo a un cambio necesario identificado.
- **Desarrollo de requisitos:** El desarrollo de requisitos implica entender los requisitos de negocio, obtener los requisitos de usuario y trasladar los requisitos de negocio y de usuario a requisitos software/de sistema.
- **Escenarios:** Un escenario es una breve descripción de un evento. Se usa para comunicar una idea de un producto o experiencia que implica interactividad. Un escenario es también un informe o una sinopsis de una acción, evento o situación en curso. El desarrollo de escenarios se usa en la planificación y obtención de requisitos.
- **Gestión de cambios:** La gestión de cambios es el proceso de desarrollar un proceso de petición de cambio planificado en una organización. Típicamente el objetivo es

maximizar los esfuerzos colectivos de toda la gente implicada en el cambio y minimizar el riesgo de fallo a la hora de implementar el cambio.

- **Gestión de requisitos:** La gestión de requisitos implica gestionar los cambios de los requisitos y mantener la consistencia entre los requisitos y otros productos de trabajo del proyecto. Es el proceso de encontrar, documentar, organizar y hacer un seguimiento de los cambios de los requisitos de un sistema.
- **Línea base:** una instantánea o un estado específico de un conjunto de productos de trabajo que han sido formalmente revisados y aprobados. Aunque el estado se actualice más adelante, siempre a través de un procedimiento de control de cambios, la línea base permanece constante y disponible como referencia del estado original para poder compararlo con el estado actual.
- **Obtención de requisitos:** El proceso de identificar las necesidades y salvar las disparidades entre las comunidades involucradas en el propósito de definir y destilar los requisitos para cumplir con las restricciones de estas comunidades.
- **Priorización de requisitos:** La priorización de requisitos es el proceso de asignar prioridades a los requisitos, basándose en sus beneficios esperados y en la importancia que tienen para la gente implicada en el proyecto, de tal forma que los que tengan prioridades mayores puedan implementarse primero, como parte de un ciclo de desarrollo incremental, iterativo...
- **Prototipos:** Son simulaciones de una aplicación que permite a los usuarios visualizar la aplicación que no está aun construida. Los prototipos ayudan a los usuarios a tener una idea de cómo va a ser el sistema y facilita la toma de decisiones relacionadas con el diseño sin esperar a que éste se haya construido. Los prototipos proporcionan a los desarrolladores de software un modelo de trabajo y pueden usarse por los clientes, analistas de negocio o gerentes para confirmar o realizar cambios en los requisitos, ayudar a definir interfaces, desarrollar componentes de colaboración y proporcionar unos mejores acuerdos.
- **Puerta de calidad:** es un punto por el que pasa cada uno de los requisitos antes de formar parte de la especificación de requisitos. Las puertas de calidad se aseguran de que cada requisito cumple con el criterio que tiene asignado.
- **Requisito:** Un requisito es una condición o capacidad con la que ha de cumplir el sistema. Es decir, algo que el producto debe hacer, o una propiedad que el producto debe tener, y que es necesaria para las personas involucradas en el negocio. Hay varios tipos de requisitos como pueden ser funcionales, de usabilidad, de fiabilidad, de rendimiento, de mantenimiento...
- **Requisitos de fiabilidad:** Requisitos que describen la fiabilidad y robustez del sistema.
- **Requisitos de negocio:** una descripción a alto nivel de lo que el sistema debe hacer. Representan los objetivos, la base del negocio, estrategias, visión, alcance y el valor esperado del desarrollo del software. Los requisitos de negocio proporcionan

la dirección que ha de seguir el proyecto y forman la base de los requisitos de usuario.

- **Requisitos de rendimiento:** Incluyen tiempos de respuesta, requisitos de precisión, de capacidad, de escalabilidad, de seguridad...Especifican quién tiene acceso autorizado al producto, y bajo qué circunstancias se concede ese acceso y a qué partes del sistema/aplicación.
- **Requisitos de sistema:** contienen los requisitos funcionales y no funcionales del sistema a un alto nivel de arquitectura. Definen las funcionalidades y características que hay que construir en el sistema/software para satisfacer los requisitos de negocio y de usuario. Esto sirve como fuente para una arquitectura, diseño y planes de pruebas detallados.
- **Requisitos de usabilidad:** Incluyen facilidad de uso, personalización e internacionalización, requisitos de accesibilidad, requisitos de usuario... Proporcionan más detalle de los requisitos de negocio; son la descripción de las tareas para que sean ejecutadas de forma correcta por el software cuando se trabaja con él. Estos requisitos describen la funcionalidad necesaria para satisfacer tareas específicas, necesidades operativas y grupos de usuario.
- **Requisitos de usuario:** entran en más detalle en los requisitos de negocio. Son una descripción de las tareas que ha de ejecutar de forma adecuada el software cuando el usuario opera con él. Estos requisitos describen la funcionalidad necesaria para satisfacer tareas específicas, necesidades operacionales y grupos de usuarios.
- **Requisitos del cliente:** Definen e identifican los requisitos del cliente, para incluir la voz del cliente, los datos, las expectativas convertidas en expresiones medibles que se usan para asegurar que se cumple con las necesidades del cliente.
- **Requisitos funcionales:** Los requisitos funcionales describen el comportamiento observable que el sistema exhibirá, con frecuencia en el contexto de una secuencia de acción del actor – repuesta del sistema. Los requisitos funcionales definen lo que el sistema hará; forman la mayor parte de la especificación de requisitos.
- **Requisitos no Funcionales:** Los requisitos no funcionales son requisitos que especifican criterios que pueden ser usados para describir la operación de un sistema más que el comportamiento específico. Los requisitos no funcionales son aquellos que describen las características globales. Los requisitos no funcionales típicos son la fiabilidad, escalabilidad, etc.
- **Requisitos tecnológicos:** un requisito que es necesario sólo debido a la tecnología elegida. Su objetivo no es satisfacer una necesidad directa del cliente.
- **Restricciones:** Las restricciones son condiciones que limitan las elecciones disponibles al diseñador o programador. Son requisitos globales. Pueden ser restricciones del propio proyecto o del diseño del producto.
- **Trazabilidad:** La trazabilidad es la capacidad de enlazar un elemento del proyecto con otro elemento del proyecto relacionado, especialmente los relacionados con los requisitos. Los elementos del proyecto involucrados en la trazabilidad se denominan



Instituto Nacional  
de Tecnologías  
de la Comunicación

elementos de trazabilidad. Típicamente los elementos de trazabilidad incluyen diferentes tipos de requisitos, elementos de análisis y diseño, artefactos de prueba (baterías de pruebas, casos de pruebas, etc.), así como documentación de soporte y material de formación.

## 9. REFERENCIAS

---

A. Abran, J.W. Moore, P. Bourque, R. Dupuis, *Guide to the Software Engineering Body of Knowledge*, IEEE Computer Society, 2004.

Gilb, Tom. *Competitive Engineering: A Handbook for Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage*. Butterworth-Heinemann, 2005

K.E. Emam, J.N. Drouin, W. Melo, *SPICE: The Theory and Practice of Software Process Improvement and Capability Determination*, IEEE Computer Society Press, 1998. Leffingwell, Dean, and Don Widrig. *Managing Software Requirements: A Use Case Approach*. Second edition. Addison-Wesley, 2003

M.B. Chrissis, M. Konrad, S. Shrum, *CMMI® Second Edition. Guidelines for Process Integration and Product Improvement*, Addison-Wesley, 2007.

R.S. Pressman, *Software Engineering: A Practitioner's Approach*, Sixth ed., McGraw-Hill, 2004.

Sommerville, Ian and Pete Sawyer. *Requirements Engineering: A Good Practice Guide*. John Wiley & Sons, 1998

Suzanne Robertson y James Robertson, *"Mastering the Requirements Process"*, Segunda edición (2006)