**Gestión de requerimientos.**

La gestión de requerimientos establece lo que el sistema debe hacer en cuanto a procesos, consultas, reportes, alarmas, interfaces, restricciones de seguridad y algunos otros elementos que la organización necesite, por lo que si no se identifican de manera correcta, el software no proporcionará al usuario la funcionalidad esperada; además, si no se determinan de manera completa y clara, no se conocerá el alcance ni será posible estimar la dimensión real del proyecto.

**Gestión de requerimientos**

Es la disciplina relacionada con el análisis, documentación y especificación de requerimientos. También proporciona los mecanismos adecuados para facilitar las actividades de análisis, documentación y verificación de éstos. Engloba todas las actividades relacionadas con descubrir, documentar y mantener un conjunto de requerimientos de un proyecto de [**software empresarial**](https://www.evaluandoerp.com/software-erp/conceptos-erp/que-es-un-erp/).

**Ingeniería de requerimientos**

El término ingeniería, implica el uso sistemático y repetible de técnicas que son usadas para asegurar que los requerimientos del sistema sean completos, consistentes y relevantes.

La Ingeniería de Requerimientos tiene como objeto principal dar, tanto al cliente como al proveedor (desarrollador), un mecanismo de aseguramiento para ambas partes, donde estén de acuerdo con el alcance funcional y técnico.

Comprende la transformación de una necesidad operacional en una descripción del sistema, los parámetros de desempeño del sistema y la configuración del sistema mediante el uso de un proceso iterativo de análisis, diseño, y prototipos.

Debe tener un complejidad que sea claro tanto para el cliente como para el proveedor (desarrollador), si existiera algo técnico, se debe incluir también una explicación para que el cliente no tenga dudas, ya que debe estar firmado por ambas partes.

**Requerimientos de un proyecto de software**

Si bien existen diferencias entre Ingeniería de Requerimientos de Sistema y la Ingeniería de Requerimientos de Software, solo mencionaremos la más importante para satisfacer a los puristas. Mientras que el origen de los requerimientos de sistema radica en las necesidades del usuario, el origen de la gestión de requerimientos de un proyecto de software se origina en los requerimientos y/o especificaciones del sistema.

El cliente es el experto en los [**procesos de su negocio**](https://www.evaluandosoftware.com/nivel-de-madurez-de-los-procesos/), por tanto sabe cuáles son sus necesidades de información; por otro lado, el proveedor es el experto en los temas de características de los sistemas y especificaciones del software.

**El origen del problema**

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

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

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

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

Jerarquía de requerimientos

**¿Por qué definir los requerimientos?**

* Una de las razones más comunes por las que un sistema falla es una mala definición de los requerimientos.
* El escribir especificaciones de sistema y de software buenas, correctas, completas y medibles es uno de los problemas más comunes en el desarrollo de software.

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

Los usuarios, por lo general saben lo que requieren, pero en muchas ocasiones no saben cómo solicitarlo, mucho menos cómo documentarlo. Para este proceso es necesario que, el usuario cuente con su proceso documentado a nivel manual y a partir de este se lleve a cabo la documentación del requerimiento.

Si bien uno de los puntos principales de los problemas presentados en una [**implementación**](https://www.evaluandosoftware.com/la-capacitacion-en-el-proceso-de-implementacion-de-un-sistema/) es la mala o nula documentación de la gestión de requerimientos, es algo que ni el cliente ni el proveedor del software se sientan a detallar adecuadamente.

Por un lado, el cliente en ocasiones no exige esto con el fin de ahorrarse tiempo o considerar que sobre la marcha se harán los ajustes necesarios. Por otra parte, el proveedor debería exigir este análisis pero se elevarían los costos del proyecto y seguramente no se concluiría la venta.

Se debe buscar un punto intermedio donde por lo menos se asegure el alcance.

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

**La principal relación de la gestión de requerimientos de un proyecto de software con el retraso de los proyectos, radica en que algunos aspectos están fuera del alcance original, el cual nunca fue documentado, por lo que se debe implementar un mecanismo formal para justificarlo**. Para que no quede lugar a malos entendidos entre los clientes y el equipo de desarrollo.

En caso de existir la necesidad de un ajuste al alcance del proyecto, lo más conveniente es que se maneje con un documento llamado “control de cambios” de manera que se documente el tiempo que pudiera retrasar el proyecto y también si va a tener algún costo.

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

Cuando el proyecto es muy grande, es cuando más se debe invertir tiempo en el establecimiento del alcance, entre más detallado menor será el riesgo de un fracaso.

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

Lo más conveniente es tener una junta o reunión de evaluación para determinar si los cambios que se requieren pueden esperar para una segunda fase o son necesarios para la puesta en marcha.

**Clasificación de los requerimientos**

**De acuerdo a la funcionalidad**

**Requerimientos funcionales**

Describen lo que el sistema debe hacer, es decir, especifican acciones que el sistema debe ser capaz de realizar, sin considerar restricciones físicas. Los requerimientos funcionales especifican el comportamiento del sistema.

**Requerimientos no funcionales**

Describen únicamente atributos del sistema o atributos del ambiente del sistema y pueden ser por ejemplo: la gestión de requerimientos de interfaz, de diseño, de [**implementación**](https://www.evaluandoerp.com/software-erp/implementar-erp/), legales, físicos, de costo, de tiempo, de calidad, de seguridad, de construcción, de operación, entre otros.

**De acuerdo al nivel de cumplimiento**

**Requerimiento obligatorio**

Es un requerimiento que debe ser implementado.

**Requerimiento recomendable**

Es deseable que sea implementado.

**Requerimiento opcional**

No es crítica su implementación.

**Responsabilidades de los usuarios**

Dentro de las responsabilidades de los usuarios están, instruir a los analistas sobre el negocio y definir el vocabulario de su área, invertir tiempo en proporcionar requerimientos y aclarar dudas, también debe ser específico y preciso sobre las necesidades del negocio y los requerimientos de un proyecto de software.

Otros factores que también se deben considerar son tomar decisiones a tiempo sobre los requerimientos, respetando las estimaciones de costo y viabilidad del equipo de desarrollo, fijando prioridades para requerimientos individuales, características del sistema o casos de uso.

# **Cómo redactar un documento de requisitos de software**

Los documentos de especificación de requisitos de software pueden ser útiles para que los gerentes de proyectos o de producto y los analistas de negocios desglosen los conceptos de alto nivel y los transformen en acciones concretas que cualquier miembro del equipo pueda seguir durante el proceso de desarrollo.

**¿Qué es un documento de especificación de requisitos de software?**

En un documento de especificación de requisitos de software (SRS, por sus siglas en inglés, o ERS, por sus siglas en español) se enumeran los requisitos, las expectativas, el diseño y los estándares de un proyecto futuro. Incluye los requerimientos comerciales generales que rigen al objetivo del proyecto, los requisitos y necesidades de los usuario finales, y la funcionalidad del producto en términos técnicos. Para hacerlo más simple, una especificación de requisitos de software ofrece una descripción detallada de cómo debería funcionar un producto de software y qué debería hacer tu equipo de desarrollo para hacerlo funcionar.



Imagina que se te ha ocurrido una idea genial para una aplicación. Has visualizado lo que quieres hacer y qué aspecto deseas que tenga, pero sabes que no bastará con describírselo verbalmente a un desarrollador y esperar que cumpla con las expectativas. Aquí es donde la especificación de requisitos de software cobra importancia.

**¿Por qué conviene usar una especificación de requisitos de software?**

Si los desarrolladores no tienen instrucciones claras con respecto a cuándo crear un [producto nuevo](https://asana.com/es/resources/product-development-process), probablemente termines dedicando más tiempo y dinero del previsto, solamente, para producir un software que coincida con lo que tienes en mente.

La composición de este documento sirve para poner las ideas en papel y organizar una lista clara de los requerimientos. El documento se transforma en la fuente única de referencias, para que todos los equipos, desde marketing a mantenimiento, estén en la misma sintonía.

Dado que la especificación de requisitos de software es un documento dinámico, también puede funcionar como centro para las comunicaciones entre las diferentes [partes interesadas](https://asana.com/es/resources/stakeholder-vs-shareholder) de un proceso de desarrollo de productos. Indefectiblemente, en todo proyecto de desarrollo de software habrá iteraciones del producto. Si anotas los cambios en la especificación de requisitos de software, todas las partes podrán validarlos en el documento. De este modo, no habrá confusiones con respecto a cuáles son los requisitos.

Qué incluir en una especificación de requisitos de software

La estructura de una especificación de requisitos de software básica cuenta con cuatro partes: una introducción, los requisitos funcionales y los del sistema, los de la interfaz externa y los no funcionales.



1. Introducción

La introducción en esta especificación es exactamente lo que esperas, un panorama global del proyecto en general. Cuando redactes la introducción, describe el propósito de la creación del producto, el público que pretendes cautivar y cómo crees que ese público usará el software. En la introducción no olvides incluir lo siguiente:

* El alcance del producto: El [alcance](https://asana.com/es/resources/scope-management-plan) debería estar vinculado a los objetivos generales y comerciales del producto, un dato que resultará particularmente importante en caso de que varios equipos o contratistas tengan acceso al documento. Enumera los beneficios, las metas y los objetivos previstos para el producto.
* El valor del producto: ¿Por qué tu producto es importante? ¿Qué beneficio le aportará a la audiencia prevista? ¿Qué función tendrá o qué problema resolverá? Pregúntate qué valor encontrará el público en tu producto.
* El público objetivo: Describe a tu público ideal. Serán ellos quienes dicten el aspecto que tendrá el producto y cómo deberás comercializarlo.
* El uso previsto: Imagina de qué manera tu público usará el producto. Enumera las funciones que brindarás y todas las formas posibles en que la audiencia podrá usar el producto dependiendo del rol. También resultará una buena práctica incluir casos de uso para ilustrar tu visión.
* Las definiciones y los acrónimos: Cada sector o negocio tiene sus propios acrónimos y su jerga particular. Dispón las definiciones de los términos que usarás en el documento para asegurarte de que todos entiendan lo que quieres decir.
* El índice: Si el documento es exhaustivo, es probable que sea bastante largo. Incluye un índice para ayudar a que los participantes encuentren exactamente lo que buscan.

Verifica que tu introducción sea clara y concisa. Recuerda que será la guía para el resto de la estructura de la especificación de requisitos de software, y te convendrá que todos interpreten lo mismo cuando usen el documento.

2. Requisitos funcionales y del sistema

Una vez que hayas terminado la introducción, será hora de volverse más específico. En los requisitos funcionales se desglosan las características y funciones que permiten que el sistema se comporte como ha sido previsto.

Usa el resumen general como referencia para controlar que los requisitos cumplan con las necesidades básicas de los usuarios a medida que completas los datos. Dependiendo del producto, puede haber cientos de requisitos funcionales por incluir. Algunos de los más comunes son los siguientes:

* Funciones “If/then” (si/o)
* Lógica de manejo de datos
* Flujos de trabajo del sistema
* Gestión de las transacciones
* Funciones administrativas
* Deberes regulatorios y de cumplimiento de normas
* Requisitos de desempeño
* Detalles sobre las operaciones llevadas a cabo en cada pantalla

Si parece mucho, intenta con un requisito a la vez. Mientras más detalles incluyas en la especificación de requisitos de software, menos problemas deberás detectar y resolver más adelante.

3. Requisitos de la interfaz externa

Los requisitos de la interfaz externa son tipos de requisitos funcionales con los que se garantiza que el sistema se comunicará correctamente con los componentes externos como:

* Las interfaces de usuarios: Son la clave de la facilidad de uso de una aplicación. Incluyen la presentación del contenido, las opciones de navegación de la aplicación y asistencia para los usuarios, entre otros componentes.
* Las interfaces de hardware: Son las características de cada interfaz que interactúa entre los componentes del software y del hardware del sistema; como los tipos de dispositivos compatibles y los protocolos de comunicación.
* Las interfaces de software: Incluyen las conexiones entre tu producto y otros componentes de software, como las bases de datos, las bibliotecas y los sistemas operativos.
* Las interfaces para comunicaciones: Los requerimientos de las funciones que se usarán en tu producto para comunicación, como los emails o los formularios integrados.

Los sistemas embebidos dependen de los requisitos de las interfaces externas. Deberías incluir cosas como los diseños de pantalla, las funciones de los botones y una descripción de en qué medida tu producto dependerá de otros sistemas.

4. Requisitos no funcionales (NRF)

La sección final de los detalles de la especificación de requisitos de software incluye los requisitos no funcionales. Mientras que con los requisitos funcionales se le indica al sistema cómo debe comportarse, con los no funcionales (NFR) se determina de qué manera el sistema implementará estas funciones. Por ejemplo, con un requisito funcional se puede determinar que el sistema imprimirá una etiqueta de empaque cada vez que un cliente pida un producto. Con una NFR se garantizará que las impresiones de las etiquetas se harán en papel blanco de 4” × 6”, el tamaño estándar para este tipo de etiquetas.

Si bien es cierto que el sistema podría funcionar incluso aunque no cumpliera con los NFR, estarías poniendo en riesgo la participación de los usuarios o las partes interesadas. Este tipo de requerimientos sirven para controlar los requisitos funcionales, de modo que se conserven los atributos como la asequibilidad del producto o su facilidad de uso.

Estos son los tipos más comunes de NFR:

* La seguridad: Qué se necesita para garantizar que cualquier información sensible de los usuarios que recopile tu software estará protegida.
* La capacidad: Lo que se necesita para satisfacer la demanda de almacenamiento ahora y lo que se necesitará más adelante. Incluye un plan para determinar cómo se adaptará el sistema a escala a medida que aumente la demanda.
* La compatibilidad: Los requisitos mínimos de hardware para el software, como la compatibilidad con los sistemas operativos y sus versiones.
* La fiabilidad y disponibilidad: Con qué frecuencia esperas que los usuarios utilicen el software y cuál es el tiempo de fallo crítico esperado con relación a un uso normal.
* La escalabilidad: La cantidad máxima de trabajo con la que el sistema se comportará como se ha previsto.
* La mantenibilidad: Cómo se debería usar la integración continua en tu aplicación para implementar funciones o reparar errores rápidamente.
* La facilidad de uso: Cuán fácil resulta usar el producto.

Entre los tipos más comunes de requisitos no funcionales se encuentran los de desempeño, los regulatorios y los requisitos medioambientales.



# Requerimientos de software

Hablemos de una parte muy importante no solo en Software Testing, sino, en Ingeniería de Software en general: Los requerimientos o requisitos.



Un requerimiento define las funciones, capacidades o atributos intrínsecos de un sistema de software, en otras palabras, describe como un sistema debe comportarse.

Existen dos tipos de requerimientos:

### ****Requerimientos funcionales****

### 

Según la oficina de gestión de proyectos, los requerimientos funcionales de un sistema, son aquellos que describen cualquier actividad que este deba realizar, en otras palabras, el comportamiento o función particular de un sistema o software cuando se cumplen ciertas condiciones. Estos requisitos deben estar especificados de manera explícita. Ejemplos:

* El campo de monto acepta únicamente valores numéricos con dos decimales.
* La pantalla de registro de pago puede imprimir los datos en pantalla a la impresora.
* Después de tres intentos fallidos de entrar al sistema, bloquear la cuenta.

### ****Requerimientos no funcionales****

Los requerimientos no funcionales definen las características o cualidades generales que se esperan de un sistema y establecen restricciones sobre el producto, el proceso de desarrollo de software y establecen restricciones externas que el software debe lograr. Ejemplos:

* **Requerimientos de usabilidad:** La usabilidad se define como el esfuerzo que necesita hacer un usuario para aprender, usar, ingresar datos e interpretar los resultados obtenidos de un software de aplicación.
* **Requerimientos de eficiencia:** Relacionado con desempeño en cuanto a tiempo de respuesta, número de operaciones por segundo, entre otras mediciones, así como consumo de recursos de memoria, procesador, espacio en disco o red.
* **Requerimientos de dependibilidad:** Engloba varios atributos como:
* **Disponibilidad:** Disposición del sistema para prestar servicio correctamente.
* **Confiabilidad:** Continuidad del servicio prestado por el sistema.
* **Integridad**: Ausencia de alteraciones inadecuadas al sistema.
* **Mantenibilidad:** Posibilidad de realizar modificaciones o reparaciones a un proceso sin afectar la continuidad del servicio.

Ahora que tenemos estos conceptos claros ¿Cómo relacionamos los requerimientos con el testing?

Las pruebas se pueden clasificar conforme a lo que verifican, en este caso se dividen en pruebas funcionales y no funcionales.

Las [**pruebas funcionales**](https://www.diariodeqa.com/post/tipos-de-pruebas-funcionales) se basan en la ejecución, revisión y retroalimentación de las funcionalidades anteriormente especificadas, es decir, los requisitos funcionales. Dentro de estas hay distintos tipos que serán detalladas en el siguiente post. Algunos tipos de pruebas funcionales son:

* Pruebas exploratorias
* Pruebas de humo
* Pruebas de regresión

Las [**pruebas no funcionales**](https://www.diariodeqa.com/post/pruebas-no-funcionales), en cambio, tienen como objetivo verificar los requisitos que se utilizan para juzgar la operación de un sistema, en palabras más simples, características que, aunque tal vez no fueron especificadas, un sistema debe contar con ellas. Ejemplo:

* Pruebas de usabilidad
* Pruebas de rendimiento
* Pruebas de compatibilidad
* Pruebas de portabilidad

En conclusión, para decir que sistema tiene calidad deben cumplirse los requisitos funcionales y no funcionales, aunque cabe destacar que esto no es lo único que se toma en cuenta para evaluar la calidad. Si un sistema cumple con los requisitos especificados y no cumple con las necesidades del cliente, pues, no tiene calidad, porque el cliente no lo utilizará.