Material descargable

Testing Manual

Introducción

Hasta este momento hemos conceptualizado los requerimientos como una categoría amplia que engloba "todo lo que puede solicitarse al iniciar un proyecto de software". Ahora, estamos en posición de ser más precisos al definir el requisito de software como una descripción minuciosa del sistema en implementación. Estos requerimientos delinean el uso práctico del producto o servicio, especificando las condiciones o capacidades a las que debe ajustarse el sistema. Pueden variar desde declaraciones abstractas de alto nivel de servicios o restricciones del sistema hasta especificaciones funcionales matemáticas detalladas.

¿Qué implica el análisis de requisitos? Este proceso determina las expectativas del usuario para un sistema en consideración. **El resultado de este análisis debe ser cuantificable y detallado**. Sirve como fundamento para los planes de prueba y el plan del proyecto, constituyendo un acuerdo sólido entre el desarrollador y el cliente. Además, el análisis de requisitos es un procedimiento que permite aclarar tanto los requisitos declarados como aquellos no expresamente mencionados, contribuyendo a validar la integridad, eliminando ambigüedades y asegurando la viabilidad del requisito.

"El impacto de un error en el análisis de requerimientos". Fuente: Foundations of software testing de Dorothy Graham, Rex Black, Erik Van Veenendaal e Isabel Evans.

En la imagen se muestra la consecuencia de un análisis deficiente de los requisitos y su impacto en el ciclo de vida del desarrollo del software.

Desafíos en la fase de análisis de requisitos en el control de calidad

A continuación, exponemos algunos de los desafíos más comunes en la fase de análisis de requisitos. Aunque en teoría lograr un alto nivel de comunicación entre los equipos parece sencillo, esta etapa del proceso suele ser la más propensa a desafíos.

- **Alcance Indefinido en la Etapa Inicial del SDLC:** En la fase inicial del Ciclo de Vida del Desarrollo de Software (SDLC), el alcance a menudo no está claramente definido, lo que puede generar confusiones y malentendidos.
- **Comprensión Ambigua de Procesos:** La comprensión ambigua de los procesos puede complicar la identificación precisa de los requisitos, ya que las interpretaciones pueden variar entre los miembros del equipo.
- **Importancia de la Comunicación entre Equipos y Partes Interesadas:** La calidad de la comunicación entre el equipo del proyecto y las partes interesadas desempeña un papel crucial en la definición precisa de los requisitos.
- **Entradas Insuficientes del Cliente y Suposiciones en UAT:** La falta de información clara por parte del cliente puede llevar a suposiciones, y esto se evidencia en la fase de Pruebas de Aceptación del Usuario (UAT).
- **Inconsistencia Dentro de un Proceso y entre Usuarios:** La inconsistencia dentro de un mismo proceso, especialmente cuando involucra a múltiples usuarios, puede generar conflictos y dificultar la definición de requisitos uniformes.
- **Puntos de Vista Conflictivos de los Clientes:** Los diferentes puntos de vista entre los clientes pueden complicar la consolidación de requisitos coherentes y consensuados.
- **Frecuencia de Nuevos Requisitos:** La introducción frecuente de nuevos requisitos a lo largo del proceso puede desafiar la estabilidad y planificación del proyecto.

Los requisitos rara vez se presentan como una lista completamente ordenada desde el inicio. La realidad profesional a menudo implica un período de tiempo entre la presentación inicial de los requisitos por parte del cliente y la obtención de una lista más definida y estable, que se pueda trabajar sin cambios significativos.

Herramientas y técnicas utilizadas para el análisis de los requisitos

Para mitigar el efecto de un mal análisis de requerimientos, existen herramientas y técnicas que podemos aplicar en este momento. Te presentamos dos:

1. **Casos de Uso:** Es una metodología utilizada en el análisis de requisitos para identificar, aclarar y organizar los requisitos. Es un conjunto de posibles secuencias de interacciones entre sistemas y usuarios en un entorno particular y relacionado con un objetivo particular.
2. **Documento de comprensión de los requisitos[1] (RUD):** el documento cubre los detalles de la comprensión de los requisitos relacionados con los puntos siguientes:
    - Suposiciones
    - Detalles del sistema
    - Requisitos del sistema lógico
    - Entidad del sistema
    - Hardware
    - Criterios de aceptación
    - Priorizar cada requisito
    - Discutir con el equipo e identificar el alcance de la prueba
    - Desglose los requisitos en tareas e historias de usuario.

¿Cómo analizar los requisitos?

- **Definir el Propósito del Software:** Comienza por comprender claramente el propósito del software que se debe desarrollar. Entender la finalidad del sistema es esencial para orientar todo el proceso de análisis.
- **Interrogar con Preguntas Clave:** Identifica los requisitos mediante preguntas fundamentales como "¿por qué?", "¿qué?", "¿quién?", "¿cómo?", etc. Estas preguntas ayudarán a desglosar los aspectos cruciales y asegurar una comprensión exhaustiva.
- **Evaluar Complejidad e Impacto:** Analiza la complejidad potencial de la aplicación y su impacto en las pruebas. Esto implica considerar la interrelación de los requisitos y cómo su implementación afectará la funcionalidad general del software.

1

- **Incorporar una Perspectiva de Pruebas:** Desde el inicio, ten en cuenta que todos los aspectos del software deberán ser sometidos a pruebas. Esto garantiza que los requisitos sean no sólo comprensibles desde una perspectiva de desarrollo, sino también evaluados desde el punto de vista de la calidad y la robustez.

Validación de requisitos

Durante la validación de requisitos, es crucial asegurar que al final de la fase de análisis de requisitos se disponga de toda la información necesaria. Para lograr esto, se deben considerar los siguientes puntos:

- **Corrección**: Realiza una exhaustiva revisión para identificar declaraciones o requisitos incorrectos. Corregir cualquier inexactitud desde el principio es esencial para evitar malentendidos y garantizar la precisión del conjunto de requisitos.
- **Completitud:**Enfrenta el desafío de buscar y encontrar cualquier requisito que pueda estar ausente. La completitud es esencial para asegurar que todos los aspectos cruciales estén abordados y no se omita información esencial.
- **Factibilidad**:Evalúa cuidadosamente la factibilidad de cada requisito. Identifica las características que son prácticas y posibles de probar, diferenciándolas de aquellas que podrían estar más allá del alcance. Esto proporciona una visión realista de lo que se puede lograr.
- **Testeabilidad**: Verifica la posibilidad de crear diversas pruebas aplicables a los requisitos. Una buena testeabilidad garantiza que los requisitos sean verificables y medibles, facilitando la validación y las pruebas posteriores.
- **No Ambigüedad:** Busca una interpretación única y clara de cada requisito. La ausencia de ambigüedad asegura que la comprensión de los requisitos sea uniforme entre todos los miembros del equipo y partes interesadas, evitando posibles malentendidos.

Actividades realizadas para el análisis de requisitos

Hay tres actividades principales que realiza el equipo de control de calidad en esta fase. Las actividades se describen a continuación.

1. **DEFINICIÓN DEL ALCANCE**

En el proceso de definición del alcance, el equipo de control de calidad desglosa y delimita las pruebas a niveles superiores, dividiéndolas en varios módulos funcionales. Además, se identifican los distintos tipos de pruebas necesarias, como pruebas de humo, pruebas de cordura, pruebas funcionales y pruebas de regresión, entre otras.

Para realizar esta tarea de manera eficaz, el equipo de control de calidad examina los requisitos previos y los detalles del entorno en el cual se llevarán a cabo las pruebas. Se recopilan detalles sobre las prioridades de las pruebas y se pone un enfoque específico en la secuencia de módulos que se validarán.

Esta fase de definición del alcance no solo establece los límites de las pruebas, sino que también sienta las bases para una planificación detallada y una ejecución efectiva de los diversos tipos de pruebas que asegurarán la calidad del producto final.

2. **RTM (MATRIZ DE TRAZABILIDAD)**

El proceso de seguimiento de requisitos es esencial para documentar las conexiones entre los requisitos y los productos de trabajo asociados con su implementación y verificación. La Matriz de Trazabilidad de Requerimientos (RTM) se erige como un elemento clave en este proceso, capturando todos los requisitos desde el Análisis de Requisitos junto con su trazabilidad en un único documento entregado al final del ciclo de vida.

La RTM se crea al inicio del proyecto, constituyendo la piedra angular del alcance y los entregables. Su naturaleza bidireccional rastrea los requisitos hacia adelante al analizar las salidas de los entregables y hacia atrás al revisar los requisitos comerciales vinculados a características específicas del producto.

Esta herramienta **es un documento crucial que mapea y rastrea los requerimientos del usuario con los casos de prueba.** Se compone de todos los requisitos propuestos por el cliente y su trazabilidad en un solo documento, entregado al concluir el desarrollo del software.

El propósito principal de la RTM es validar que todos los requisitos se verifiquen mediante casos de prueba, garantizando que ninguna funcionalidad quede fuera del alcance durante las pruebas de software. Para asegurar que ningún requisito quede excluido del ciclo de pruebas, empleamos herramientas fundamentales:

- **Escenario de Prueba**: Representa uno o más flujos de negocio o flujos críticos ejecutados concurrentemente por usuarios virtuales sobre la infraestructura objetivo.
- **Caso de Prueba:** Documento o plantilla en un sistema de gestión de pruebas que establece variables y condiciones predeterminadas para verificar la funcionalidad de una aplicación. Este "cómo" se complementa con el "qué", representado por el escenario de prueba, que es una secuencia de casos de prueba que se ejecutan para verificar la funcionalidad de la aplicación.

Al entender la interacción entre casos de prueba y escenarios de prueba, se optimiza la validación de requisitos y se garantiza una cobertura completa durante el proceso de pruebas de software.

Explorando la Matriz de Trazabilidad: Entendiendo sus Elementos Clave

- **Cobertura de Requerimientos:** Se visualiza la cobertura de requerimientos con el número de casos de prueba, expresado típicamente en porcentaje. Esta métrica proporciona una visión clara del alcance y la exhaustividad de las pruebas.
- **Estado de Diseño y Ejecución:** La matriz refleja el estado de diseño y ejecución para casos de prueba específicos, ofreciendo una perspectiva inmediata sobre el progreso y la efectividad de las pruebas.
- **Defectos Relacionados :**Los defectos identificados y su estado actual también pueden ser indicados en la misma matriz, brindando una comprensión integral de la calidad del software.

- **Registro de Actividades de Prueba:** La matriz sirve como un registro centralizado de todas las actividades de prueba, facilitando un monitoreo eficiente del progreso y proporcionando transparencia en las operaciones del equipo de pruebas.

Además, es posible realizar el seguimiento de requerimientos utilizando herramientas de gestión de pruebas disponibles. Algunos de los objetivos clave que se logran mediante esta herramienta incluyen:

- Confirmación de la cobertura del 100% de prueba, asegurando que todas las pruebas diseñadas se hayan ejecutado completamente.
- Identificación de requerimientos no probados o inconsistencias en la documentación.
- Exposición de defectos generales o el estado de ejecución, con un enfoque particular en los requerimientos del negocio.
- Facilitación del análisis y estimación del impacto en el trabajo del equipo de control de calidad, especialmente en la revisión o reelaboración de casos de prueba.

¿Qué parámetros debemos incluir en la RTM?

- ID de requerimiento (ID: es la identificación o número/letras que lo identifica al requisito, el cuál es único, no se repite)
- Tipo de requisito y descripción
- Casos de prueba con estado

REQUERIMIENTO NRO.	DESCRIPCIÓN	TEST CASE ID	STATUS
123	Login del sistema	TC01, TC02, TC03	TC01: Pass TC02: Pass
345	Creación de Ticket	TC04, TC05, TC06, TC07, TC08, TC09, TC010	TC04: Pass TC05: Pass TC06: Pass TC06: Fail TC07: No run
456	Consultar Ticket	TC011, TC012, TC013, TC014	TC011: Pass TC012: Fail TC013: Pass TC014: No run

Imagen: Ejemplo RTM. Fuente: elaboración propia.

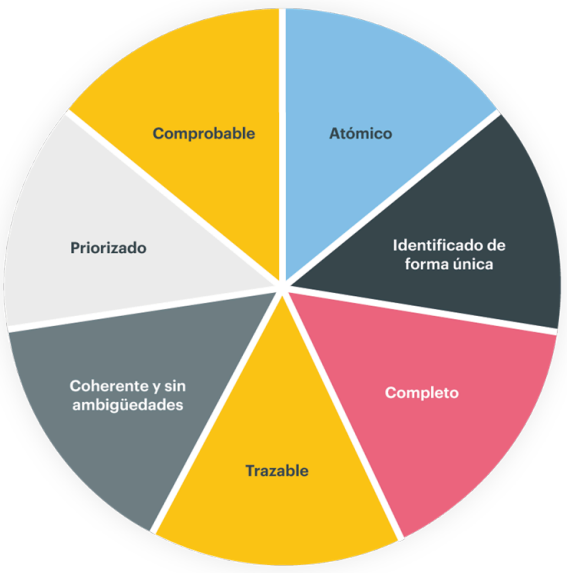
¿Quieres ver un ejemplo práctico?. Te dejamos 👉**acceso a un modelo** sencillo, detallando el paso a paso en su confección

3. ANÁLISIS DE AUTOMATIZACIÓN

La automatización se configura como una estrategia de optimización que permite la replicación consistente de pruebas en cada nueva versión de software disponible para su evaluación. La implementación de la automatización implica la necesidad de habilidades programáticas, ya que los profesionales de QA especializados en automatización crean sus propias pruebas para cada caso que desean examinar.

Durante la fase de requisitos, el equipo de control de calidad realiza un análisis exhaustivo del alcance de la automatización, centrándose particularmente en las pruebas de regresión. En caso de que la automatización se incorpore al alcance, el equipo toma decisiones cruciales: selecciona la herramienta a utilizar, define las funcionalidades que serán objeto de automatización, establece el marco de tiempo y asigna los recursos necesarios para el desarrollo de la automatización. Una vez concluido este análisis, el equipo de control de calidad presenta el Informe de Viabilidad de Automatización a diversas partes interesadas para su aprobación.

Cómo analizar los requisitos



Un requisito bien elaborado debe incorporar estas características esenciales:

- **Atómico**: Cada requisito debe ser atómico, detallado al nivel mínimo y sin la posibilidad de descomponerse en componentes separados. La unidad indivisible asegura claridad y precisión.
- **Identificado de forma única :** La identificación única de cada requisito es fundamental. Esto evita confusiones y garantiza que cada requisito sea claramente distinguido de los demás.
- **Completo:** Cada requisito debe ser completo en sí mismo. No debe haber lagunas o información faltante que pueda generar malentendidos o ambigüedades.
- **Coherente y sin ambigüedades:** La coherencia y la ausencia de ambigüedades son esenciales. Los requisitos deben ser consistentes entre sí y libres de interpretaciones dudosas para una comprensión uniforme.
- **Trazable:** La trazabilidad es clave, ya que permite seguir y entender la relación entre diferentes niveles de requisitos. Esto facilita el rastreo de cambios y la comprensión del impacto en todo el sistema.
- **Priorizado:** Cada requisito debe ser priorizado, proporcionando al equipo una guía clara sobre qué implementar primero y qué se puede abordar en etapas posteriores. Esto optimiza la planificación y ejecución del proyecto.
- **Comprobable**: Todos los requisitos deben ser comprobables, lo que significa que su cumplimiento puede ser evaluado de manera objetiva. Esto facilita la verificación y validación durante todo el proceso de desarrollo.

Al adherirse a estas características, se garantiza que los requisitos sean claros, completos y gestionables, sentando las bases para un desarrollo de software eficiente y libre de malentendidos.

Considera un ejemplo de un sistema de software educativo donde un estudiante puede registrarse para diferentes cursos.

A continuación te presentamos una tabla con tres columnas:

1. La primera columna indica la calidad del requisito.
2. La segunda columna indica el requisito mal formulado.
3. La tercera columna es igual a la segunda columna, pero el requisito fue transformado a un buen requisito.

REQUISITO DE CALIDAD	EJEMPLO DE MAL REQUISITO	EJEMPLO DE BUEN REQUISITO
Atómico	Los estudiantes podrán matricularse en cursos de grado y posgrado	Los estudiantes podrán matricularse en cursos de pregrado Los estudiantes podrán matricularse en cursos de posgrado
Identificado	1.Los estudiantes podrán inscribirse en cursos de pregrado 1.Los estudiantes podrán inscribirse en cursos de posgrado	1.Inscripción al curso 1.1.Los estudiantes podrán matricularse en cursos de pregrado 1.2.Los estudiantes podrán matricularse en cursos de posgrado
Completo	Un usuario profesor iniciará sesión en el sistema proporcionando su nombre de usuario, contraseña y otra información relevante	Un usuario profesor iniciará sesión en el sistema proporcionando su nombre de usuario, contraseña y código de departamento
Coherente y sin ambigüedades	Un estudiante tendrá cursos de pregrado o cursos de posgrado, pero no ambos. Algunos cursos estarán abiertos tanto para estudiantes de pregrado como para posgraduados	Un estudiante tendrá ya sea estudiantes de pregrado o de posgrado, pero no ambos
Rastreable	¿Mantener la información del estudiante asignada a BRD req.ID?	Mantener la información del estudiante: asignado a BRD req ID 4.1
Prioridad	Estudiante registrado priorizado-Prioridad 1Mantener información de usuario-Prioridad 1Inscribir cursos-Prioridad 1Ver boleta de calificaciones-Prioridad 1	Registrar estudiante-Prioridad 1Mantener información de usuario-Prioridad 2Inscribir cursos-Prioridad 1Ver boleta de calificaciones-Prioridad3
Comprobable	Cada página del sistema se cargará en un período de tiempo aceptable	Registrar estudiante e inscribir cursos Las páginas del sistema se cargarán en 5 segundos

Imagen 7.12: Ejemplos de requisitos. Fuente: elaboración propia.

Resultado de la fase de análisis de requerimientos

Luego de completar la fase de "Análisis de requerimientos", tendremos esta documentación[2] a disposición de los equipos:

- Documento de comprensión de requisitos.
- Escenarios de alto nivel.
- Estrategia de prueba de alto nivel y aplicabilidad de prueba.

Análisis de requisitos de software aplicado

Un análisis exhaustivo de los requisitos no solo ahorra tiempo, sino que también elimina errores, reduce las suposiciones entre los equipos y genera satisfacción entre los desarrolladores.

Un requisito de software es la piedra angular que detalla las necesidades, ya sean funcionales o no funcionales, que deben implementarse en un sistema. La funcionalidad implica proporcionar servicios específicos al usuario, como el **requisito funcional** de una aplicación bancaria que permite al cliente ver el saldo de su cuenta al seleccionar "Ver saldo".

No obstante, los requisitos de software no se limitan a lo funcional; también pueden ser **no funcionales**, como un requisito de rendimiento que dicta que cada pantalla de aviso del sistema debe ser visible para los usuarios durante 5 segundos.

Para garantizar claridad y especificidad, los requisitos de software se expresan comúnmente como declaraciones. Esta práctica facilita la comprensión y evita malentendidos, sentando así las bases para el desarrollo preciso y eficiente del sistema.

Tipos de requisitos

- **Requisitos comerciales:** Estos requisitos, extraídos del caso comercial del proyecto, establecen directrices de alto nivel. Por ejemplo, en un sistema de servicios bancarios móviles para el sudeste asiático, el requisito comercial para India podría ser el resumen de cuenta y la transferencia de

---

2

fondos, mientras que para China sería el resumen de cuenta y el pago de facturas.

PAÍS	Empresa proveedora de Funcionalidades o servicios bancarios
INDIA	Resumen de cuenta y transferencia de fondos
CHINA	Resumen de cuenta y pago de facturas

Imagen: Ejemplos de tipos de requisitos.

- **Requisitos arquitectónicos y de diseño**: Este nivel de requisitos, más detallado que los comerciales, define el diseño general necesario para implementar los requisitos comerciales. Tomando como referencia una organización educativa, los casos de uso de arquitectura y diseño podrían incluir funciones como inicio de sesión y detalles del curso. Estos casos se traducen en requisitos que guían la estructura y la interfaz del sistema.

CASO DE USO BANCARIO	REQUISITO
Pago de facturas	Este caso de uso describe cómo un cliente puede iniciar sesión en la banca neta y utilizar el servicio de pago de facturas. El cliente podrá ver un tablero de facturas pendientes de facturadores registrados. Puede agregar, modificar y eliminar un detalle del facturador. El cliente puede configurar alertas por SMS, correo electrónico para diferentes acciones de facturación. Puede ver el historial de las facturas pagadas en el pasado. Los actores que inician este caso de uso son clientes bancarios o personal de soporte.

Imagen.: Caso de uso bancario y requisito.

- **Requisitos del sistema y de integración:** En el nivel más detallado, encontramos los requisitos del sistema y de integración, proporcionando una descripción minuciosa de cada aspecto. Utilizando un ejemplo del módulo de pago de facturas, estos requisitos se presentan en forma de

historias de usuarios, empleando un lenguaje comercial cotidiano. Estos detalles abundantes permiten a los desarrolladores comenzar la codificación, especificando, por ejemplo, los requisitos para agregar un emisor de facturas.

PAGOS DE FACTURAS	REQUERIMIENTOS
Agregar Facturadores	• Nombre del proveedor de servicios públicos • Relación Número de cliente • Pagos automáticos: sí/no • Pagar la factura completa: sí/no • Límite de pago automático: no pague si la factura supera el monto especificado

Imagen : Pago de facturas y requerimientos.

En ciertos proyectos, puede suceder que no se reciba una documentación formal de requisitos. Sin embargo, existen diversas fuentes alternativas para recopilar la información esencial que respaldará el diseño y desarrollo del software. Algunas de estas fuentes incluyen:

- **Transferencia de Conocimientos:** Conversaciones con colegas o empleados que ya trabajan en el proyecto pueden ser una fuente invaluable de información. La experiencia previa de quienes están familiarizados con el contexto del proyecto puede proporcionar perspectivas clave.
- **Colaboración con Stakeholders:** Mantener discusiones con el analista comercial, el gerente de producto, el líder del proyecto y los desarrolladores ofrece una vía efectiva para comprender las necesidades y expectativas. Estas interacciones directas permiten obtener claridad sobre los objetivos del proyecto.
- **Análisis de Versiones Anteriores:** Examinar la versión anterior del sistema ya implementada puede revelar requisitos existentes y patrones de funcionamiento. Esto proporciona una base valiosa para la evolución del software.

- **Revisión de Documentación Anterior:** Analizar documentos de requisitos anteriores del proyecto proporciona información histórica que puede ser relevante para el desarrollo actual. Esto incluye documentos de requisitos y guías de instalación previas.
- **Revisión de Errores Anteriores:** Los informes de errores anteriores son una fuente valiosa. Algunos informes de errores se convierten en solicitudes de mejora que pueden ser implementadas en la versión actual del software.
- **Guía de Instalación:** Consultar la guía de instalación, si está disponible, brinda detalles sobre los requisitos de instalación, lo que es crucial para la correcta implementación y funcionamiento del software.
- **Análisis del Dominio:** Comprender el dominio o el conocimiento de la industria que el equipo intenta implementar proporciona una base sólida para identificar requisitos específicos del contexto.

Sea cual sea la fuente de requisitos que se utilice, es imperativo documentarlos de manera adecuada. La revisión por parte de otros miembros del equipo con experiencia y conocimientos enriquece la calidad de la información recopilada y asegura una comprensión compartida dentro del equipo de desarrollo.