Universidad Intercontinental de la Empresa

Asignatura

Arquitectura del Software 

Profesor

Yago Fontenla Seco

{*yago.fontenla1@uie.edu*}

Contenido

Definición y clasificación de los requisitos de software
Referencia: Sommerville, Capítulo 4.
Requisitos funcionales vs no funcionales.

Documentación de requisitos: Especificación y modelos
Referencia: Sommerville, Capítulo 4 .
Plantillas para el Documento de Requisitos de Software (SRS).

Especificación de requisitos
Referencia: Sommerville, Capítulo 4.
Métodos formales y semi-formales de especificación y lenguajes de especificación estructurada.

Técnicas de adquisición y análisis de requisitos
Referencia: Sommerville, Capítulo 4.
Técnicas de entrevistas, encuestas, observación y análisis de documentos.

Validación y gestión de requisitos
Referencia: Sommerville, Capítulo 4.
Cómo validar requisitos y gestionar cambios en ellos durante el ciclo de vida del software.

Contenido

Definición y clasificación de los requisitos de software
Referencia: Sommerville, Capítulo 4.
Requisitos funcionales vs no funcionales.

Documentación de requisitos: Especificación y modelos
Referencia: Sommerville, Capítulo 4 .
Plantillas para el Documento de Requisitos de Software (SRS).

Especificación de requisitos
Referencia: Sommerville, Capítulo 4.
Métodos formales y semi-formales de especificación y lenguajes de especificación estructurada.

Técnicas de adquisición y análisis de requisitos
Referencia: Sommerville, Capítulo 4.
Técnicas de entrevistas, encuestas, observación y análisis de documentos.

Validación y gestión de requisitos
Referencia: Sommerville, Capítulo 4.
Cómo validar requisitos y gestionar cambios en ellos durante el ciclo de vida del software.

Ciclo de Vida del Software – Etapas

Un proceso de software es una serie de actividades relacionadas que conduce a la elaboración de un producto de software. El ciclo de vida del software abarca todas las fases por las que pasa un sistema de software, desde su concepción hasta su retiro.

Existen distintas **metodologías o modelos de proceso** de software, pero **todos deben incluir las actividades fundamentales**:

1. **Especificación**: Definir qué debe hacer el software.
2. **Desarrollo**: Crear y codificar el software.
3. **Validación**: Verificar que cumple con los requisitos.
4. **Evolución**: Modificarlo para responder a nuevas necesidades.

Estas actividades están presentes en todos los procesos de software. Dada su complejidad, **incluyen subactividades como la validación de requerimientos**, diseño arquitectónico y pruebas unitarias. También se realizan actividades de soporte, como la documentación y la gestión de la configuración.

Especificación del software

La **especificación del software** o **ingeniería de requerimientos** (requisitos) implica definir lo que debe hacer el sistema y las limitaciones bajo las cuales debe operar. Consiste en el proceso de comprender y definir qué servicios se requieren del sistema, así como la identificación de las restricciones sobre la operación y el desarrollo del sistema.

Durante este proceso se documentan los **requerimientos funcionales** y **no funcionales** que guiarán el desarrollo y posterior validación del sistema. Esto incluye:

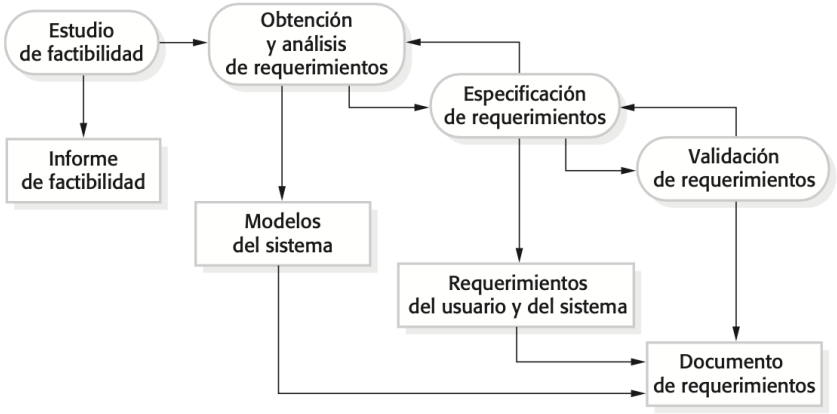
- **Requerimientos funcionales**: Describen el comportamiento que el sistema debe exhibir.
- **Requerimientos no funcionales**: Definen las restricciones en cuanto a rendimiento, seguridad, fiabilidad, etc.

El objetivo es **crear una base sólida para el diseño y desarrollo del software**, **minimizando ambigüedades** y **asegurando que se cumplan las expectativas del cliente**.

Especificación - Etapas

La **especificación del software** o **ingeniería de requerimientos** es una etapa particularmente crítica del proceso de software. Existen cuatro actividades principales en el proceso de ingeniería de requerimientos:

1. **Estudio de factibilidad** (viabilidad)
2. **Obtención y análisis de requerimientos** (requisitos)
3. **Especificación de requerimientos** (requisitos)
4. **Validación de requerimientos** (requisitos)

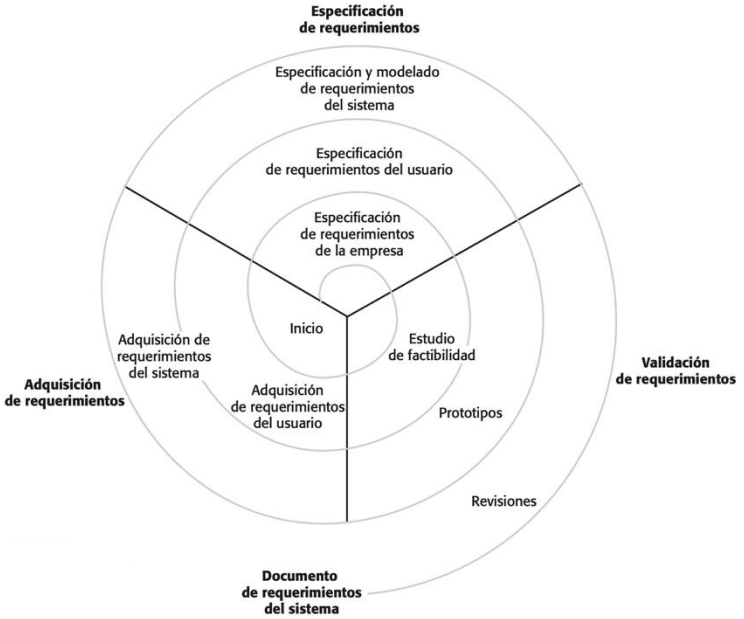


Especificación - Etapas

¿Cómo se lleva a cabo el proceso de especificación del software o ingeniería de servicios?

Inicialmente vimos este proceso como un proceso secuencial, pero en la práctica la ingeniería de requisitos es un proceso iterativo en el que las actividades están entrelazadas.

Este proceso se organiza de **forma iterativa**, donde en cada iteración se refina y detalla la comprensión de los requisitos. A medida que avanza el proceso, se invierte más esfuerzo en el análisis detallado de los requisitos funcionales y no funcionales, ajustándose a la etapa del desarrollo y al tipo de sistema.



Especificación - Estudio de factibilidad (viabilidad)

El **estudio de factibilidad** es la primera fase del proceso de ingeniería de requerimientos. Se realiza para determinar si el sistema propuesto es viable desde el punto de vista técnico y financiero. Incluye:

- **Evaluación técnica**: ¿Es posible desarrollar el sistema con las tecnologías de software y hardware disponibles?
- **Evaluación de coste-beneficio**: ¿Es rentable desarrollar el sistema, considerando las restricciones presupuestales?
- **Resultados**: Un informe que recomienda si debe o no continuar el proyecto hacia un análisis más detallado.

Esta fase es rápida y económica, pero crítica para evitar inversiones en proyectos inviables.

Especificación - Obtención y análisis de requerimientos

Esta fase consiste en **derivar los requerimientos** del sistema mediante varias técnicas:

- **Observación**: Estudio de los sistemas existentes.
- **Entrevistas y reuniones**: Discusiones con usuarios y proveedores potenciales.
- **Prototipos**: Desarrollo de modelos o prototipos del sistema para entender mejor las funcionalidades requeridas.

El objetivo es captar todos los requerimientos necesarios y obtener una visión clara de las expectativas del usuario para el sistema. Esto puede incluir el desarrollo de uno o más modelos de sistemas y prototipos, lo que ayuda a entender el sistema que se va a especificar.

Especificación - Especificación de requerimientos

La **especificación de requerimientos** consiste en documentar la información recopilada durante el análisis en un **documento formal** llamado **documento de especificación de requisitos** o Software Requirements Specificacion (SRS) document. Este documento incluye:

- **Requerimientos de usuario**: Descripciones abstractas del sistema orientadas a clientes y usuarios finales.
- **Requerimientos del sistema**: Descripciones detalladas de la funcionalidad y características técnicas que el sistema debe cumplir para los desarrolladores.

El objetivo es garantizar que todos los involucrados tengan una comprensión clara y acordada de lo que el sistema debe hacer.

Especificación - Validación de requerimientos

La **validación de requerimientos** tiene como propósito asegurar que los requerimientos documentados sean **realistas, consistentes y completos**. Durante esta fase:

- Se identifican errores y ambigüedades en el documento de requerimientos.
- Se realizan correcciones para mejorar la precisión y evitar problemas en fases posteriores del desarrollo.
- Se asegura que el sistema desarrollado realmente cumpla con las expectativas del cliente.

Es una etapa crucial para garantizar la calidad y viabilidad del software final.

¿Qué es un requisito?

Un requisito es una **descripción formal de lo que un sistema de software debe hacer**, incluyendo los servicios que ofrece y las restricciones bajo las cuales debe operar. Estos requisitos reflejan las **necesidades de los clientes** o usuarios para cumplir un propósito específico, como controlar un dispositivo, procesar un pedido o buscar información.

El término “requisito” puede tener diferentes niveles de abstracción: en algunos casos, se refiere a enunciados de alto nivel que describen de manera general los servicios o restricciones del sistema, mientras que en otros, puede ser una definición detallada y formal de las funciones y comportamientos que el sistema debe implementar.

La ingeniería de requerimientos es el proceso de descubrir, analizar, documentar y verificar estos servicios y restricciones

Requisitos de usuario y de sistema

Algunos de los problemas que surgen durante el proceso de ingeniería de requerimientos son resultado del fracaso de hacer una separación clara entre esos diferentes niveles de descripción. En este texto se distinguen con el uso del término “requerimientos del usuario” para representar los requerimientos abstractos de alto nivel; y “requerimientos del sistema” para caracterizar la descripción detallada de lo que el sistema debe hacer.

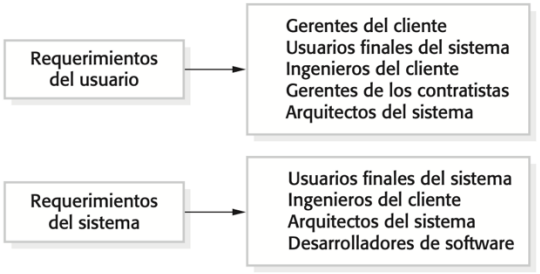
Los requerimientos del usuario y los requerimientos del sistema se definen del siguiente modo:

1. Los **requerimientos del usuario** son enunciados, en un lenguaje natural junto con diagramas, acerca de qué servicios esperan los usuarios del sistema, y de las restricciones con las cuales éste debe operar.

2. Los **requerimientos del sistema** son descripciones más detalladas de las funciones, los servicios y las restricciones operacionales del sistema de software. El documento de requerimientos del sistema (llamado en ocasiones especificación funcional) tiene que definir con exactitud lo que se implementará. Puede formar parte del contrato entre el comprador del sistema y los desarrolladores del software.

Requisitos de usuario y de sistema

Los diferentes niveles de requerimientos son útiles debido a que **informan sobre el sistema a distintos tipos de lector**. El requerimiento del usuario es muy general. Los requerimientos del sistema ofrecen información más específica sobre los servicios y las funciones del sistema que se implementará.



Los primeros no, están interesados en la manera en que se implementará el sistema, y quizá sean administradores a quienes no les atraigan las facilidades detalladas del sistema.

Mientras que los segundos necesitan conocer con más precisión qué hará el sistema, ya que están inmersos en la implementación del sistema.

Definición del requerimiento del usuario

1. El MHC-PMS elaborará mensualmente informes administrativos que revelen el costo de los medicamentos prescritos por cada clínica durante ese mes.

Especificación de los requerimientos del sistema

1.1 En el último día laboral de cada mes se redactará un resumen de los medicamentos prescritos, su costo y las clínicas que los prescriben.
1.2 El sistema elaborará automáticamente el informe que se imprimirá después de las 17:30 del último día laboral del mes.
1.3 Se realizará un reporte para cada clínica junto con los nombres de cada medicamento, el número de prescripciones, las dosis prescritas y el costo total de los medicamentos prescritos.
1.4 Si los medicamentos están disponibles en diferentes unidades de dosis (por ejemplo, 10 mg, 20 mg) se harán informes por separado para cada unidad de dosis.
1.5 El acceso a los informes de costos se restringirá a usuarios autorizados en la lista de control de acceso administrativo.

Requisitos funcionales

Los requisitos funcionales son declaraciones sobre los **servicios que debe proporcionar el sistema** y cómo debe comportarse ante determinadas entradas. En algunos casos, también pueden especificar comportamientos que el sistema no debe ejecutar.

Los requisitos funcionales incluyen:
Entrada y salida de datos.
Comportamiento bajo ciertas condiciones.
Procesos específicos del sistema.

Importancia:
Definir adecuadamente los requisitos funcionales es crucial para garantizar que el sistema entregue los servicios esperados por los usuarios y se comporte de manera adecuada en diferentes escenarios de uso.

Requisitos funcionales

Por ejemplo, veamos algunos casos de requerimientos funcionales para el sistema MHC-PMS, que se usan para mantener información de pacientes que reciben tratamiento por problemas de salud mental:

1. Un usuario podrá buscar en todas las clínicas las listas de citas.
2. El sistema elaborará diariamente, para cada clínica, una lista de pacientes que se espera que asistan a cita ese día.
3. Cada miembro del personal que usa el sistema debe identificarse de manera indivi- dual con su número de ocho dígitos.

En principio, la especificación de los requerimientos funcionales de un sistema debe ser completa y consistente. Totalidad significa que deben definirse todos los servicios requeridos por el usuario. Consistencia quiere decir que los requerimientos tienen que evitar definiciones contradictorias.

Requisitos no funcionales

Los requisitos no funcionales son **restricciones o limitaciones sobre los servicios** o funciones del sistema. Se aplican al sistema como un todo, afectando su arquitectura general más que funciones específicas.

Tipos de requisitos no funcionales:

- **Rendimiento:** Transacciones por segundo, tiempo de respuesta, etc.
- **Fiabilidad:** Tiempo promedio entre fallas, tasa de fallos permitida.
- **Seguridad:** Acceso restringido a usuarios autorizados, protección ante ataques.
- **Usabilidad:** Facilidad de uso para los usuarios después de un período de formación.
- **Mantenibilidad:** Facilidad con la que el sistema puede ser actualizado o corregido.
- **Escalabilidad:** Capacidad del sistema para crecer sin perder rendimiento.

Requisitos no funcionales

Características:

Los requerimientos no funcionales **afectan más la arquitectura global** de un sistema que los componentes individuales. Por ejemplo, para garantizar que se cumplan los requerimientos de rendimiento, quizá se deba organizar el sistema para minimizar las comunicaciones entre componentes.

Un requerimiento no funcional individual, como un requerimiento de seguridad, **podría generar algunos requerimientos funcionales** relacionados que definan nuevos servicios del sistema que se requieran. Además, también podría generar requerimientos que restrinjan los requerimientos ya existentes.

Importancia:

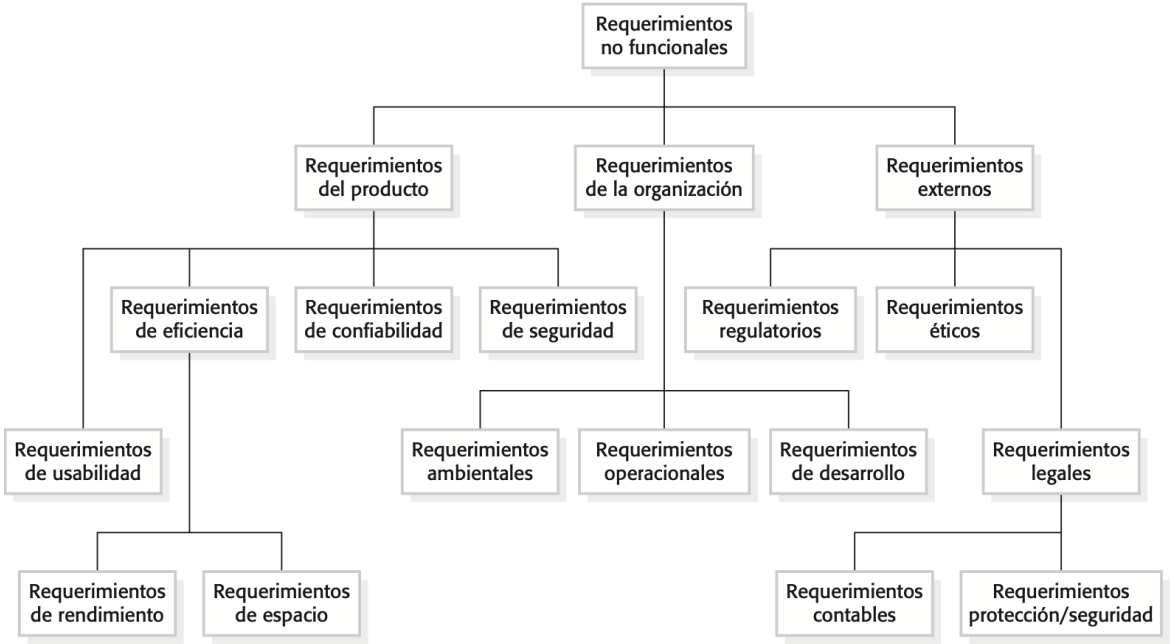
A menudo, los requisitos no funcionales son más críticos que los funcionales, ya que, si no se cumplen, todo el sistema puede fallar en cumplir su propósito.

Requisitos no funcionales

Ejemplos:
Un requisito no funcional de rendimiento puede establecer que un sistema médico debe estar disponible en todas las clínicas durante el horario laboral con un tiempo de inactividad máximo de 5 segundos por día.

Relación con los requisitos funcionales:
En muchos casos, los requisitos no funcionales pueden generar nuevos requisitos funcionales para asegurar que las restricciones se cumplan. Por ejemplo, un requisito de seguridad no funcional puede generar la necesidad de un sistema de autenticación funcional.

Requisitos no funcionales



Requisitos no funcionales

Siempre que sea posible, se deberán escribir de manera **cuantitativa los requerimientos no funcionales**, de manera que puedan **ponerse objetivamente a prueba**. En la tabla se muestran una serie de métricas que se utilizan para especificar propiedades no funcionales del sistema.

Propiedad	Medida
Rapidez	Transacciones/segundo procesadas Tiempo de respuesta usuario/evento Tiempo de regeneración de pantalla
Tamaño	Mbytes Número de chips ROM
Facilidad de uso	Tiempo de capacitación Número de cuadros de ayuda
Fiabilidad	Tiempo medio para falla Probabilidad de indisponibilidad Tasa de ocurrencia de falla Disponibilidad
Robustez	Tiempo de reinicio después de falla Porcentaje de eventos que causan falla Probabilidad de corrupción de datos en falla
Portabilidad	Porcentaje de enunciados dependientes de objetivo Número de sistemas objetivo

Contenido

Definición y clasificación de los requisitos de software
Referencia: Sommerville, Capítulo 4.
Requisitos funcionales vs no funcionales.

Documentación de requisitos: Especificación y modelos
Referencia: Sommerville, Capítulo 4 .
Plantillas para el Documento de Requisitos de Software (SRS).

Especificación de requisitos
Referencia: Sommerville, Capítulo 4.
Métodos formales y semi-formales de especificación y lenguajes de especificación estructurada.

Técnicas de adquisición y análisis de requisitos
Referencia: Sommerville, Capítulo 4.
Técnicas de entrevistas, encuestas, observación y análisis de documentos.

Validación y gestión de requisitos
Referencia: Sommerville, Capítulo 4.
Cómo validar requisitos y gestionar cambios en ellos durante el ciclo de vida del software.

Documento de requermientos del Software

¿Qué es el Documento de Requisitos del Software o Software Requirements Specification (SRS) Document?

Es un documento formal que recopila y describe los requisitos funcionales y no funcionales que debe cumplir un sistema de software. Sirve como contrato entre el cliente y el equipo de desarrollo, asegurando que ambos comprendan lo que el sistema debe hacer.

Contenido base del documento SRS

- **Objetivos del sistema:** Define el propósito del sistema y los beneficios que aportará.
- **Alcance:** Delimita lo que se incluye y lo que queda fuera del desarrollo del sistema.
- **Requisitos funcionales:** Especifica las funciones y características que el sistema debe proporcionar (ej. procesamiento de transacciones, autenticación de usuarios).
- **Requisitos no funcionales:** Impone restricciones sobre el rendimiento, la seguridad, la usabilidad, etc. (ej. el sistema debe responder en menos de 2 segundos).
- **Casos de uso:** Describe los escenarios de interacción entre los usuarios y el sistema.
- **Especificaciones técnicas:** Proporciona detalles sobre plataformas, lenguajes de programación y otras tecnologías a utilizar.

Esquema

Una posible organización para un documento de requerimientos basada en un estándar del IEEE para documentos de requerimientos (IEEE, 1998).

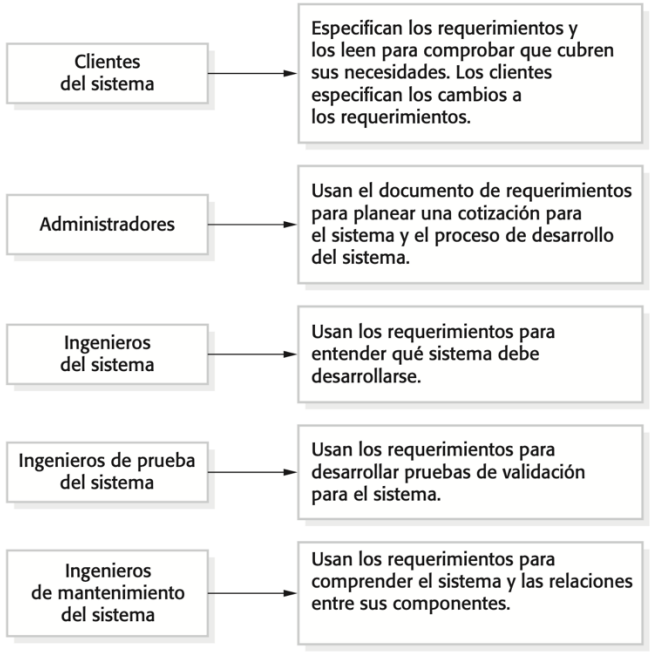
1. **Prefacio:** Define el número esperado de lectores del documento, incluye la historia de versiones y un resumen de los cambios en cada versión.
2. **Introducción:** Describe la necesidad del sistema, las funciones que proporcionará y cómo se ajusta a los objetivos empresariales o estratégicos.
3. **Glosario:** Define los términos técnicos utilizados en el documento, evitando conjeturas sobre el conocimiento del lector.
4. **Definición de Requisitos del Usuario:** Detalla los servicios que el sistema ofrecerá al usuario y los requisitos no funcionales, utilizando un lenguaje claro y accesible.
5. **Arquitectura del Sistema:** Ofrece una visión de alto nivel de la arquitectura del sistema, incluyendo la distribución de las funciones y los componentes reutilizables.
6. **Especificación de Requisitos del Sistema:** Incluye los requisitos funcionales y no funcionales del sistema, con todos los detalles técnicos y las interfaces necesarias.
7. **Modelos del Sistema:** Puede contener modelos gráficos que describan las relaciones entre los componentes del sistema, como modelos de flujo de datos o de objeto.
8. **Evolución del Sistema:** Describe cómo evolucionará el sistema en respuesta a cambios en hardware, necesidades del usuario o requisitos tecnológicos.
9. **Apéndices:** Proporciona información específica y detallada sobre hardware, bases de datos y configuraciones necesarias para el sistema.
10. **Índice:** Incluye un índice alfabético o de diagramas que facilite la navegación por el documento.

Usos del SRS

¿Cómo se usa el SRS?

Distintos usuarios hacen un uso diferente de este document **en base a sus intereses** y conocimiento sobre el sistema software que se desarrollará.

La diversidad de posibles usuarios significa que el documento de requerimientos debe ser un **compromiso** entre la **comunicación** de los requerimientos a los clientes, la **definición** de los requerimientos con detalle preciso para desarrolladores y examinadores, y la inclusión de información sobre la posible **evolución** del sistema.



SRS en metodologías ágiles

¿Siempre hay que definir un SRS?

Son esenciales los documentos de requerimientos cuando un contratista externo diseña el sistema de software. Sin embargo, los métodos de desarrollo ágiles argumentan que los requerimientos cambian tan rápidamente que un documento de requerimientos se vuelve obsoleto tan pronto como se escribe, así que el esfuerzo se desperdicia en gran medida.

En lugar de un documento formal, los enfoques como la programación extrema (Beck, 1999) recopilan de manera incremental requerimientos del usuario y los escriben en tarjetas como historias de usuario. De esa manera, el usuario da prioridad a los requerimientos para su implementación en el siguiente incremento del sistema.

Contenido

Definición y clasificación de los requisitos de software
Referencia: Sommerville, Capítulo 4.
Requisitos funcionales vs no funcionales.

Documentación de requisitos: Especificación y modelos
Referencia: Sommerville, Capítulo 4 .
Plantillas para el Documento de Requisitos de Software (SRS).

Especificación de requisitos
Referencia: Sommerville, Capítulo 4.
Métodos formales y semi-formales de especificación y lenguajes de especificación estructurada.

Técnicas de adquisición y análisis de requisitos
Referencia: Sommerville, Capítulo 4.
Técnicas de entrevistas, encuestas, observación y análisis de documentos.

Validación y gestión de requisitos
Referencia: Sommerville, Capítulo 4.
Cómo validar requisitos y gestionar cambios en ellos durante el ciclo de vida del software.

Especificación de requisitos

La **especificación de requerimientos** es el proceso de documentar formalmente los requerimientos del usuario y los requerimientos del sistema en un documento de requisitos. El objetivo es lograr una descripción precisa del sistema que sea fácil de entender tanto para los clientes como para los desarrolladores. Para ello, los requisitos deben ser:

Claros y comprensibles: Evitar ambigüedades que puedan llevar a malentendidos.

Completos: Cubrir todas las funcionalidades y restricciones necesarias.

Consistentes: Evitar conflictos entre diferentes requisitos.

Los **requerimientos del usuario** deben describir los requerimientos funcionales y no funcionales, de forma que sean **comprensibles para los usuarios** del sistema que no cuentan con un conocimiento técnico detallado. Deberían especificar sólo el **comportamiento externo** del sistema.

Los **requerimientos del sistema** son versiones extendidas de los requerimientos del usuario que los **ingenieros de software** usan como punto de partida para el diseño del sistema. **Añaden detalles y explican** cómo el sistema debe brindar los requerimientos del usuario.

El **documento de requerimientos no debe incluir detalles de la arquitectura o el diseño del sistema**. Por ello, los requisitos de usuario, no deben tener jerga de software, anotaciones estructuradas o formales. Deben escribirse en lenguaje natural, con tablas y formas sencillas, así como diagramas intuitivos.

Notaciones para la especificación de requisitos

Existen diversas notaciones utilizadas para escribir los requerimientos de un sistema. Entre las más comunes se encuentran:

Lenguaje natural: La forma más usada para escribir requerimientos en una prosa entendible por todos.

Lenguaje natural estructurado: Usa un formato estandarizado para asegurar la consistencia.

Lenguajes de descripción de diseño: Más abstractos, se usan para definir un modelo operacional del sistema.

Anotaciones gráficas: Como los diagramas UML, usados para describir la interacción del sistema.

Especificaciones formales: Se basan en matemáticas para reducir la ambigüedad en sistemas críticos.

Notación	Descripción
Enunciados en lenguaje natural	Los requerimientos se escriben al usar enunciados numerados en lenguaje natural. Cada enunciado debe expresar un requerimiento.
Lenguaje natural estructurado	Los requerimientos se escriben en lenguaje natural en una forma o plantilla estándar. Cada campo ofrece información de un aspecto del requerimiento.
Lenguajes de descripción de diseño	Este enfoque usa un lenguaje como un lenguaje de programación, pero con características más abstractas para especificar los requerimientos al definir un modelo operacional del sistema. Aunque en la actualidad este enfoque se usa raras veces, aún tiene utilidad para especificaciones de interfaz.
Anotaciones gráficas	Los modelos gráficos, complementados con anotaciones de texto, sirven para definir los requerimientos funcionales del sistema; los casos de uso del UML y los diagramas de secuencia se emplean de forma común.
Especificaciones matemáticas	Dichas anotaciones se basan en conceptos matemáticos como máquinas o conjuntos de estado finito. Aunque tales especificaciones sin ambigüedades pueden reducir la imprecisión en un documento de requerimientos, la mayoría de los clientes no comprenden una especificación formal. No pueden comprobar que representa lo que quieren y por ello tienen reticencia para aceptarlo como un contrato de sistema.

Especificación en lenguaje natural

La **notación en lenguaje natural** es la forma **más común** de especificar requerimientos de software. Es simple, intuitiva y adecuada para usuarios sin conocimientos técnicos profundos. Sin embargo, tiene problemas como:

Ambigüedad: Diferentes personas pueden interpretar el mismo requerimiento de forma distinta.

Falta de estructura: Puede ser difícil asegurar que todos los requerimientos estén organizados adecuadamente.

Recomendaciones:

- **Distinguir entre requisitos obligatorios y deseables:**
- - Requisitos obligatorios: Usar el verbo “debe” para señalar las funciones que el sistema debe soportar.
- - Requisitos deseables: Utilizar el verbo “debería” para los requisitos no obligatorios o que son opcionales.
- **Resaltar las partes clave del requisito:** Emplear negritas, cursivas o colores para destacar los elementos más importantes de los requisitos, facilitando su comprensión rápida.
- **Evitar el uso de jerga técnica:** No asumir que los lectores entienden términos técnicos. Evitar el uso de palabras como “arquitectura” o “módulo” sin explicarlas. Además, es recomendable no usar abreviaturas ni acrónimos que puedan generar confusión.
- **Asociar una razón con cada requisito:** Cada requisito debe tener una razón asociada que explique por qué fue incluido. Esto es útil cuando los requisitos cambian, ya que permite evaluar la importancia de cada cambio.

Especificación en lenguaje natural – Ejemplo

Estandarizar el formato de los requisitos: Utilizar un formato estándar para expresar los requisitos ayuda a evitar omisiones y facilita su revisión. Un formato útil podría incluir el requisito en una oración breve, seguido de una **explicación** del motivo o fuente del requisito, para saber a quién consultar en caso de cambios.

Aquí se pueden ver como ejemplo, dos casos de uso para un software embebido de control de una bomba de insulina automatizada especificados en lenguaje natural.

3.2 Si se requiere, cada 10 minutos el sistema medirá el azúcar en la sangre y administrará insulina. (*Los cambios de azúcar en la sangre son relativamente lentos, de manera que no son necesarias mediciones más frecuentes; la medición menos periódica podría conducir a niveles de azúcar innecesariamente elevados.*)

3.6 Cada minuto, el sistema debe correr una rutina de autoevaluación, con las condiciones a probar y las acciones asociadas definidas en la tabla 1. (*Una rutina de autoevaluación puede detectar problemas de hardware y software, y prevenir al usuario sobre el hecho de que la operación normal puede ser imposible.*)

En ambos ejemplos como se incluye una explicación del motivo o fuente del requisito, para saber a quién consultar en caso de cambios.

Especificación estructurada

La **notación estructurada** es un enfoque para especificar requerimientos utilizando un **formato estándar** que impone más estructura que el lenguaje natural tradicional. Este tipo de notación conserva la claridad del lenguaje natural, pero asegura que los requerimientos sean más consistentes y uniformes. Algunos elementos clave son:

- **Formato Estandarizado:** Se emplean plantillas o formularios para especificar cada requisito, asegurando que todos los campos clave, como razones, dependencias y fuentes, sean cubiertos.
- **Claridad y Organización:** El uso de plantillas permite organizar la información de manera clara, reduciendo la posibilidad de omisiones o interpretaciones ambiguas.
- **Simplicidad y Expresividad:** Aunque estructurado, mantiene la expresividad del lenguaje natural, permitiendo que los usuarios no técnicos comprendan los requerimientos.
- **Aplicaciones Comunes:** Se puede estructurar en torno a objetos, funciones o eventos que el sistema debe manejar.

En proyectos donde es necesario asegurar una consistencia estricta entre los requerimientos, la notación estructurada se utiliza para reducir errores y mejorar la claridad de la especificación.

Especificación estructurada

En proyectos donde es necesario **asegurar una consistencia estricta entre los requerimientos**, la notación estructurada se utiliza para reducir errores y mejorar la claridad de la especificación.

Una forma estándar para especificar requerimientos funcionales, debe incluir la siguiente información:

1. Una **descripción de la función** o entidad a especificar.
2. Una **descripción de sus entradas** y sus procedencias.
3. Una **descripción de sus salidas** y a dónde se dirigen.
4. Información sobre los **datos requeridos** para el cálculo u otras entidades en el sistema que se utilizan (la parte “requiere”).
5. Una **descripción de la acción** que se va a tomar.
6. Si se usa un enfoque funcional, una **precondición** establece lo que debe ser verdadero antes de llamar a la función, y una **postcondición** especifica lo que es verdadero después de llamar a la función.
7. Una descripción de los **efectos colaterales** (si acaso hay alguno) de la operación.

Especificación estructurada – Ejemplo

Bomba de insulina/Software de control/SRS/3.3.2

Función	Calcula dosis de insulina: nivel seguro de azúcar.
Descripción	Calcula la dosis de insulina que se va a suministrar cuando la medición del nivel de azúcar actual esté en zona segura entre 3 y 7 unidades.
Entradas	Lectura del azúcar actual (r2), las dos lecturas previas (r0 y r1).
Fuente	Lectura del azúcar actual del sensor. Otras lecturas de la memoria.
Salidas	CompDose: la dosis de insulina a administrar.
Destino	Ciclo de control principal.
Acción	CompDose es cero si es estable el nivel de azúcar, o cae o si aumenta el nivel pero disminuye la tasa de aumento. Si el nivel se eleva y la tasa de aumento crece, CompDose se calcula entonces al dividir la diferencia entre el nivel de azúcar actual y el nivel previo entre 4 y redondear el resultado. Si la suma se redondea a cero, en tal caso CompDose se establece en la dosis mínima que puede entregarse.
Requerimientos	Dos lecturas previas, de modo que puede calcularse la tasa de cambio del nivel de azúcar.
Precondición	El depósito de insulina contiene al menos la dosis individual de insulina máxima permitida.
Postcondición	r0 se sustituye con r1, luego r1 se sustituye con r2.
Efectos colaterales	Ninguno.

Contenido

Definición y clasificación de los requisitos de software
Referencia: Sommerville, Capítulo 4.
Requisitos funcionales vs no funcionales.

Documentación de requisitos: Especificación y modelos
Referencia: Sommerville, Capítulo 4 .
Plantillas para el Documento de Requisitos de Software (SRS).

Especificación de requisitos
Referencia: Sommerville, Capítulo 4.
Métodos formales y semi-formales de especificación y lenguajes de especificación estructurada.

Adquisición y análisis de requisitos
Referencia: Sommerville, Capítulo 4.
Técnicas de entrevistas, encuestas, observación y análisis de documentos.

Validación y gestión de requisitos
Referencia: Sommerville, Capítulo 4.
Cómo validar requisitos y gestionar cambios en ellos durante el ciclo de vida del software.

Adquisición y análisis de requisitos

Después de un estudio de viabilidad inicial, la siguiente etapa del proceso de ingeniería de requerimientos es la adquisición y el análisis de requerimientos.

La adquisición y análisis de requisitos es la fase del proceso de ingeniería de requisitos en la que los ingenieros trabajan con los usuarios y clientes para descubrir qué servicios debe proporcionar el sistema, las restricciones de hardware y otras especificaciones. Este proceso es crucial para entender tanto los requerimientos del usuario como los requerimientos del sistema.

Las **actividades clave** en este proceso incluyen:

1. Descubrimiento de Requisitos
2. Clasificación y Organización
3. Priorización y Negociación
4. Especificación

Descubrimiento de requisitos

¿Cómo se descubren los requisitos del sistema?

El **descubrimiento o adquisición de requisitos** es el proceso de recopilar información sobre el sistema requerido y los sistemas existentes, así como de separar, a partir de esta información, los requerimientos del usuario y del sistema.

En este proceso se interactúa con los participantes del sistema para descubrir sus requerimientos. También los requerimientos de dominio de los participantes y la documentación se descubren durante esta actividad. Se utilizan técnicas como entrevistas, observaciones, y análisis de documentos para recopilar información sobre los sistemas actuales y los nuevos requerimientos.

Descubrimiento de requisitos

¿Qué fuentes de información se utilizan para el descubrimiento de requisitos?

Las **fuentes de información** durante la fase de descubrimiento de requerimientos incluyen documentación, participantes del sistema y especificaciones de sistemas similares.

Usuarios: Quienes interactúan directamente con el sistema.

Documentación: Especificaciones de sistemas existentes.

Sistemas similares: Información que se puede transferir de sistemas relacionados.

Clasificación y organización

Una vez descubiertos, ¿cómo se clasifican?

En esta actividad, los requisitos identificados durante el descubrimiento son agrupados y organizados de manera coherente. Los requisitos relacionados se colocan juntos, generalmente utilizando un modelo arquitectónico para asociarlos con diferentes subsistemas. En la práctica, la ingeniería de requerimientos y el diseño arquitectónico no son actividades separadas completamente.

Este proceso ayuda a dar estructura a los requisitos, facilitando la visualización de la arquitectura del sistema.

Priorización y negociación

¿Cómo se priorizan los requisitos?

A menudo los requisitos entran en **conflicto** entre los diferentes participantes del proyecto. La priorización se encarga de determinar cuáles son más importantes, mientras que la negociación se enfoca en resolver los conflictos.

Este proceso incluye:

Reuniones con los participantes para lograr compromisos.

Priorización de los requisitos en base a su importancia y viabilidad.

Para realizar esta priorización, se hacen estudios de viabilidad de cada uno de los requisitos y se evalúa su dificultad e importancia (cuánto aportan o mejoran el sistema) y se priorizan en base a este ratio. Pueden utilizarse gráficos coste-beneficio para realizar esta priorización.

Especificación

En esta actividad, los requisitos se documentan en un formato que permita su utilización en el diseño y desarrollo del sistema. La especificación puede ser formal o informal, dependiendo del nivel de detalle requerido.

Los requisitos se organizan y refinan iterativamente a lo largo del proceso.

El objetivo es lograr una descripción precisa del sistema que sea fácil de entender tanto para los clientes como para los desarrolladores.

Dificultades en el proceso de adquisición

La adquisición y la comprensión de los requerimientos por parte de los participantes del sistema es un **proceso difícil** por diferentes razones:

1. Los participantes con frecuencia **no saben lo que quieren** de un sistema, excepto en términos muy generales; pueden encontrar difícil articular qué quieren que haga el sistema; pueden hacer **peticiones inalcanzables** porque no saben qué es factible y qué no lo es.
2. Los participantes en un sistema expresan naturalmente los requerimientos con sus **términos y conocimientos implícitos de su trabajo**. Los ingenieros de requerimientos, sin experiencia en el dominio del cliente, podrían no entender dichos requerimientos.
3. Diferentes participantes tienen **distintos requerimientos** y pueden expresarlos en **variadas formas**. Los ingenieros de requerimientos deben descubrir todas las fuentes potenciales de requerimientos e identificar similitudes y conflictos.
4. Factores **políticos** llegan a influir en los requerimientos de un sistema. Los administradores pueden solicitar requerimientos específicos del sistema, porque éstos les permitirán aumentar su influencia en la organización.
5. El **ambiente económico y empresarial** donde ocurre el análisis es dinámico. Inevitablemente cambia durante el proceso de análisis. Puede cambiar la importancia de requerimientos particulares; o bien, tal vez surjan nuevos requerimientos de nuevos participantes a quienes no se consultó originalmente.

Técnicas de descubrimiento – Entrevistas

Las entrevistas son una técnica común para recopilar información directamente de los usuarios y otros participantes. Pueden ser:

Estructuradas: Con preguntas predefinidas.

No estructuradas: Conversaciones abiertas que permiten obtener más información.

Son útiles para descubrir tanto requisitos funcionales como no funcionales.

Técnicas de descubrimiento – Escenarios

Los escenarios son descripciones de cómo se espera que los usuarios interactúen con el sistema. Proporcionan ejemplos específicos que permiten a los participantes visualizar cómo el sistema respondería en situaciones reales.

Simulan interacciones con el sistema.

Se pueden complementar con diagramas o prototipos.

Técnicas de descubrimiento – Casos de Uso

Los casos de uso son una técnica estructurada que identifica los actores (usuarios o sistemas) y sus interacciones con el sistema. Cada caso de uso describe una función del sistema desde la perspectiva de un usuario.

- Utiliza diagramas y descripciones textuales.
- **Muy común en el modelado orientado a objetos.**

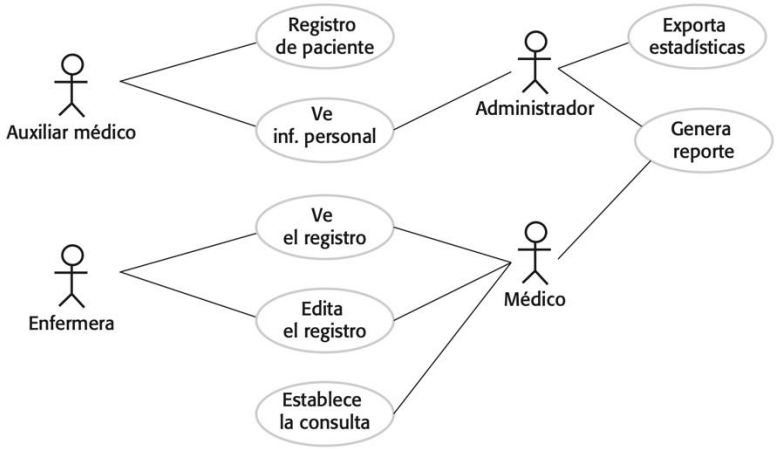
Los casos de uso identifican las interacciones individuales entre el sistema y sus usuarios u otros sistemas. Cada caso de uso debe documentarse con una descripción textual. Entonces pueden vincularse con otros modelos en el UML que desarrollará el escenario con más detalle.

Ejemplo:
*El **establecimiento de consulta** permite que dos o más médicos, que trabajan en diferentes consultorios, vean el mismo registro simultáneamente. Un médico inicia la consulta al elegir al individuo involucrado de un menú desplegable de médicos que estén en línea. Entonces el registro del paciente se despliega en sus pantallas, pero sólo el médico que inicia puede editar el registro. Además, se crea una ventana de chat de texto para ayudar a coordinar las acciones. Se supone que, de manera separada, se establecerá una conferencia telefónica para comunicación por voz.*

Técnicas de descubrimiento – Casos de Uso

Los **casos de uso** se documentan con el empleo de un **diagrama de caso de uso** de alto nivel. El conjunto de casos de uso representa todas las interacciones posibles que se describirán en los requerimientos del sistema.

Los actores en el proceso, que pueden ser individuos u otros sistemas, se representan como figuras sencillas. Cada clase de interacción se constituye como una elipse con etiqueta. Líneas vinculan a los actores con la interacción. De manera opcional, se agregan puntas de flecha a las líneas para mostrar cómo se inicia la interacción.



Relación entre Casos de Uso y Requisitos

La **especificación de requisitos** implica **definir detalladamente lo que el sistema debe hacer**, tanto a nivel funcional como no funcional, y abarca la recopilación de información sobre lo que los diferentes actores esperan del sistema.

Una vez establecidos estos requisitos, los **casos de uso** sirven para detallar **cómo los usuarios interactuarán con el sistema**, especificando las secuencias de acciones que permitirán cumplir con esos requisitos.

La **diferencia** clave entre ambos conceptos **radica en su enfoque**: la especificación de requisitos describe qué debe hacer el sistema, mientras que los casos de uso detallan cómo los actores interactúan con el sistema para cumplir con esos objetivos.

Los casos de uso son una herramienta útil para **validar y afinar los requisitos funcionales**, **enfocándose en las interacciones específicas** entre los usuarios y el sistema

Relación entre Casos de Uso y Requisitos

Una vez definidos los requisitos y los casos de uso, tiene sentido **vincular a cada caso de uso** los requisitos tanto funcionales como no funcionales que le apliquen. Relacionar cada caso de uso con los requisitos correspondientes ayuda a garantizar que el sistema cumpla con lo que se espera de él en todos los aspectos. Esto no solo asegura que **cada caso de uso cubra los comportamientos esperados**, sino también que el sistema cumpla con características clave como rendimiento, seguridad o usabilidad. Este proceso de vinculación permite:

1. **Trazabilidad:** Facilita el seguimiento de cómo los requisitos específicos se implementan a través de los casos de uso.
2. **Cobertura completa:** Garantiza que todos los requisitos funcionales estén cubiertos por al menos un caso de uso y que no haya requisitos olvidados.
3. **Validación y verificación:** Durante la fase de pruebas, podrás validar que el sistema satisface los requisitos funcionales y verificar que cumple con los no funcionales bajo escenarios específicos.

En la práctica, se usa una matriz de trazabilidad para hacer explícitas estas relaciones entre casos de uso y requisitos, lo que es especialmente útil en proyectos grandes o complejos.

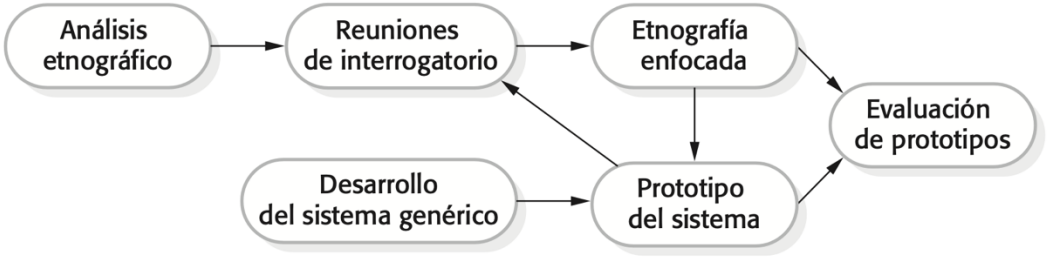
Técnicas de descubrimiento – Etnografía

La etnografía es una técnica de observación en la que los analistas **se sumergen en el entorno de trabajo de los usuarios**. Observan cómo interactúan con los sistemas actuales y cómo llevan a cabo sus tareas, lo que ayuda a descubrir requisitos implícitos que no siempre son expresados durante entrevistas.

Ventajas:

Útil para sistemas con un alto componente social u organizacional.

Revela interacciones y requisitos que de otro modo pasarían desapercibidos.



Contenido

Definición y clasificación de los requisitos de software
Referencia: Sommerville, Capítulo 4.
Requisitos funcionales vs no funcionales.

Documentación de requisitos: Especificación y modelos
Referencia: Sommerville, Capítulo 4 .
Plantillas para el Documento de Requisitos de Software (SRS).

Especificación de requisitos
Referencia: Sommerville, Capítulo 4.
Métodos formales y semi-formales de especificación y lenguajes de especificación estructurada.

Adquisición y análisis de requisitos
Referencia: Sommerville, Capítulo 4.
Técnicas de entrevistas, encuestas, observación y análisis de documentos.

Validación y gestión de requisitos
Referencia: Sommerville, Capítulo 4.
Cómo validar requisitos y gestionar cambios en ellos durante el ciclo de vida del software.

Validación de requisitos

La **validación de requisitos** es el proceso mediante el cual se **verifica** que los requisitos documentados **reflejan realmente** lo que el cliente desea del sistema. Este **proceso es crucial**, ya que errores en los requisitos pueden resultar muy costosos si se descubren durante el desarrollo o después de la implementación del sistema.

El proceso de validación incluye las siguientes comprobaciones:

- **Validez:** Asegurar que los requisitos propuestos son realmente lo que el cliente necesita y que se consideran todas las funciones relevantes.
- **Consistencia:** Verificar que no hay conflictos entre los requisitos. Es decir, que no existan descripciones contradictorias de las mismas funciones o restricciones incompatibles.
- **Completitud:** El documento de requisitos debe definir todas las funciones esperadas y las restricciones necesarias para el sistema.
- **Realismo:** Los requisitos deben ser técnicamente realizables dentro de las limitaciones presupuestarias y de tiempo. Esto incluye evaluar si la tecnología disponible puede soportar los requisitos solicitados.
- **Verificabilidad:** Los requisitos deben escribirse de manera que sea posible verificarlos mediante pruebas específicas que demuestren que el sistema cumple con ellos.

Importancia:

Corregir problemas de requisitos después del desarrollo o durante la implementación puede resultar mucho más costoso que corregir errores de diseño o codificación, ya que los cambios en los requisitos pueden implicar modificaciones en todo el sistema.

Técnicas de validación

Existen diversas técnicas que pueden usarse de manera individual o combinada para validar los requisitos de un sistema. Estas técnicas ayudan a identificar errores, inconsistencias y problemas antes de que el sistema sea desarrollado:

- **Revisiones de Requisitos:** Un equipo de revisores analiza sistemáticamente los requisitos para identificar errores, ambigüedades o inconsistencias. Esta técnica es eficaz para descubrir problemas de forma temprana.
- **Creación de Prototipos:** Se desarrolla un prototipo ejecutable del sistema y se presenta a los usuarios finales y clientes, permitiendo que interactúen con él y verifiquen si cumple con sus necesidades. Los prototipos ayudan a visualizar cómo funcionará el sistema antes de su desarrollo completo.
- **Generación de Casos de Prueba:** Se diseñan pruebas basadas en los requisitos para comprobar su verificabilidad. Si una prueba es difícil de diseñar, esto suele indicar que el requisito es ambiguo o complejo, y que debe reconsiderarse. Esta técnica es fundamental en metodologías como la programación extrema (XP).

Desafíos en la Validación:

Es difícil para los usuarios visualizar completamente cómo el sistema se ajustará a sus necesidades reales. Como resultado, es probable que se requieran cambios en los requisitos incluso después de la validación formal y de haber acordado el documento de requisitos.

Administración de requisitos

La administración de requisitos es el proceso mediante el cual se controla y gestiona el cambio en los requisitos a lo largo del ciclo de vida del sistema. Los requisitos de los sistemas grandes siempre están sujetos a cambio, debido a la naturaleza dinámica del problema que intentan resolver y los entornos donde se implementan.

Razones por las que los requisitos cambian:

- **Entornos en constante cambio:** Después de la instalación, el entorno empresarial y tecnológico puede evolucionar, requiriendo que el sistema se ajuste a nuevos hardware, sistemas, o regulaciones.
- **Desconexión entre clientes y usuarios:** A menudo, quienes financian el sistema no son los mismos que lo usan. Los clientes imponen restricciones organizacionales, mientras que los usuarios, tras usar el sistema, descubren nuevas necesidades que requieren cambios en los requisitos.
- **Diferentes prioridades y usuarios diversos:** Los sistemas grandes tienen una comunidad diversa de usuarios, lo que genera conflictos entre los requisitos de diferentes grupos. Con el tiempo, la experiencia revela que el equilibrio de apoyo a distintos usuarios debe cambiar.

Planeación de la administración de requisitos

La planeación es una primera etapa esencial en el proceso de administración de requerimientos. Esta etapa establece el nivel de detalle que se requiere en la administración de requerimientos. Durante la etapa de administración de requerimientos, usted tiene que decidir sobre:

- **Identificación de requisitos:** Cada requisite debe identificarse de manera exclusive, de forma que pueda tener referencia cruzada con otros requisites.
- **Un proceso de administración del cambio:** Éste es el conjunto de actividades que valoran el efecto y costo de los cambios. En la siguiente sección se estudia con más detalle este proceso.
- **Políticas de seguimiento:** Dichas políticas definen las relaciones entre cada requeri- miento, así como entre los requerimientos y el diseño del sistema que debe registrarse. La política de seguimiento también tiene que definir cómo mantener dichos registros.
- **Herramientas de apoyo:** La administración de requerimientos incluye el proce- samiento de grandes cantidades de información acerca de los requerimientos. Las herramientas disponibles varían desde sistemas especializados de administración de requerimientos, hasta hojas de cálculo y sistemas de bases de datos simples.

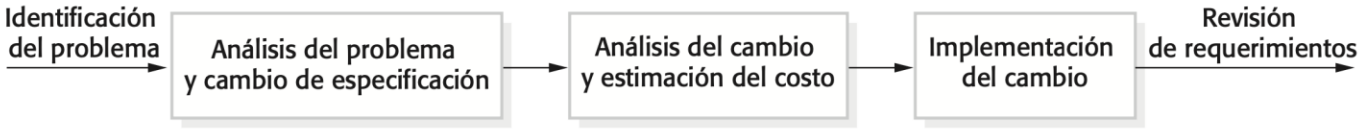
Gestión del cambio en los requisitos

Existen tres etapas principales de un proceso de administración del cambio:

1. Análisis del problema y especificación del cambio El proceso comienza con la identificación de un problema en los requerimientos o, en ocasiones, con una propuesta de cambio específica. Durante esta etapa, el problema o la propuesta de cambio se analizan para comprobar que es válida. Este análisis retroalimenta al solicitante del cambio, quien responderá con una propuesta de cambio de requerimien- tos más específica, o decidirá retirar la petición.

2. Análisis del cambio y estimación del costo El efecto del cambio propuesto se valora usando información de seguimiento y conocimiento general de los requerimientos del sistema. El costo por realizar el cambio se estima en términos de modificaciones al documento de requerimientos y, si es adecuado, al diseño y la implementación del sistema. Una vez completado este análisis, se toma una decisión acerca de si se procede o no con el cambio de requerimientos.

3. Implementación del cambio Se modifican el documento de requerimientos y, donde sea necesario, el diseño y la implementación del sistema. Hay que organizar el documento de requerimientos de forma que sea posible realizar cambios sin reescritura o reorganización extensos. Conforme a los programas, la variabilidad en los documentos se logra al minimizar las referencias externas y al hacer las secciones del documento tan modulares como sea posible. De esta manera, secciones individuales pueden modificarse y sustituirse sin afectar otras partes del documento.



Universidad Intercontinental de la Empresa

Asignatura

Arquitectura del Software 

Profesor

Yago Fontenla Seco

{*yago.fontenla1@uie.edu*}