Clase 06

Análisis de Sistemas

Materia:

Ingeniería de Requerimientos

Docente contenidista: PEPE, Jonathan Leonel

Revisión: Coordinación

Contenido

Clase 6

¡Te damos la bienvenida a la materia **Ingeniería de Requerimientos**!

En esta clase vamos a ver los siguientes temas:

- Especificación de requerimientos.
- Limitaciones en los requerimientos.
- Documento SRS (elementos, redacción y organización).

¿Listos para continuar?

En la clase anterior abordamos diferentes técnicas que podemos utilizar para relevar datos en la etapa de elicitación.

Hoy conoceremos un tipo de documento cuyo propósito es proporcionar una descripción completa del producto de software a desarrollar. A este documento, a partir de ahora, lo llamaremos SRS.

¡Seguimos adelante!

Especificación de requerimientos

Comencemos recordando la definición formal de **Especificación de requerimientos**:

> *"La especificación de requerimientos es el proceso de escribir, en un documento de requerimientos, los requerimientos del usuario y del sistema"*

Los requerimientos/requisitos de un sistema describen los servicios que ha de ofrecer el sistema y las restricciones asociadas a su funcionamiento.

Sigamos recordando otras definiciones clave:

Requerimientos:
Propiedades o restricciones determinadas de forma precisa que deben satisfacerse.

Requerimientos funcionales:
Qué debe hacer un sistema, sus funcionalidades esperadas.

Requerimientos no funcionales:
Cómo debe ser el sistema, sus atributos y características esperadas.

Los requerimientos…

- Se suelen especificar en lenguaje natural.
- Se expresan de forma individual.
- Se organizan de forma jerárquica a distintos niveles de detalle.
- Se numeran para facilitar su gestión (*RF* para los funcionales, *RNF* para los no funcionales).

A su vez, los requerimientos han de ser…

- **Claros** y **concretos**, evitando imprecisiones y ambigüedades, sin figuras retóricas (*recurso lingüístico que modifica el mensaje con el objetivo de embellecerlo*).
- **Correcto**, con una y sólo una interpretación.
- **Completos**, con toda la información necesaria para su comprensión.
- **Priorizado**, según el grado de necesidad.
- **Consistentes**, sin contradecir otros requerimientos.
- **Rastreable**, a la necesidad de usuario que satisface.

Por otro lado, los requerimientos han de indicar…

- Lo que se **espera** del sistema (funcionalidad y atributos).
- La **justificación** de por qué ha de ser así.
- Los **criterios de aceptación** que sean aplicables para verificar su cumplimiento.

Buenas prácticas

Ian Sommerville en su obra *Ingeniería de Software* establece una serie de buenas prácticas para minimizar una interpretación errónea de los requerimientos:

- Elaborar un formato estándar y asegurarse de que todas las definiciones de requerimientos se adhieran a dicho formato.
    - Al estandarizar el formato es menos probable cometer omisiones y es más sencillo comprobar los requerimientos.
- Utilizar el lenguaje de manera clara para distinguir entre **requerimientos obligatorios y deseables**.
- Usar texto resaltado (negrita, cursiva o color) para seleccionar partes clave del requerimiento.
- No inferir que los lectores entienden el lenguaje técnico de la ingeniería de software.
- Siempre que sea posible, asociar una razón con cada requerimiento de usuario que explique por qué se incluyó el requerimiento.
- Para los **requerimientos funcionales**:
    - Deben estar redactados de tal forma que sean comprensibles para usuarios sin conocimientos técnicos avanzados.
    - Deben especificar el comportamiento externo del sistema y evitar, en la medida de lo posible, establecer características de su diseño.
    - Deben priorizarse distinguiendo entre requisitos obligatorios y requisitos deseables.
- Para los **requerimientos no funcionales**:
    - Deben especificarse cuantitativamente (siempre que sea posible) para que se pueda verificar su cumplimiento.

A continuación, veamos algunos ejemplos:

REQUERIMIENTO FUNCIONAL: Matriculación de alumnos

La matriculación será realizada de forma interactiva.

El alumno seleccionará la materia a la cual desea matricularse.

El listado mostrará únicamente las materias que aún no ha aprobado del plan de estudio.

Para aceptar la matriculación se realizarán las validaciones descriptas a continuación:

- El alumno cumple la condición de alumno regular. Es decir, aprobó, al menos, una materia en los últimos dos años.
- El alumno tiene aprobadas todas las materias correlativas.

Se generará un comprobante de matriculación que el alumno podrá descargar e imprimir.

REQUERIMIENTO NO FUNCIONAL: Interfaces

- Hardware:
    - El sistema se debe implementar sobre la infraestructura existente en las oficinas administrativas de la organización.
    - Procesador i3 de 1.8 GHz, 4Gb de RAM, tarjeta de video de 64 MB, 15 GB libres en disco.
- Software:
    - La aplicación debe ser compatible con todas las versiones de Microsoft Windows, desde Windows 7.

Limitaciones en los requerimientos

En este punto, debemos ser consciente de que existen una gran cantidad de **limitaciones**, las cuales deben ser consideradas.

Esto significa que no siempre somos "*libres*" para diseñar la solución más adecuada para el negocio, porque no entran dentro de las limitaciones impuestas.

***¿Y cuáles son estas limitaciones*?**

La más obvia es la limitación financiera (*presupuesto*) o técnica (*hardware del cliente*).

El diseño sólo puede proporcionar la solución que se ajusta a todas las limitaciones conocidas actualmente, o bien intentar renegociar o elevar algunas de las limitaciones.

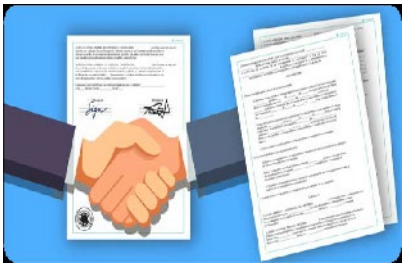
A esto lo conocemos como factibilidad del proyecto.

Documento SRS

Sabemos que la comunicación es la clave del éxito en el desarrollo de software.

La mala comunicación y los requisitos poco claros son algunas de las principales razones por las que los proyectos de software fracasan.

Los requisitos claros y bien comunicados ayudan a los equipos de desarrollo a crear el producto adecuado, lo que representa la base del desarrollo exitoso de productos.



En este sentido, un SRS (***Especificación de Requisitos de Software***) es un documento cuyo propósito es proporcionar una descripción completa de un producto de software a desarrollar.

El SRS es generalmente aprobado al final de la fase de Ingeniería de Requisitos, la fase más temprana en el proceso de desarrollo de software.

"El uso del SRS puede eliminar y prevenir errores en la fase de diseño, ya que cualquier requisito contradictorio y funciones que necesiten validación pueden ser corregidos en este punto y las partes interesadas pueden ser contactadas para su reevaluación"

Siempre es significativamente menos costoso hacer cambios al principio del proceso de desarrollo de software que más tarde, cuando ya se han gastado incontables horas y una gran cantidad de energía y recursos.

Estructura

No hay dos documentos SRS idénticos (*extensión, orden, elementos constitutivos*) porque todos los proyectos de software son diferentes.

Sin embargo, un documento SRS comúnmente incluye:

1. **El propósito de la aplicación**:
   El público objetivo, el uso previsto, el alcance de la misma y un glosario de términos.
2. **Descripción general de la aplicación**:
   Las necesidades de los usuarios, las suposiciones y dependencias que rodean a la aplicación.
3. **Detalle de los requisitos específicos**:
   Los requisitos de interfaz externa, los requisitos funcionales y los requisitos no funcionales.

El propósito de la aplicación

Dentro de este apartado se encuentra:

- **Público objetivo**:
  Describir al público a quien está destinado el producto.
- **Uso previsto**:
  Enumerar todas las formas posibles en que la audiencia podrá usar el producto dependiendo del rol.
- **Alcance del producto**:
  Brindar una descripción detallada del producto, indicando las funcionalidades para cada macroproceso y el propósito para el cual fue diseñado.
- **Glosario y acrónimos**:
  Colocar las definiciones de los términos que se usarán en el documento (*acrónimos, tecnicismo, jerga propia del negocio*) para asegurar que todos entiendan lo que se quiere decir.
- **Índice**:
  Incluir un índice para orientar a quien lee el documento.

Descripción general de la aplicación

Dentro de este apartado se encuentra:

- **Necesidades del usuario**:
  Referir a los problemas que los usuarios podrán resolver/automatizar con el sistema, es decir, aquello que motivará su adquisición y uso.
- Listar los usuarios directos e indirectos que utilizarán el producto.
- **Suposiciones y dependencias**:
  Describir cualquier factor que pueda interferir con el cumplimiento del objetivo, como así también, las dependencias de factores externos al proyecto.

Detalle de los requisitos específicos

Dentro de este apartado se encuentra:

- **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 ser:
  - **Las interfaces de usuarios**:
    Se refiere a la facilidad de uso de una aplicación. Incluye la presentación de contenido (*menú*), las opciones de navegación, entre otros.
  - **Las interfaces de hardware**:
    Se refiere a la interacción entre los componentes del software y del hardware del sistema. Incluye los dispositivos compatibles, el umbral óptimo de rendimiento, entre otros.
  - **Las interfaces de software**:
    Se refiere a la conexión e integración del producto con otros componentes de software. Incluye programas y bases de datos externas, bibliotecas, sistemas operativos, entre otros.
- **Requisitos funcionales**:
  Describir **qué** debe hacer el sistema, es decir, la funcionalidad requerida para el mismo. Incluye su comportamiento específico, las transformaciones que realiza sobre las entradas para producir salidas, las excepciones, la interacción con el ambiente, la respuesta ante un estímulo. Estos criterios tienen que ver con "*qué se creará*" en lugar de "*cómo*", no entrar en detalle sobre las pilas de tecnología, ya que pueden cambiar a medida que avanza el proyecto.
- **Requisitos no funcionales**:
  Indicar **cómo** debe ser el sistema, sus atributos, propiedades emergentes y características que definen de qué manera el sistema realizará el trabajo. Establecer los criterios de acuerdo con la eficacia con la que debe operar el sistema: umbrales de rendimiento, seguridad, compatibilidad, usabilidad, fiabilidad, mantenibilidad, portabilidad y adecuación funcional están incluidos en esta área (*Norma ISO 25000*).

Buenas prácticas

A continuación, mencionaremos una serie de buenas prácticas que debemos tener muy presente para la confección del documento SRS:

- **Utilizar un lenguaje claro y conciso**:
  Evitar la jerga técnica y las siglas que no se entiendan universalmente. Usar un lenguaje claro y directo, garantizando que todas las partes interesadas puedan comprender fácilmente el contenido.
- **Incluir ayudas visuales**:
  Mejorar la comprensión incorporando diagramas junto con descripciones textuales. Las ayudas visuales pueden proporcionar una representación más intuitiva de conceptos complejos y comportamientos del sistema.
- **Priorizar requisitos**:
  Definir claramente la prioridad de cada requisito. Utilizar etiquetas como "*imprescindible*", "*debería tener*" y "*es bueno tener*" para indicar la importancia relativa de las diferentes características. La priorización ayuda al equipo de desarrollo a centrarse primero en la funcionalidad crítica.
- **Mantenerlo actualizado**:
  Mantener el control de versiones del documento SRS. Actualizar periódicamente el documento para reflejar cualquier cambio en los requisitos del proyecto, el alcance o los comentarios de las partes interesadas. Contar con un registro de cambios claro para realizar un seguimiento (*rastreo*) de las modificaciones a lo largo del tiempo.
- **Involucrar a las partes interesadas**:
  Colaborar estrechamente con todas las partes interesadas relevantes durante todo el proceso de desarrollo del SRS. Esta participación fomenta una comprensión compartida de los objetivos del proyecto y de lo que se espera del sistema.
- **Ser integral**:
  No dejar lugar a interpretaciones o suposiciones. Proporcionar descripciones detalladas y completas de cada requisito, incluidos los aspectos funcionales y no funcionales.
  La ambigüedad en los requisitos puede provocar malentendidos y retrasos en el proyecto.

- **Utilizar un formato estructurado**:
  Organice el documento SRS en secciones bien definidas. Un formato estructurado facilita a los lectores localizar información específica. Incorporar un índice para cada sección del documento de manera de facilitar aún más su localización.
- **Garantizar la capacidad de prueba**:
  Escribir los requisitos de una manera que facilite las pruebas y la validación. Cada requisito debe ser verificable, permitiendo crear casos de prueba que validen si el sistema cumple con los criterios especificados. Es esencial contar con criterios de aceptación claros para cada requisito.
- **Evitar la ambigüedad**:
  Eliminar la ambigüedad en los requisitos. Utilizar un lenguaje preciso, evitando términos vagos y asegurar que no haya lugar para múltiples interpretaciones de un requisito. Las ambigüedades pueden provocar malentendidos que impacten negativamente en el desarrollo del sistema.
- **Considerar la escalabilidad futura**:
  Pensar en la escalabilidad a largo plazo del sistema de software. Anticiparse a posibles necesidades futuras y asegurarse de que el documento SRS las tenga en cuenta. Este enfoque proactivo puede ahorrar tiempo y recursos en el futuro.
- **Revisar y validar**:
  Realizar revisiones exhaustivas del documento SRS con las partes interesadas, incluido el cliente, el equipo de desarrollo y los expertos en la materia. Abordar cualquier discrepancia, inconsistencia o ambigüedad que surja durante el proceso de revisión. La validación garantiza que el documento represente con precisión los objetivos del proyecto.
- **Obtener aprobación formal**:
  Después de finalizar el documento SRS, obtener la aprobación formal del cliente o patrocinador del proyecto.
  Esto **formaliza el acuerdo** sobre el alcance y los requisitos del proyecto, proporcionando una base clara para el desarrollo.

Hemos llegado así al final de esta clase en la que vimos:

- Especificación de requerimientos.
- Limitaciones en los requerimientos.
- Documento SRS (elementos, redacción y organización).

Te esperamos en la **clase en vivo** de esta semana.
No olvides realizar el **desafío semanal**.

¡Hasta la próxima clase!

Bibliografía

Del Águila Cano, I. M. (2019). Ingeniería de requisitos.
Material didáctico. Cuaderno de teoría (Vol. 35). Universidad Almería.

Ramos, D., Noriega, R., Laínez, J. R., & Durango, A. (2017).
Curso de Ingeniería de Software: 2ª Edición. IT Campus Academy.

Vazquez, C. E., & Simoes, G. S. (2016).
Ingeniería de Requisitos: Software orientado al negocio. Brasport.