Sistemas de información 1

¿Qué es la determinación de requerimientos?

La determinación de requerimientos es el estudio de un sistema para conocer cómo trabaja y cuáles son las necesidades de la organización que no son satisfechas por el sistema actual.

Requerimiento: es una característica que el sistema debe poseer para lograr un objetivo, y por lo tanto, dicha característica debe ser incluida en el nuevo sistema.

Especificación de Requerimientos:

Requerimientos del usuario: son declaraciones en lenguaje natural y en diagramas, de los servicios que se espera que el sistema provea y las restricciones bajo las cuales debe operar. Su función es describir los requerimientos funcionales y no funcionales de forma tal que sean comprensibles por quienes no tienen un conocimiento técnico detallado.

Requerimientos del sistema: establecen con detalle los servicios y restricciones del sistema. Esta especificación es el punto de partida para el diseño del sistema, por parte de los Ingenieros de Software.

-Los requerimientos pueden clasificarse en tres categorías:

Requerimientos funcionales: Describen las funciones fundamentales a las que debe ajustarse el sistema. En algunos casos, los requerimientos funcionales de los sistemas también declaran explícitamente lo que el sistema no debe hacer.

Requerimientos no funcionales: Son aquellos requerimientos que no se refieren, directamente, a las funciones específicas que deberá entregar el sistema, sino a las propiedades emergentes del mismo como la fiabilidad, tiempo en que debe dar respuesta o la capacidad de almacenamiento.

Requerimientos del dominio: Son requerimientos que provienen del dominio de aplicación del sistema y que reflejan las características de ese dominio. Estos pueden ser a su vez, funcionales o no funcionales.

-Problemas que pueden surgir:

Falta de claridad: Es difícil utilizar el lenguaje en forma precisa y no ambigua sin detallar el documento y convertirlo en algo complicado de leer.

Confusión de requerimientos: No se distinguen claramente los requerimientos funcionales de los no funcionales, las metas del sistema y la información que es pertinente al diseño del sistema.

Conjunción de requerimientos: Se expresa como un único requerimiento a un conjunto de requerimientos diferentes.

Determinación de Requerimientos:

El proceso de determinación de los requerimientos es uno de los procesos más importantes, ya que de él dependerá gran parte del éxito del proyecto.

Elicitación: Captura las necesidades de los clientes que provienen del dominio del problema y de los usuarios, que serán recuperados a través de las técnicas de recolección de datos.

Especificación: Se encarga de representar el conocimiento en un conjunto de notaciones, utilizando las técnicas de documentación.

Validación: Los modelos de requerimientos se presentarán al usuario para que éste certifique que representa sus necesidades. También serán contrastados contra el dominio del problema. Si los modelos son correctos y están completos, el proceso de requerimientos ha concluido, de lo contrario es devuelto al subproceso de especificación con los problemas encontrados.

Actividades para la determinación de requerimientos:

-Las actividades a llevar a cabo pueden ser:

Reutilización de los requerimientos: Si el analista ya ha trabajado antes analizando otros dominios de aplicación similares, sabrá los requerimientos propios del dominio, así como también donde habitualmente se producen los problemas y pudiendo, de esta manera anticipar los requerimientos e investigar directamente las áreas de conflicto.

Investigación del sistema: Estudio del sistema actual utilizando las herramientas para detectar hechos.

Se lleva a cabo aplicando técnicas, herramientas y habilidades que le permitan al analista determinar y documentar las características del sistema.

Requerimientos Básicos:

Cuando se conduce una investigación, se debe procurar entender cada una de las actividades que se desarrollan y que quedaron dentro del límite del sistema, establecido en la etapa de reconocimiento.

Para entender estas actividades, debe estudiar: El procesamiento que se realiza, Las entradas al procesamiento, Las salidas del procesamiento, Las restricciones impuestas por los límites de tiempo y la carga de trabajo, Los controles efectuados, El usuario de las salidas.

Entender el Proceso: Para comenzar, se debe intentar entender lo básico de la actividad. Para hacerlo, lo primero es entender que es lo qué se hace y con qué finalidad. Esto dará una primera aproximación al entendimiento del sistema y las características que sirven para describirlo.

Identificar las Entradas y Salidas: El siguiente paso es identificar cuáles son los datos necesarios para realizar la actividad y qué información se produce. La identificación de las entradas y salidas, se obtiene fundamentalmente de los documentos que utilizan los usuarios para la realización de las tareas y de los documentos que producen.

Identificar el Volumen y la Frecuencia: La identificación de la cantidad de datos necesarios para la realización de la actividad o de los datos que producen, así como también de la frecuencia de realización, darán una idea del impacto que esta actividad posee sobre el sistema global. Esto permitirá, como consecuencia, valorar la importancia y su beneficio asociado en realizar los ajustes necesarios a esta actividad. Si la frecuencia es poca, y los volúmenes también son pocos, probablemente el costo asociado del ajuste, respecto de los beneficios obtenidos, sean muy altos como para que una solución pase por aquí. Si por el contrario, la actividad tiene una alta frecuencia o es muy tediosa su realización se pueden alcanzar altos beneficios ajustando la actividad.

Identificar los Controles: Identificar que hay problemas de controles en el sistema es un hecho sumamente importante, dado que amplía el costo del sistema actual haciendo que la solución propuesta tenga mayores beneficios.

Identificar a los Usuario de las Salidas: La identificación de los usuarios de las salidas ayuda a comprender las relaciones existentes entre los subsistemas y la importancia de éstas para los destinatarios. Luego, se puede ver cómo las salidas ayudan a los usuarios a completar el propósito de las actividades que realizan y verificar si es posible realizar mejoras, dado que las salidas aquí serán las entradas para las actividades de los destinatarios.

Técnicas para Elicitación de Requerimientos

Partiendo del usuario:

Existen una serie de dificultades que debemos considerar, puede darse el caso de que el usuario tenga poca claridad para expresarse, otra de las dificultades, es que se produzcan diferencias entre el usuario y el analista.

-Existen una serie de técnicas para elicitar requerimientos partiendo del usuario:

Entrevistas de comienzo y final abierto: El analista deja que el usuario hable respecto de cómo desarrolla sus tareas Son de gran utilidad para obtener visiones globales de cómo se realizan las tareas, pero, por lo general, no suelen ser una manera eficaz de obtener información detallada al respecto.

Entrevistas estructuradas: en este caso el analista tiene una serie de preguntas formuladas con antelación al momento de la entrevista. Tienen por finalidad direccionar al usuario hacia aspectos específicos de requerimientos a elicitar. Debido a ello son de gran utilidad para obtener información detallada. Se utilizan preguntas cerradas, abiertas, de sondeo y de control.

Brainstorming**:** Este tipo de reuniones, en las cuales varios usuarios debaten entre sí respecto de la manera en que se debe dar solución a una cuestión, es un espacio que ayuda al analista a comprender el dominio del problema. También puede utilizarse esta técnica para resolver los problemas de diferencias entre usuarios y analistas o para reducir la falta de consenso respecto a determinados aspectos clave del proyecto.

Escenarios / Casos de Uso:

Un escenario es una historia que ilustra como un sistema puede satisfacer determinadas necesidades de los usuarios. Consiste en una descripción idealizada, pero detallada, de una instancia específica de interacción usuario – sistema. Tienen la ventaja de que los usuarios encuentran mucho más fácil contar su experiencia a través de "contar una historia".

Análisis de la documentación existente:

Un formulario es una fuente muy importante para la elicitación de requerimientos pues: es un modelo formal de datos, por lo general suelen contener información sobre la organización, sus instrucciones de uso encierran conocimiento sobre el dominio y su análisis puede automatizarse.

Como desventaja, podemos plantear que muchas veces los formularios contienen información que ya no se utiliza o ha dejado de ser relevante. A este punto debe prestársele suma atención pues se corre el riesgo de incorporar en las estructuras de datos, atributos o datos elementales que dejaron de tener relevancia en el dominio de la aplicación.

Lenguaje Natural:

El lenguaje natural es la forma más habitual en la cual se representa el conocimiento, independientemente del tipo de soporte que se esté utilizando para su persistencia (oral o escrito). Es habitual que la mayoría de lo que vale la pena conocer sobre el dominio de un problema puede formularse en lenguaje natural.

Hay dos categorías de elicitación en lenguaje natural: las que se basan en la interacción con el usuario y los enfoques que elicitan desde un texto escrito en lenguaje natural.

Administración de Requerimientos

La administración de requerimientos es el proceso de comprender y controlar los cambios en los requerimientos del sistema.

¿Porqué los requerimientos cambian? Con el tiempo, el entorno del sistema y los objetivos del negocio seguramente cambiarán, por lo tanto, los requerimientos deben evolucionar para reflejar esto.

Requerimientos duraderos: Son relativamente estables, derivan de la actividad principal de la organización y están relacionados directamente con el dominio del sistema.

Requerimientos volátiles: Estos cambiarán probablemente con el desarrollo del sistema o después que este se haya puesto en operación. Pueden ser clasificados de la siguiente manera -Requerimientos mutantes: Son los que cambian debido a los cambios en el ambiente en el que opera la organización.

-Requerimientos emergentes: Emergen al incrementarse la comprensión del cliente en el desarrollo del sistema.

- Requerimientos consecutivos: Son el resultado de introducir el sistema. Esta introducción puede cambiar los procesos de la organización y abrir nuevas formas de trabajar que generarán nuevos requerimientos.

-Requerimientos de compatibilidad: Dependen de sistemas particulares o procesos de negocios dentro de la organización.

Planeación de la administración de requerimientos

Durante la etapa de administración de requerimientos se debe decidir sobre:

La identificación de requerimientos: cada requerimiento debe identificarse de forma única.

Un proceso de administración del cambio: este es un conjunto de actividades que evalúa el impacto y costo de los cambios.

Políticas de rastreo: Definen la relación entre los requerimientos, los requerimientos y el diseño del sistema que se debe registrar y la manera que los registros se deben mantener.

Ayudas de herramientas CASE: Las herramientas que se pueden utilizar van desde sistemas de administración de requerimientos especiales, hasta hojas de cálculo y sistemas sencillos de bases de datos.

Existen tres tipos de información de rastreo que se deben mantener:

La información de rastreo de la fuente: vincula los requerimientos con quienes los propusieron y la razón de estos.

La información de rastreo de los requerimientos: vincula los requerimientos dependientes en el documento de requerimientos.

La información de rastreo del diseño: vincula los requerimientos a los módulos del diseño en el cual serán implementados.

La administración de requerimientos necesita ayuda automática y las herramientas **CASE** que se utilizarán deben elegirse en la etapa de planeación. Se requiera ayuda de estas herramientas para:

Almacenar requerimientos: los requerimientos deben mantenerse en un almacén de datos seguro y administrado. Accesible para todos los que estén relacionados con el proceso de ingeniería de requerimientos.

Administrar el cambio: este proceso se simplifica si se dispone de una herramienta de ayuda.

Administrar el rastreo: para ayudar a descubrir relaciones entre los requerimientos, están disponibles algunas herramientas que utilizan técnicas de lenguaje natural.

OBJETIVOS DE LAS CASE:

Las herramientas CASE se producen para dar respuesta a alguno de los siguientes objetivos o a algún subconjunto de ellos:

-Mejorar la productividad en el desarrollo y mantenimiento del software: Las herramientas CASE deben dar visibilidad al proceso de desarrollo de software así como a la documentación del mismo. En la fase de mantenimiento contar con documentación actualizada del producto facilita esta actividad así como asiste en la evaluación del impacto de los cambios.

-Aumentar la calidad de los productos de software: Un producto desarrollado aplicando un proceso controlado y visible tiene mayor probabilidad de tener una adecuada calidad que uno que se desarrolla sin aplicar modelos de proceso preestablecidos.

-Reducir el tiempo y costo del proceso de desarrollo y facilitar el mantenimiento de los productos de software: La capacidad de dotar de visibilidad al proceso y a la documentación del producto resulta esenciales al momento de estimar el costo y el tiempo para la fase de mantenimiento.

-Mejorar la planificación y administración de los proyectos de desarrollo de software. 1. Las herramientas CASE facilitan la administración de los proyectos posibilitando el trabajo en equipo y la coordinación de tareas entre quienes participan del proceso.

-Aumentar la biblioteca de conocimiento de una organización ayudando a la búsqueda de soluciones para los requisitos: Administrar una biblioteca de requerimientos es el punto de partida para la reutilización de requerimientos y la reutilización del software.

Automatizar el desarrollo del software, la documentación, la generación de código, las pruebas de errores y la gestión del proyecto: Todas las tareas que puedan automatizarse del proceso de desarrollo de software nos permiten disminuir el tiempo de desarrollo así como contar con el resultado de esas actividades sin necesidad de la intervención humana.

Ayudar a la reutilización del software, portabilidad y estandarización de la documentación: La unificación de la documentación de los productos que se administran con una herramienta CASE facilita la intervención de distintos profesionales en el proceso de desarrollo de software que no participaran en el proceso desde su generación. Esta característica disminuye fuertemente la dependencia de algunas personas en particular para la evolución de los productos de software.

-Posibilitar la gestión de todas las fases de desarrollo de software con una misma herramienta: Las herramientas CASE unifican en un solo entorno las distintas tareas que se llevan a cabo durante el proceso de desarrollo de software.

Facilitar el uso de las distintas metodologías propias de la ingeniería del software: Por lo general, las herramientas CASE trabajan aplicando las metodologías de uso frecuente en la Ingeniería de Software unificando el lenguaje mediante el que se comunican las ideas y modelos.

COMPONENTES DE LAS HERRAMIENTAS CASE :

Diccionario de Datos o Repositorio: en el cual se almacenarán todos los elementos que se definan desde la herramienta. Este diccionario debiera ser administrado por la CASE como un sistema de base de datos

Meta Modelo: es la definición de las técnicas y metodologías que soporta la herramienta CASE. Podemos representarlo como la configuración de la manera en que operará la herramienta.

Facilidad para la carga de datos: la herramienta debiera tener la posibilidad de importar datos provenientes de otro software y prever la descarga de información de manera automática. Podemos definir a esta característica como la capacidad de interactuar con otros sistemas.

Capacidad de comprobar errores de consistencia: efectuando análisis de exactitud, integridad y consistencia de los esquemas generados por la herramienta.

Interfaz de usuario amigable: la interfaz de usuario deberá permitir editar texto, dibujar modelos, etc.

TIPO DE HERRAMIENTAS CASE:

CASE de alto nivel: son las herramientas que automatizan o apoyan las fases finales o superiores del ciclo de vida del desarrollo de sistemas como la planificación de sistemas, el análisis y el diseño de sistemas.

CASE de bajo nivel: son las herramientas que automatizan o apoyan las fases finales o inferiores del ciclo de vida como el diseño detallado de sistemas, la implantación de y las tareas de soporte.

CASE de ciclo de vida: son las herramientas que apoyan las actividades que tienen lugar a lo largo de todo el ciclo de vida, se incluyen actividades como la gestión de proyectos y la estimación de esfuerzo.

En la actualidad, podrían clasificarse a las herramientas CASE de la siguiente manera:

TOOLKIT: es una colección de herramientas integradas que permiten automatizar un conjunto de tareas de algunas de las fases del ciclo de vida: Planificación estratégica, Análisis, Diseño, Generación de código.

WORKBENCH: Son conjuntos integrados de herramientas que dan soporte a la automatización del proceso completo de desarrollo del producto de software. Permiten cubrir el ciclo de vida completo. El producto final aportado por ellas es el código ejecutable y su documentación.

UPPER CASE: Planificación estratégica, Requerimientos.

MIDDLE CASE: Análisis y Diseño.

LOWER CASE: Generación de código, test e implantación.

La SRS es una especificación para un producto de software en particular, programa, o conjunto de programas que realizan ciertas funciones en un ambiente específico.

Naturaleza de la SRS:

-Los problemas básicos que se presentan al escribir una SRS van dirigidos a lo siguiente: La funcionalidad,Las interfaces Externas, Los Atributos, Las restricciones de diseño impuestas sobre una implementación.

Ambiente de la SRS:

La SRS debe definir todos los requisitos del software correctamente. Un requisito del Software puede existir debido a la naturaleza de la tarea a ser resuelta o debido a una característica especial del proyecto.

-No debe describir ningún detalle de diseño o implementación. Éstos deben describirse en la fase del diseño del proyecto.

-No debe imponer restricciones adicionales sobre el software. Éstas son especificadas apropiadamente en otros documentos, tales como los planes de Aseguramiento de la Calidad del Software.

De esta forma, una SRS apropiada limita el rango de diseños válidos pero no especifica un diseño particular.

Características de una buena SRS:

Correcta: Una SRS es correcta si, y sólo si, cada requisito declarado es un requerimiento que el software debe alcanzar.

No ambigua: Una SRS es no ambigua si, y sólo si, cada requisito declarado tiene sólo una interpretación. Esto requiere que, como mínimo, cada característica del producto final sea descripta usando un único término.

Dificultad del lenguaje natural: Los requisitos son a menudo escritos en lenguaje natural (por ejemplo, castellano) el lenguaje natural es inherentemente ambiguo. Una SRS en lenguaje natural debería ser revisada por un equipo independiente para identificar el uso ambiguo del lenguaje para que pueda corregirse.

Lenguajes de especificación de requisitos: Una manera de evitar la ambigüedad inherente al lenguaje natural es escribir la SRS en un lenguaje de especificación de requisitos particular. Sus procesadores de lenguaje detectan errores léxicos, sintácticos, y semánticos automáticamente. Una desventaja en el uso de tales lenguajes es la cantidad de tiempo requerido para aprenderlos.

Herramientas de Representación:

En general, los métodos de requisitos y lenguajes y las herramientas que los apoyan entran en tres categorías generales - objeto, proceso y comportamiento.

-Los enfoques orientados a objetos organizan los requisitos en términos de objetos del mundo real, sus atributos, y los servicios realizados por esos objetos.

- Los enfoques basados en procesos organizan los requisitos dentro de jerarquías de funciones que se comunican vía el flujo de datos.

-Los enfoques de comportamiento describen el comportamiento externo del sistema en términos de algunas nociones abstractas, (tales como cálculo de predicados), funciones matemáticas o máquinas de estado.

El grado en el cual estas herramientas y métodos pueden ser útiles para preparar una SRS dependen del tamaño y complejidad de los programas.

Completa:

Una SRS está completa si, y sólo si, incluye los siguientes elementos:

-Todos los requerimientos importantes ya sean relacionados con la funcionalidad, performance, restricciones del diseño, atributos o interfaces externas. En particular cualquier requerimiento externo impuesto por una especificación del sistema debería ser acordado y tratado.

- La definición de las respuestas del software a todos los posibles datos de entrada del sistema en todas las situaciones posibles.

- Tener todas las etiquetas llenas y referencias a todas las figuras, tablas, diagramas en la SRS y definición de todos los términos y unidades de medida.

Uso de ASDs: Cualquier SRS que usa la frase "A ser definido" (ASD) no es una SRS completa.

El ASD es, sin embargo, ocasionalmente necesario y debe acompañarse por:

-Una descripción de las condiciones que causan el ASD (por ejemplo, por qué una respuesta no es conocida) para que la situación pueda resolverse;

-Una descripción de lo que debe hacerse para eliminar el ASD que es responsable para su eliminación y por cómo debe eliminarse.

Consistente: La consistencia se refiere a la consistencia interna. Si una SRS no está de acuerdo con algún documento de alto nivel, tal como la especificación de requerimientos de sistema, entonces no es correcta.

Consistencia interna: Una SRS es internamente consistente si, y sólo si, no hay ningún subconjunto individual de la SRS que esté en conflicto.

Clasificación por importancia y/o estabilidad: Una SRS está ordenada por importancia y/o estabilidad si cada requerimiento en ella ha sido identificado para indicar tanto la importancia como la estabilidad del requerimiento particular.

Grado de estabilidad: Un método de identificar requerimientos es usar la dimensión de estabilidad. La estabilidad puede ser expresada en términos del número de cambios esperados para algún requerimiento basado en la experiencia o el conocimiento de la ocurrencia de algún evento que efectúe la organización, funciones, o la gente que es ayudada por el sistema de software.

Grado de necesidad: Otra manera de ordenar requerimientos es distinguir las clases de requisitos que hay: el esencial, el condicional y opcional.

Esenciales: Implica que el software no debería ser aceptado hasta que estos requerimientos sean provistos en la manera acordada.

Condicionales: Implica que hay requerimientos que podrían hacer más relevante al producto de software, pero no lo harían inaceptable si están ausentes.

Opcionales: Implica una clase de funciones que podrían o no ser importantes. Esto le da al proveedor la oportunidad de proponer algo que exceda la SRS

Verificable: Una SRS es verificable si, y sólo si, cada requisito establecido en ella es verificable.

Modificable: Una SRS es modificable si, y sólo si, su estructura y estilo son tales que los cambios a los requerimientos pueden ser hechos fácil, completamente y consistentemente mientras se mantiene la estructura y estilo.

Rastreable: Una SRS es rastreable si el origen de cada uno de sus requisitos está claro y si ella facilita referenciar a cada requerimiento en un futuro desarrollo o documentación.

Preparación común para la SRS

El proceso de desarrollo de software debería comenzar por el acuerdo entre proveedor y cliente acordando respecto del software completo que debe hacerse. Este acuerdo, en la forma de una SRS, debería ser preparado en forma conjunta. Esto es importante porque habitualmente ni los clientes ni los proveedores están calificados para escribir una buena SRS solos.

-Los clientes no entienden el proceso de diseño y desarrollo de software tan bien como para escribir una SRS usable.

-Los Proveedores usualmente no entienden los problemas del cliente y el campo de aplicación tan bien como para especificar requerimientos para un sistema satisfactorio.

Evolución de la SRS

La SRS puede necesitar modificarse a medida que progresa el desarrollo del producto de software.