IEEE Recomendaciones para un SRS
Sintesis para la Tesis

Alejandro Kondrasky

Índice

1. Definiciones

1. ***Contrato:*** Un documento legal acordado tanto por el consumidor como por el proveedor. Este incluye requerimientos tanto técnicos como organizacionales , costo, y la planificación del producto. Este también podrá contener datos informales pero útiles como las expectativas y compromisos por parte de ambos.

2. ***Consumidor:*** La entidad que paga por el producto y usualmente decide los requerimientos del mismo. Para este caso el consumidor sera el mismo que el proveedor.

3. ***Proveedor:*** La entidad que proveerá el producto al consumidor. Para este caso el consumidor sera el mismo que el proveedor.

4. ***Usuario:*** La entidad que interactuara con el producto. No tiene por que ser ninguna de las entidades anteriores.

2. Consideraciones para un buen SRS

2.1. Naturaleza del SRS

El SRS es la especificación para un software particular que realiza una serie de funciones en un entorno especifico. Este debería ser escrito tanto por representantes del proveedor como del consumidor.

A continuación detallaremos las siguientes características en un SRS:

1. ***Funcionalidad:*** Que es lo que debería hacer?

2. ***Interfases externas:*** Como interactúa con el usuario, hardware y otros software?

3. ***Performance:*** Cual es la velocidad, accesibilidad, tiempos de respuesta y de recuperación,

etc; para las funciones que dicho software realiza ?

4. ***Atributos:*** Que consideraciones debemos tomar en cuanto a portabilidad, correctitud, mantenibilidad, seguridad, etc ?

5. ***Restricciones de diseño e implementación:*** Cuales son los estándares , lenguajes, políticas, recursos máximos, entornos operativos, etc?

2.2. Entorno del SRS

El software podrá ser auto contenido o parte de un sistema mayor. En el segundo caso deberemos tener en cuenta cuales van a ser las interfaces requeridas, performance externa y requerimientos funcionales para con dicho software.

Como el SRS tiene un rol específico en el proceso de desarrollo del software, debe cuidarse de no ir mas allá de los limites de dicho rol.

Esto significa que el SRS :

- Definir correctamente todos los requerimientos de software. Estos existirán por la naturaleza de la tarea a resolver o por características especiales del proyecto.

- No debería describir ningún detalle de diseño u implementación.

- No debería agregar restricciones adicionales al software.

Entonces un SRS bien escrito solo limita el rango de diseños validos, pero no especifica ningún en particular.

2.3. Características de un buen SRS

2.3.1. Correctitud

Es correcto sii cada requerimiento declarado debería ser satisfecho por el software.

No existen herramientas para asegurarlo. EL SRS debería ser comparado con cualquier especificación aplicable como otros SRS otros estándares aplicables.

2.3.2. Exactitud

Sera exacto sii cada requerimiento estipulado tiene una sola interpretación. Como mínimo, requiere que cada característica del producto sea descripta con un solo termino. En el caso de que un termino pueda tener múltiples significados en un contexto especifico, este deberá tener un glosario donde se especifica mas detalladamente su significado.

2.3.3. Completitud

Es completo sii incluye los siguientes elementos :

1. Todo requerimiento significante como funcionalidad, atributos, performance, restricciones de diseño y interfases externas.

2. Definición de la respuesta del software a todas las clases de entradas realizables en todas las situaciones posibles, tanto para entradas validas como invalidas[1].

3. Etiquetas y referencias de todas las figuras, tablas y diagramas en el SRS y definición de todos los términos y unidades de medida.

2.3.3.1. Uso de TBD[2] Cualquier SRS que contenga TBDs no esta terminado. Pueden ser necesarios pero deben estar siempre acompañados por :

1. Una descripción de la causa del mismo[3] para así saber como resolverla.

2. Una descripción de lo que se debe hacer para eliminarlo, quien deberá hacerlo y cuando.

2.3.4. Consistencia

2.3.4.1. Consistencia Interna Es consistente internamente sii ningún subconjunto de requerimientos individuales entran en conflicto.

Existen 3 tipos de situaciones por las que pueden haber conflictos:

1. Características especificadas de un objeto del mundo real podrían conflictuar, como por ejemplo la forma de presentar un resultado u otros similares.

[1] Especialmente las Invalidas!!!
[2] To be Determinated.
[3] "Por que no se sabe la respuesta".

2. Conflictos de tipo lógico o temporal entre 2 acciones especificas.

3. Dos o mas requisitos describen a un mismo objeto de diferente forma. El uso de terminología estándar y definiciones promueve la consistencia.

2.3.4.2. Ordenarlos por importancia Se esta ordenado por importancia si cada requerimiento tiene un identificador que indica la importancia del mismo. Existen los siguientes grados de importancia:

1. *Esencial:* El software no sera aceptado si no cumple con dicho requerimiento de la manera acordada.

2. *Condicional:* Son requerimientos que mejorarían al software pero no se requieren para que este sea aceptado.

3. *Opcionales:* Cosas extra que solo hacedlas si te sobra tiempo.

2.3.5. Verificabilidad

Es verificable sii cada requerimiento lo es, los cuales lo son sii existe un proceso finito que lo chequee. Generalmente los ambiguos no son verificables. Requerimientos descriptos con adjetivos subjetivos son no verificables, tampoco los que requieran de plazos de tiempo o recursos excesivos para el desarrollo de dicho software.

En caso de no poder ser verificable , debería ser removido o redefinido para que lo sea.

2.3.6. Modificabilidad

Es modificable sii su estructura y estilo son tales que es fácil realizar cualquier cambio en los requerimientos en forma completa y consistente, manteniendo la estructura y estilo. Esta requiere de :

1. Tener una organización coherente y fácil de usar con tablas de contenido, índices y explicitas referencias cruzadas.

2. No haya redundancias (no repetición de requerimientos).

3. Expresar cada requerimiento por separado antes de mezclarlos con otros[4].

2.3.7. Rastreabilidad

Es rastreable si el origen de cada requerimiento es claro y facilita el referenciamiento de cada requerimiento en el futuro desarrollo del documento. Dos tipos de rastreamiento son recomendados:

1. *Backwards:* Depende de que cada requerimiento se refiera explícitamente a su fuente en los documentos iniciales.

2. *Forward:* Esto depende de que cada requerimiento tenga un nombre o nro único.

2.4. Evolución del SRS

Estos documentos evolucionan a medida de que se desarrolla el producto, por lo que suele ser imposible planificar ciertos detalles al inicio del proyecto. Por esta razón se deben tener las siguientes consideraciones:

1. Cada requerimiento debe ser especificado completamente tal como son conocidos en dicho momento, aun que se sabe inevitable la necesidad de una revisión. El hecho de que están incompletos debería ser recalcado

2. Un proceso formal debería iniciarse para identificar, controlar, rastrear y reportar los cambios del proyecto. Aprobados dichos cambios deberían ser incorporados en el SRS de manera tal que :

   a) Provea una exacta y completa reseña de los cambios.

   b) Permita revisar la actual y nueva versión del SRS.

3. Partes de un SRS

3.1. Introducción

Esta debería proveer un resumen del SRS entero. Debería contener las siguientes secciones:

[4] Usar referencias cruzadas para esto... Podrías armarte un grafo (que debería ser un arbol!) muestre como se entrecruzan los requerimientos.

3.1.1. Propósito

Debe:

1. Delineamientos de su propósito.
2. Audiencia a la que va dirigido.

3.1.2. Alcance

Debe:

1. Identificar el nombre del tipo de producto.
2. Explicar a grandes rasgos que hará y, si es necesario, que no.
3. Describir las aplicaciones de dicho software, incluyendo beneficios relevantes, objetivos y metas.

3.1.3. Definiciones, anacronismos y abreviaturas

Esta sección debe proveer las definiciones de todos los términos, anacronismos y abreviaturas requeridas para interpretar correctamente el SRS. Esta información puede ser proveída mediante varios apéndices o haciendo referencia a otros documentos.

3.1.4. Referencias

Debe:

1. Proveer una lista completa de todos los documentos referidos en el SRS.
2. Identificarlos con titulo, nro de reporte[5], fecha y organización que lo publico.
3. Especificar las fuentes de donde fueron obtenidas las referencias.

Esta información puede ser proveída mediante un apéndice o haciendo referencia a otro documento.

3.1.5. Resumen

Debe:

1. Describir que es lo que contiene el resto del SRS.
2. Explicar como este esta organizado.

[5] Si se aplica

3.2. Descripción General

Esta sección debe describir los factores generales que afectan el producto y sus requerimientos. Esta sección no contiene requerimientos específicos, sino que provee un pantallazo general de dichos requerimientos y facilita su entendimiento.

3.2.1. Perspectiva del Producto

Esta sección se encarga de comparar el producto con otros similares. Aclare si este es independiente y auto contenido[6]. Tiene las siguientes secciones :

3.2.1.1. Interfaces de Usuario Debe especificar lo siguiente:

1. *Las características lógicas de cada interfase entre el usuario y el producto.* Esta incluye la configuraciones necesarias para cumplir con los requerimientos de software.
2. *Todos los aspectos de optimización de la interfase.* Esta puede ser simplemente una lista de como y como no el sistema aparecerá ante el usuario[7]. Como otros, estos requerimientos deberán ser verificables.

3.2.1.2. Interfaces de Software Debe especificar que otro software(e.g., bases de datos, sistemas operativos, etc)e interfases con otras aplicaciones. Para cada uno de estos se debe proveer:

- Nombre.
- Descripción.
- Nro de especificación.
- Nro de versión.
- Fuente.

Para las interfases se debe proveer:

- Discutir el propósito de interconectar el producto con dicho software.
- Definir la interfase en términos del contenido de los mensajes y formato. En los casos en que esta este bien documentada, con una referencia a la documentación alcanza.

[6] En caso de no serlo lea el IEEE en ingles :P
[7] Por ejemplo verbose o quiet

3.2.1.3. Restricciones de Memoria Especificar cualquier restricción o característica a tener en cuenta tanto para memorias primarias como secundarias.

3.2.1.4. Funciones del producto Esta debe proveer un sumario de las funciones principales del producto.