IEEE-STD-830-1998 : ESPECIFICACIONES DE LOS REQUISITOS DEL SOFTWARE

1. Definiciones

En general las definiciones de los términos usados en estas especificaciones están conforme a las definiciones proporcionadas en IEEE Std 610.12-1990.

1.1 Contrato:

Un documento es legalmente obligatorio y en el estarán de acuerdo las partes del cliente y proveedor. Esto incluye los requisitos técnicos y requerimientos de la organización, costo y tiempo para un producto. Un contrato también puede contener la información informal pero útil como los compromisos o expectativas de las partes involucradas.

1.2 Cliente:

La persona (s) que pagan por el producto y normalmente (pero no necesariamente) definen los requisitos. En la práctica el cliente y el proveedor pueden ser miembros de la misma organización.

1.3 Proveedor:

La persona (s) que producen un producto para un cliente.

1.4 Usuario:

La persona (s) que operan o actúan recíprocamente directamente con el producto. El usuario (s) y el cliente (s) no es (son) a menudo las mismas persona(s).

2. Las consideraciones para producir un buen SRS.

Estas cláusulas proporcionan información a fondo que deben ser consideradas al momento de producir un SRS. Esto incluye lo siguiente:

a) la Naturaleza del SRS;
b) el Ambiente del SRS;
c) las Características de un buen SRS ;
d) la preparación de los Joins del SRS;
e) la evolución de SRS;
f) Prototipos;
g) Generando el diseño en el SRS;
h) Generando los requisitos del proyecto en el SRS.

2.1 Naturaleza del SRS

El SRS son especificaciones para un producto del software en particular, programa, o juego de programas que realizan ciertas funciones en un ambiente específico. El SRS puede escribirse por uno o más representantes del proveedor, uno o más representantes del cliente, o por ambos. La Subclausula 2.4 recomienda ambos.

Los problemas básicos que se presentan al escribir un SRS van dirigidos a lo siguiente:

a) La Funcionalidad.

¿Qué se supone va hacer el software ?

b) Las interfaces Externas.

¿Cómo el software actúa recíprocamente con las personas, el hardware de los sistemas, otro hardware, y otro software?

c) La Actuación.

¿Cuál es la velocidad, la disponibilidad, tiempo de la contestación, tiempo de la recuperación de varias funciones del software, etc.?

d) Los Atributos.

¿Qué portabilidad tiene, exactitud, el mantenimiento, la seguridad, las consideraciones etc.?

e) Las restricciones del diseño que impusieron en una aplicación.

¿Hay algún requerimiento Standard, idioma de aplicación, las políticas para la integridad del banco de datos, los límites de los recursos, operando en que ambiente (s) etc.?

2.2 Ambiente del SRS

Es importante considerar la parte que el SRS representa en el diseño del proyecto total que se define en IEEE Std 610.12-1990. El software puede contener toda la funcionalidad del proyecto esencialmente o puede ser parte de un sistema más grande. En el último caso habrá un SRS que declarará las interfaces entre el sistema y su software modular, y pondrá que función externa y requisitos de funcionalidad tiene con el software modular.

Otras normas, relacionan a otras partes del ciclo de vida de software para que pueda complementar los requisitos del software.

Desde que el SRS tiene un papel específico en el proceso de desarrollo de software, el que define el SRS debe tener el cuidado para no ir más allá de los límites de ese papel. Esto significa que:

a) 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.

b) no debe describir cualquier plan o detalles de aplicación. Éstos deben describirse en la fase del diseño del proyecto.

c) no debe imponer las restricciones adicionales en el software. Éstos se especifican propiamente en otros documentos.

2.3 Características de un buen SRS.

Un SRS debe ser:

a) Correcto;
b) Inequívoco;
c) Completo;
d) Consistente;
e) Delinear que tiene importancia y/o estabilidad;
f) Comprobable;
g) Modificable;
h) Identificable.

2.3.1 Correcto

Un SRS es correcto si, y sólo si, cada requisito declarado se encuentra en el software. No hay ninguna herramienta o procedimiento que aseguran la exactitud. Alternativamente el cliente o el usuario pueden determinar si el SRS refleja las necesidades reales correctamente. Identificando los requerimientos hace este procedimiento más fácil y hay menos probabilidad al error.

2.3.2 Inequívoco

Un SRS es inequívoco si, y sólo si, cada requisito declarado tiene sólo una interpretación. Como un mínimo, se requiere que cada característica de la última versión del producto se describa usando un único término.

En casos dónde un término en un contexto particular tenga significados múltiples, el término debe ser incluido en un glosario dónde su significado es hecho más específico.

Un SRS es una parte importante del proceso de requisitos del ciclo de vida de software y se usa en el diseño, aplicación, supervisión, comprobación, aprobación y pruebas como está descrito en IEEE Std 1074-1997.

El SRS debe ser inequívoco para aquéllos que lo crean y para aquéllos que lo usan. Sin embargo, estos grupos no tienen a menudo el mismo fondo y por consiguiente no tienden a describir los requisitos del software de la misma manera.

Las Subclauses 2.3.2.1 a través de 2.3.2.3 recomiendan cómo evitar la ambigüedad.

2.3.2.1 Trampas del idioma natural.

Los requisitos son a menudo escritos en el idioma natural (por ejemplo, inglés) el idioma natural es inherentemente ambiguo. Un idioma natural que SRS podría ser revisado por una parte independiente para identificar el uso ambiguo del idioma para que pueda corregirse.

2.3.2.2 Idiomas de especificación de requisitos

Una manera de evitar la ambigüedad inherente en el idioma natural es escribir el SRS en un idioma de especificación de requisitos particular. Sus procesadores del idioma descubren muchos errores léxicos, sintácticos, y semánticos automáticamente.
Una desventaja en el uso de tales idiomas es que la premura de tiempo exigió aprenderlos. También, muchos usuarios no-técnicos los encuentran ininteligible. Es más, estos idiomas tienden a ser buenos a expresar ciertos tipos de requisitos y dirigirse a ciertos tipos de sistemas. Así, ellos pueden influir en los requisitos de las maneras sutiles.

2.3.2.3 Representación hecha con herramientas

En general, los métodos de requisitos e idiomas y las herramientas que los apoyan entran en tres categorías generales - el objeto, procesos y conductual.
El término objetos-orientados organizan los requisitos en lo que se refiere a los objetos en el mundo real , sus atributos, y los servicios realizados por esos objetos.
El término procesos organizan los requisitos en las jerarquías de funciones que comunican vía el flujo de datos.
El términos conductuales describen conducta externa del sistema por lo que se refiere a alguna noción de lo abstracto, las funciones matemáticas o el estado de las máquinas.

El grado en que se usan estas herramientas y los métodos pueden ser útiles preparando un SRS pero depende del tamaño y complejidad del programa. Aún usando cualquiera de estos términos es mejor retener las descripciones del idioma natural. Así, clientes poco familiar con las anotaciones el SRS puede entender todavía.

2.3.3 Completo

Un SRS está completo si, y sólo si, incluye los elementos siguientes:

a) Los requisitos están relacionados a la funcionalidad, el desarrollo, las restricciones del diseño, los atributos y las interfaces externas. En particular debe reconocerse cualquier requisito externo impuesto por una especificación del sistema y debe tratarse.

b) La definición de las respuestas del software a todos los posibles datos de la entrada del sistema y a toda clase de situaciones. Una nota que es importante especificar son las contestaciones a las entradas válidas e inválidas a ciertos valores.

c) Tener todas las etiquetas llenas y referencias a todas las figuras, tablas, diagramas en el SRS y definición de todas las condiciones y unidades de medida.

2.3.3.1 Uso de TBDs

Cualquier SRS que usa la frase "para ser determinado" (TBD) no es un SRS completo. El TBD es, sin embargo, ocasionalmente necesario y debe acompañarse por:

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

b) Una descripción de lo que debe hacerse para eliminar el TBD que es responsable para su eliminación y por como debe eliminarse.

2.3.4 Consistente

La consistencia se refiere a la consistencia interior. Si un SRS no está de acuerdo con algún documento del superior-nivel, como una especificación de requisitos de sistema, entonces no es correcto.

2.3.4.1 Consistencia interior

Un SRS es internamente consistente si, y sólo si, ningún subconjunto de requisitos individuales genero conflicto en él.
Los tres tipos de conflictos probables en un SRS son:

a) Las características especificadas en el mundo real de los objetos pueden chocar. Por ejemplo,
1) el formato de un informe del rendimiento puede describirse en un requisito como tabular pero en otro como textual.

2) un requisito puede declarar que todo las luces serán verdes mientras otro puede declarar que todo las luces sean azules.

b) puede haber conflicto lógico o temporal entre dos acciones especificadas. Por ejemplo,
1) un requisito puede especificar que el programa sumará dos entradas y otro puede especificar que el programa los multiplicará.

2) un requisito puede declarar que "A" siempre debe seguir "B", mientras otro puede requerir que "A"y B" ocurran simultáneamente.

c) Dos o más requisitos pueden describir el mismo mundo real del objeto pero uso las condiciones diferentes para ese objeto. Por ejemplo, una demanda del programa para una entrada del usuario puede llamarse una "sugerencia" en un requisito y una "señal" en otro. El uso de terminología normal y definiciones promueve la consistencia.

2.3.5 Delinear que tiene importancia y/o estabilidad

Un SRS debe delinear la importancia y/o estabilidad si cada requisito en él tiene un identificador para indicar la importancia o estabilidad de ese requisito en particular.
Típicamente, todos los requisitos que relacionan a un producto del software no son igualmente importantes. Algunos requisitos pueden ser esenciales, sobre todo para las aplicaciones de vida crítica, mientras otros pueden ser deseables.

Cada requisito en el SRS debe identificarse para representar estas diferencias, aclarar y ser explícito. Identificando los requisitos de la manera siguientes:

a) Tienen los clientes que dar las consideraciones muy cuidadosamente a cada requisito para que se clarifique cualquier omisión que ellos pueden tener.

b) Tener diseñadores que hagan diseños correctos y pongan el mismo esfuerzo en todos los niveles del producto del software.

2.3.5.1 Grado de estabilidad

Puede expresarse la estabilidad por lo que se refiere al número de cambios esperados a cualquier requisito basado en experiencia o conocimiento de eventos venideros que afectan la organización, funciones y a las personas que apoyan el sistema del software.

2.3.5.2 Grado de necesidad

Otra manera de alinear los requisitos es distinguir las clases de requisitos que hay: el esencial, el condicional y optativo.

a) Esencial.

Implica que el software no será aceptable a menos que estos requisitos se proporcionan de una manera convenida.

b) el Condicional.

Implica que éstos son requisitos que reforzarían el producto del software, pero no lo haría inaceptable si ellos están ausentes.

c) Optativo.

Implica una clase de funciones que pueden o no pueden valer la pena. Esto le da la oportunidad de proponer algo que excede el SRS al proveedor.

2.3.6 Comprobable

Un SRS es comprobable si, y sólo si, cada requisito declarado es comprobable. Un requisito es comprobable si, y sólo si, allí existe algún proceso rentable finito con que una persona o la máquina puede verificar que el producto del software reúne el requisito. En general cualquier requisito ambiguo no es comprobable.

Los requisitos de No-verificable incluyen las declaraciones como "trabaja bien", "interface humana buena" y "normalmente pasará" no pueden verificarse los requisitos de esos porque es imposible de definir las condiciones "bueno," "bien" o "normalmente". La declaración que "el programa nunca entrará en una vuelta infinita" es el no-verificable porque la comprobación de esta calidad es teóricamente imposible.

Un ejemplo de una declaración comprobable es:

El rendimiento del programa se producirá dentro de 20 seg de evento 60% del tiempo; y se producirá dentro de 30 seg de evento 100% del tiempo.

Esta declaración puede verificarse porque usa condiciones concretas y las cantidades mensurables.

Si un método no puede inventarse para determinar si el software reúne un requisito particular, entonces ese requisito debe quitarse o debe revisarse.

2.3.7 Modificable

Un SRS es modificable si, y sólo si, su estructura y estilo son tales que puede hacerse cualquier cambio a los requisitos fácilmente, completamente y de forma consistente mientras conserva la estructura y estilo. Para que sea modificable se requiere un SRS que contenga:

a) Tiene un coherente y fácil de usar en la organización de volúmenes de información, un índice y las referencias cruzadas explícitas;

b) no sea redundante (es decir, el mismo requisito no debe aparecer en más de un lugar en el SRS);

c) Exprese cada requisito separadamente, en lugar de intercarlarlas con otros requisitos.
La redundancia no es un error, pero puede llevar fácilmente a los errores. La redundancia puede ayudar hacer un SRS más leíble de vez en cuando, pero un problema puede generarse cuando el documento redundante se actualiza. Por ejemplo, un requisito puede alterarse en un solo lugar dónde aparece. El SRS se pone incoherente entonces. Siempre que la redundancia sea necesaria, el SRS debe incluir la cruz explícita - las referencias para hacerlo modificable.

2.3.8 Identificable

Un SRS es identificable si el origen de cada uno de sus requisitos está claro y si facilita las referencias de cada requisito en el desarrollo futuro o documentación del mismo. Lo siguiente que se recomiendan dos tipos de identificabilidad:

a) el identificable dirigido hacia atrás (es decir, a las fases anteriores de desarrollo).
Esto depende explícitamente en cada requisito la referencias de su fuente en los documentos más antiguos.

b) el identificable delantero (es decir, a todos los documentos desovados por el SRS).
Esto depende en cada requisito en el SRS que tiene un único nombre o número de la referencia. El identificable delantero del SRS es especialmente importante cuando el producto del software entra en el funcionamiento y fase de mantenimiento. Como el código y documentos del plan se modifican, es esencial poder determinar el juego completo de requisitos que pueden afectarse por esas modificaciones.

2.4 Preparación de los JOIN del SRS

El proceso de desarrollo de software debe empezar con el proveedor y con el acuerdo del cliente en lo que el software completado debe hacer. Este acuerdo, en la forma de un SRS, debe prepararse juntamente. Esto es importante porque ni el cliente ni el proveedor son calificables para escribir exclusivamente un buen SRS.

a) Clientes normalmente no entienden bien el diseño del software y proceso de desarrollo bastante bien como para escribir un SRS utilizable.

b) Los Proveedores normalmente no entienden bien el problema de los clientes y campo de acción bastante bien como para que especifique los requisitos para un sistema satisfactorio.

Por consiguiente, el cliente y el proveedor deben trabajar para producir juntos un buen escrito y completamente entendible SRS.

Una situación especial existe cuando el sistema y su software los dos se están definiéndose concurrentemente. Entonces la funcionalidad, interfaces, desarrollo y otros atributos, restricciones del software no son los predefinidos, sino se definen juntamente y están sujetos a la negociación y al cambio. Esto lo hace más difícil, pero no menos importante, para encontrar las características declaradas en 2.3. en particular, un SRS que no obedece los requisitos de su especificación de sistema de padre es incorrecto.

2.5 Evolución de SRS

El SRS puede necesitar evolucionar así como el desarrollo de las actualizaciones del producto de software. Puede ser imposible de especificar un poco a detalle en el momento que el proyecto se inicia (por ejemplo, puede ser imposible de definir toda la estructura de la pantalla para un programa interactivo durante la fase de requisitos). Los cambios adicionales pueden suceder según como las deficiencias se vayan descubriendo, las limitaciones e inexactitudes en el SRS.

Dos consideraciones en este proceso son las siguientes:

a) Deben especificarse los requisitos completamente como se es conocido en el momento, aun cuando las revisiones evolutivas pueden preverse como inevitable. El hecho que ellos están incompletos debe ser anotado.

b) Un proceso de cambio formal debe comenzarse para identificarse, el control, dejar huella e informe de lo que proyectaron los cambios.

Los cambios aprobados en los requisitos deben incorporarse en el SRS de semejante manera acerca de que:

1) proporcione un lineamiento de la auditoria exacta y completa de cambios;
2) el permiso de la revisión actual y reemplazó de los cambios en en SRS.

2.6 Prototipos.

Los prototipos frecuentemente se usan durante una fase de los requisitos de un proyecto. Muchas herramientas existen para generar un prototipo para exhibir algunas características de un sistema, ser creado muy rápidamente y fácilmente.

Los prototipos son útiles por las razones siguientes:

a) El cliente puede ver el prototipo y reaccionar a él que leer el SRS y reaccionar a él. Así, el prototipo proporciona la regeneración rápida.

b) El prototipo despliega aspectos de anticiparse a la conducta de los sistemas. Así, no sólo produce las respuestas sino también las nuevas preguntas. Esto ayuda a ver el alcance en el SRS.

c) Un SRS basado en un prototipo tiende a sufrir menos cambios durante el desarrollo, así se acorta el tiempo de desarrollo.

Un prototipo debe usarse como una manera de sacar los requisitos del software. Pueden extraerse algunas características como pantalla o formatos del reporte directamente del prototipo. Otros requisitos pueden ser inferidos ejecutando los experimentos con el prototipo.

2.7 Generando el diseño en el SRS

Un requisito especifica una función externa visible o atributo de un sistema. Un diseño describe un subcomponente particular de un sistema y/o sus interfaces con otros subcomponentes. El diseñador del SRS debe distinguir claramente entre identificar las restricciones del diseño requeridos y proyectar un plan específico. La nota es que cada requisito en el SRS limita las alternativas del plan. Esto no significa, sin embargo, que cada requisito es el plan.

El SRS debe especificar qué funciones serán realizadas, con qué datos, para producir qué resultados, en qué situación y para quien. El SRS se debe enfocar en los servicios a ser realizados. El SRS normalmente no debe especificar los puntos del plan como lo siguiente:

a) Partir el software en módulos;
b) Asignando las funciones a los módulos;
c) Describiendo el flujo de información o controles entre los módulos;
d) Escogiendo las estructuras de los datos.

2.7.1 Requisitos del plan necesarios

En casos especiales algunos requisitos pueden restringir el plan severamente. Por ejemplo, seguridad o requisitos de seguridad pueden reflejarse directamente en el plan como la necesidad a:

a) Guarde ciertas funciones en los módulos separadamente;
b) El permiso sólo limitó la comunicación entre algunas áreas del programa;
c) La integridad de datos mediante chequeos para las variables críticas.

Los ejemplos de restricciones del diseño válidos son requisitos físicos, requisitos del desarrollo, normas de desarrollo de software y software de calidad según los standares.
Por consiguiente, los requisitos deben declararse de un punto de vista completamente externo. Al usar a modelos para ilustrar los requisitos, recuerda que el modelo sólo indica la conducta externa, y no especifica un plan.

2.8 Requisitos del proyecto generados en el SRS

El SRS debe dirigir el producto del software, no el proceso de producir el producto del software.
Los requisitos del proyecto representan una comprensión entre el cliente y el proveedor sobre materias contractuales que pertenecen a la producción de software y así no deben ser incluidos en el SRS. Éstos normalmente incluyen los puntos como:

a) el Costo;
b) Los tiempos de la entrega;
c) Informando los procedimientos;
d) Los métodos de desarrollo de Software;
e) La convicción de Calidad;
f) La Aprobación y criterio de la comprobación;
g) Los procedimientos de aceptación.

Se especifican los requisitos del proyecto en otros documentos, generalmente en un plan de desarrollo de software, un software de calidad o una declaración de trabajo.

3. Las partes de un SRS

Estas partes se colocan en Figura 1 en un contorno que puede servir como un ejemplo por escribir un SRS.
Un SRS no tiene que seguir este contorno o usar los nombres dado aquí para sus partes, un buen SRS debe incluir toda la información que se mencionó aquí.

Tabla de Contenidos
1. Introducción
   1.1 Propósito
   1.2 Alcance
   1.3 Definiciones, siglas, y abreviaciones
   1.4 Referencias
   1.5 Apreciación global
2. Descripción global
   2.1 Perspectiva del producto
   2.2 Funciones del producto
   2.3 Características del usuario
   2.4 Restricciones
   2.5 Atención y dependencias
3. Los requisitos específicos (Vea del 3.3.1 al de 3.3.8)
Apéndices
Indice

Figure 1 - el Prototipo el contorno de SRS

3.1 Introducción (Sección 1 del SRS)

La introducción del SRS debe proporcionar una apreciación global del SRS completo. Debe contener las subdivisiones siguientes:

a) el Propósito;
b) el Alcance;
c) las Definiciones, siglas, y abreviaciones;
d) las Referencias;
e) la Apreciación global.

3.1.1 Propósito (1.1 del SRS)

Esta subdivisión debe:

a) Delinear el propósito del SRS;
b) Especifique a que público intencional va dirigido el SRS.

3.1.2 Alcance (1.2 del SRS)

Esta subdivisión debe:

a) Identifique el producto (s) del software para ser diseñado por el nombre (por ejemplo, Anfitrión DBMS, el Generador del Reporte, etc.);

b) Explique eso que el producto (s) del software que hará y que no hará.

c) Describe la aplicación del software especificándose los beneficios pertinentes, objetivos, y metas;

d) Sea consistente con las declaraciones similares en las especificaciones de niveles superiores (por ejemplo, las especificaciones de los requisitos del sistema), si ellos existen.

3.1.3 Definiciones, siglas, y abreviaciones (1.3 del SRS)

Esta subdivisión debe proporcionar las definiciones de todas las condiciones, las siglas, y abreviaciones que exigen interpretar el SRS propiamente. Esta información puede proporcionarse por la referencia a uno o más apéndices en el SRS o por la referencia a otros documentos.

3.1.4 Referencias (1.4 del SRS)

Esta subdivisión debe:

a) Proporcione una lista completa de todas las referencias de los documentos en otra parte en el SRS;

b) Identifique cada documento por el título, número del reporte (si es aplicable), fecha, y publicación de la organización;

c) Especifique las fuentes de las referencias de donde se obtuvieron.

Esta información puede proporcionarse por la referencia a un apéndice o a otro documento.

3.1.5 Apreciación global (1.5 del SRS)

Esta subdivisión debe:

a) Describa lo que el resto del SRS contiene;
b) Explica cómo el SRS es organizado.

3.2 Descripción global (Sección 2 del SRS)

Esta sección del SRS debe describir los factores generales que afectan el producto y sus requisitos. Esta sección no declara los requisitos específicos. En cambio, mantiene un fondode esos requisitos que se definen en detalle en Sección 3 del SRS y les hacen más fácil entender.
Esta sección normalmente consiste en seis subdivisiones, como sigue:

a) la perspectiva del Producto;
b) las funciones del Producto;
c) las características del Usuario;
d) las restricciones;
e) las Asunciones y dependencias;
f) Prorrateando de requisitos.

3.2.1 Perspectiva del producto (2.1 del SRS)

Esta subdivisión del SRS debe poner el producto en la perspectiva con otros productos relacionados. Si el producto es independiente y totalmente autónomo, debe declararse que así es. Si el SRS define un producto que es un componente de un sistema más grande, como frecuentemente ocurre, entonces esta subdivisión debe relacionar los requisitos de ese sistema más grande a la funcionalidad del software y debe identificar las interfaces entre ese sistema y el software.

Un diagrama del bloque que muestra los componentes mayores del sistema más grande, las interconexiones, y las interfaces externas pueden ser útiles.

Esta subdivisión también debe describir cómo el software opera dentro de las varias restricciones. Por ejemplo, estos restricciones podrían incluir:

a) las interfaces del Sistema;
b) las interfaces del Usuario;
c) las interfaces del Hardware;
d) las interfaces del Software;
e) las interfaces de Comunicaciones;
f) la Memoria;
g) los Funcionamientos;
h) los requisitos de adaptación del Site.

3.2.1.1 Interfaces del sistema.

Esto debe listar cada interfaz del sistema y debe identificar la funcionalidad del software para lograr el requisito del sistema y la descripción de la interfaz para empatar el sistema.

3.2.1.2 Interfaces con el usuario.

Esto debe especificar a lo siguiente:

a) Las características lógicas de cada interfaz entre el producto del software y sus usuarios.
Esto incluye las características de la configuración (por ejemplo, formatos de la pantalla requeridos, página o esquemas de la ventana, los reportes o menús o disponibilidad de llaves de la función programables) necesario para lograr los requisitos del software.

b) Todos los aspectos para perfeccionar la interfaz con la persona que debe usar el sistema.
Esto puede comprender una lista de lo que hace y no hace simplemente delante de cómo el sistema aparecerá al usuario. Un ejemplo puede ser un requisito para la opción de mensajes de error largos o cortos. Como todos, estos requisitos deben ser comprobables, debe especificarse en los Atributos de Sistema de Software bajo una sección tituló Facilidad de Uso.

3.2.1.3 Interfaces con el hardware.

Esto debe especificar las características lógicas de cada interfaz entre el producto del software y los componentes del hardware del sistema. Esto incluye las características de la configuración (el número de puertos, la instrucción set, etc.), también cubre como qué dispositivos serán apoyados, cómo ellos serán apoyados y protocolos. Por ejemplo, el apoyo de las terminales puede especificarse cuando tienen full-screen.

3.2.1.4 Interfaces con el software.

Esto debe especificar el uso de otros productos del software requeridos (por ejemplo, un sistema de dirección de datos, un sistema operativo o un paquete matemático) e interfaces con otros sistemas de la aplicación (por ejemplo, la unión entre el Sistema de Cuentas, el Sistema por Cobrar y un Sistema del Mayor General). Para cada uno el producto del software requirió proporcionarse:

- El nombre;
- El código mnemotécnico;
- El número de la especificación;
- El número de la versión;
- La fuente.

Para cada interfaz, lo siguiente debe proporcionarse:
- La discusión del propósito de la interfaz del software en relación con el producto del software.

- La definición de la interfaz por lo que se refiere a los mensajes contenidos y formatos. No es necesario detallar cualquiera bien la documentación de la interfaz, pero una referencia al documento que define la interfaz se requiere.

3.2.1.5 Interfaces de comunicaciones

Esto debe especificar las varias interfaces a las comunicaciones como los protocolos de las redes locales, etc.,

3.2.1.6 Restricciones de memoria

Esto debe especificar cualquier característica aplicable y límites en la memoria primaria y la memoria secundaria.

3.2.1.7 Funcionamientos

Esto debe especificar los funcionamientos normales y especiales requeridos por el usuario como:

a) Los varios modos de funcionamientos en la organización del usuario (por ejemplo, los funcionamientos de iniciar el usuario);

b) los Periodo de funcionamientos interactivos y periodo de funcionamientos desatendido;

c) Datos que procesan las funciones de apoyo;

d) el Apoyo y funcionamientos de la recuperación.

La NOTA - Esto a veces se especifica como la parte del User Interfaces Sectión.

3.2.1.8 Requisitos de adaptación del Site.

Esto debe:

a) Defina los requisitos para cualquier dato o la secuencia de inicialización que son específico a un sitio dado, la misión o el modo operacional (por ejemplo, los límites de seguridad, etc.);

b) Especifique el sitio o los rasgos que se deben relacionar que deben modificarse para adaptar el software a una instalación particular.

3.2.2 Funciones del Producto (2.2 del SRS)

Esta subdivisión del SRS debe proporcionar un resumen de las funciones mayores que el software realizará.
Por ejemplo, un SRS para un programa de contabilidad puede acostumbrar esta parte a dirigirse al mantenimiento de Cuenta de Cliente, declaración del cliente y preparación

de la factura sin mencionar la inmensa cantidad de detalle que cada uno de esas funciones requiere.
A veces el resumen de la función que es necesario para esta parte puede tomarse directamente de la sección de la especificación en el nivel superior (si uno existe) eso asigna las funciones particulares al producto del software. Note que eso es por causa de la claridad.

a) Las funciones deben organizarse en cierto modo eso hace la lista de funciones entendible al cliente o a cualquiera nada más leyendo el documento la primera vez.

b) Pueden usarse los métodos Textuales o gráficos para mostrar las funciones diferentes y sus relaciones.
No se piensa que el diagrama muestra un diseño de un producto, sino simplemente muestra la relación lógica entre las variables.

3.2.3 Características del usuario (2.3 del SRS)

Esta subdivisión del SRS debe describir esas características generales de los usuarios intencionales del producto que incluye nivel educativo, experiencia, y la especialización técnica.

3.2.4 Restricciones (2.4 del SRS)

Esta subdivisión del SRS debe proporcionar una descripción general de cualquier otro punto que limitará las opciones de los diseñadores. Éstos incluyen:

a) las políticas reguladoras;
b) las limitaciones del Hardware;
c) las Interfaces a otras aplicaciones;
d) el funcionamiento Paralelo;
e) las funciones de la Auditoría;
f) las funciones de Control;
g) los requisitos de lenguaje;
h) los protocolos Señalados (por ejemplo, XON-XOFF, ACK-NACK);
i) los requisitos de Fiabilidad;
j) Credibilidad de la aplicación;
k) la Seguridad y consideraciones de seguridad.

3.2.5 Atenciones y dependencias (2.5 del SRS)

Esta subdivisión del SRS debe listar cada uno de los factores que afectan los requisitos declarados en el SRS.
Estos factores no son las restricciones del diseño en el software pero son, más bien, cualquier cambio a ellos eso puede afectar los requisitos en el SRS. Por ejemplo, una suposición puede ser que un sistema operativo específico estará disponible en el hardware designado para el producto del software. Si, de hecho, el sistema operativo no está disponible, los SRS tendrían que cambiar de acuerdo con entonces.

3.2.6 Prorratear los requisitos (2.6 del SRS)

Esta subdivisión del SRS debe identificar requisitos que pueden tardarse hasta las versiones futuras del sistema.

3.3 Requisitos específicos (Sección 3 del SRS)

Esta sección del SRS debe contener todos los requisitos del software a un nivel de detalle suficiente para permitirles a los diseñadores diseñar un sistema para satisfacer esos requisitos, y a los auditores a probar que el sistema satisface esos requisitos. A lo largo de esta sección, cada requisito declarado debe ser externamente perceptible por los usuarios, operadores u otros sistemas externos. Estos requisitos deben incluir por lo menos una descripción de cada entrada (el estímulo) en el sistema, cada salida (la contestación) del sistema, y todas las funciones realizadas por el sistema en la salida a una entrada o en el apoyo de la salida. Esta es la parte más grande y más importante del SRS, los principios siguientes aplican:

a) deben declararse los requisitos específicos en la conformidad con todas las características descritas en 2.3.

b) los requisitos específicos deben tener referencias cruzadas a documentos más actuales que los relacionan.

c) Todos los requisitos deben ser singularmente identificables.

d) debe prestarse la atención debida a organizar los requisitos para aumentar al máximo la legibilidad.

Antes de examinar maneras específicas de organizar los requisitos es útil entender los varios puntos como que comprenden los requisitos descritos en 3.3.1 a través de 3.3.7.

3.3.1 Interfaces externas

Ésta debe ser una descripción detallada de todas las entradas y salidas del sistema del software. Debe complementar las descripciones de la interfaz en 3.2 y no debe repetirse la información allí.
Debe incluir ambas entradas/salidas y debe estructurarse como sigue:

a) el nombre de artículo;
b) la descripción de propósito;
c) la fuente de entrada o destino de salida;
d) el rango válido, exactitud, y/o tolerancia;
e) las unidades de medida;
f) tiempos;
g) las relaciones a otras entradas/salidas;
h) el formato de pantalla /organización;
i) el formato de ventanas/organización;
j) los formatos de los datos;
k) los formatos de los comandos;
l) fin de mensajes.

3.3.2 Funciones

Los requisitos funcionales deben definir las acciones fundamentales que deben tener lugar en el software, aceptando y procesando las entradas, procesando y generando las salidas. Éstos generalmente se listan como "debe" declaraciones que empiezan con "El sistema debe…."
Éstos incluyen:

a) verificar la validez sobre las entradas
b) la secuencia exacta de las operaciones
c) las contestaciones a las situaciones anormales, incluyendo
   1) overflow
   2) facilidades de comunicación
   3) manejo de errores y recuperación
d) el efecto de parámetros
e) la relación de salidas a las entradas, incluyendo
   1) las secuencias de entrada/salidas
   2) las fórmulas de entrada y su conversión a la salida

Puede ser apropiado dividir los requisitos funcionales en subfunciones o subprocesos. Esto no implica que el plan del software también se dividirá así.

3.3.3 Requisitos del desarrollo.

Esta subdivisión debe especificar los requerimientos estáticos y dinámicos que se pusieron en el software o en la interacción humana con el software en conjunto. Los requisitos estáticos pueden incluir a lo siguiente:

a) El número de terminales a ser apoyadas;
b) El número de usuarios simultáneos ser apoyados;
c) La cantidad y tipo de información que se manejara.

A veces se identifican los requisitos estáticos bajo una sección separada titulada la Capacidad. Por ejemplo, los requisitos dinámicos pueden incluir los números de transacciones, tareas y la cantidad de datos a ser procesado dentro de ciertos periodos de tiempo para las condiciones del trabajo normales y máximas.
Todos que estos requisitos deben declararse en las condiciones mensurables. Por ejemplo,

95% de las transacciones se procesarán en menos de 1 seg.

La NOTA - normalmente se especifican límites numéricos aplicados a una función específica como la parte de la descripción de subinciso de proceso de esa función.

5.3.4 Requisitos del banco de datos lógicos

Esto debe especificar los requisitos lógicos para cualquier información que será puesta en un banco de datos. Esto puede incluir a lo siguiente:

a) los tipos de información usadas por varias funciones;

b) la frecuencia de uso;
c) accediendo las capacidades;
d) las entidades de los datos y sus relaciones;
e) las restricciones de integridad;
f) requerimientos en la retención de datos.

3.3.5 Restricciones del diseño.

Esto debe especificar las restricciones del diseño que pueden imponerse por otros standares, las limitaciones del hardware, etc.,

3.3.5.1 Aceptación de las normas

Esta subdivisión debe especificar los requisitos derivados de standares existentes o regulaciones. Ellos pueden incluir a lo siguiente:

a) el formato del reporte;
b) los nombres de los datos;
c) los procedimientos de contabilidad;
d) los lineamientos de la Auditoría.

Por ejemplo, esto podría especificar los requisito para el software y rastrear la actividad del proceso. Se necesita rastrear algunas aplicaciones para encontrarse al menos las normas reguladoras o financieras. Por ejemplo, un requisito de rastro de auditoría puede declarar que deben grabarse todos los cambios a un banco de datos de la nómina en un archivo del rastro con los valores antes del proceso y después del proceso.

3.3.6 Atributos del software del sistema.

Hay varios atributos del software que puede servir como los requisitos. Es importante que los atributos se especifique para que su logro pueda verificarse objetivamente. Subclauses 3.3.6.1 a través de 3.3.6.5 proporcionan una lista parcial de ejemplos.

3.3.6.1 Fiabilidad

Esto debe especificar que los factores exigieron establecer la fiabilidad requerida del sistema del software al momento de la entrega.

3.3.6.2 Disponibilidad

Esto debe especificar que los factores exigieron garantizar un nivel de disponibilidad definido para el sistema como un punto de control, la recuperación y al iniciar.

3.3.6.3 Seguridad

Esto debe especificar los factores que protegen el software del acceso accidental o malévolo, uso, modificación, destrucción o descubrimiento. Los requisitos específicos en esta área podrían incluir la necesidad a:

a) Utilice ciertas técnicas de encriptamiento;
b) Tenga Log de entrada o históricos de datos;
c) Asigne ciertas funciones a módulos diferentes;
d) Restrinja las comunicaciones entre algunas áreas del programa;
e) La integridad de datos se verifique para variables críticas.

3.3.6.4 Mantenimiento

Esto debe especificar atributos de software que relaciona a la facilidad de mantenimiento del propio software. Puede haber algún requisito con toda seguridad de modularidad, interfaces, la complejidad, etc. no deben ponerse los requisitos aquí.

3.3.6.5 portabilidad

Esto debe especificar atributos de software que relaciona a la facilidad de poner el software a otro servidor y/o sistemas operativos. Esto puede incluir a lo siguiente:

a) el Porcentaje de componentes con código cliente-servidor;
b) el Porcentaje de código del cliente-servidor;
c) el Uso de un idioma portátil probado;
d) el Uso de un compilador particular o subconjunto de lenguajes;
e) el Uso de un sistema operativo particular.

3.3.7 Organizar los requisitos específicos.

Por algo los requisitos detallados de los sistemas triviales tienden a ser extenso. Por esta razón, se recomienda que sean cuidadosos de organizar éstos de una manera óptima para que sean entendibles.

3.3.7.1 Modo del sistema

Algunos sistemas se comportan diferentes dependiendo del modo de operación. Por ejemplo, un sistema de control puede tener juegos diferentes de funciones que dependen de su control: entrenando, normal o emergencia. Al organizar esta sección por el modo, el contorno en A.1 o A.2 debe usarse. La opción depende de las interfaces y del desarrollo que son dependientes del modo de acceso.

3.3.7.2 Clases de usuario

Algunos sistemas proporcionan juegos diferentes de funciones a las clases diferentes de usuarios. Por ejemplo, un sistema de mando de ascensor presenta las capacidades diferentes a los pasajeros, obreros de mantenimiento y bomberos. Al organizar esta sección por la clase del usuario, el contorno en A.3 debe usarse.

3.3.7.3 objetos

Los objetos son entidades del mundo real que tienen una contraparte dentro del sistema. Por ejemplo, en un sistema que supervisa pacientes, los objetos incluyen a los pacientes, los sensores, enfermeras, los cuartos, médicos, las medicinas, etc. Asociado con cada

objeto un juego de atributos a está (de ese objeto) y funciones (realizadas por ese objeto). Estas funciones también se llaman servicios, métodos o procesos. Al organizar esta sección por el objeto, el contorno en A.4 debe usarse. Nota que al poner los objetos puede compartir atributos y servicios. Éstos se agrupan como las clases.

3.3.7.4 Rasgo

Un rasgo es un servicio externamente deseado por el sistema que puede exigir a una secuencia de entradas efectuar el resultado deseado. Por ejemplo, en un sistema del teléfono, los rasgos incluyen la llamada local, llamada remitida y llamada en conferencia. Cada rasgo generalmente se describe en una secuencia de estímulo-contestación.

3.3.7.5 Estímulo

Algunos sistemas pueden organizarse mejor describiendo sus funciones por lo que se refiere a los estímulos. Por ejemplo, pueden organizarse las funciones de un avión automático que aterriza, el sistema en las secciones para la pérdida del control, esquivación del viento, el cambio súbito en el destino, la velocidad vertical excesiva, etc. Al organizar esta sección por el estímulo, el contorno en A.6 debe usarse.

3.3.7.6 Contestación

Algunos sistemas pueden organizarse mejor describiendo todas las funciones en el apoyo de la generación de una contestación. Por ejemplo, pueden organizarse las funciones de un sistema del personal en secciones que corresponden a todas las funciones asociadas con los sueldos generados, todas las funciones asociadas con generar una lista actual de empleados, etc. El contorno en A.6 (con todas las ocurrencias de estímulo reemplazadas con la contestación) debe usarse.

3.3.7.7 Jerarquía Funcional

Cuando ninguno de los esquemas orgánicos anteriores demuestra ser útil, la funcionalidad global puede organizarse en una jerarquía de funciones organizada por cualesquier entradas comunes, rendimientos comunes o el acceso de los datos interiores común. Los datos fluyen pueden usarse diagramas y diccionarios de datos para mostrar las relaciones entre las funciones y datos. Al organizar esta sección por la jerarquía funcional, el contorno en A.7 debe usarse.

3.3.8 Comentarios adicionales

Siempre que un nuevo SRS se contemple, más de una de las técnicas organizacionales dadas en 3.3.7.7 pueden ser apropiadas. En tal caso, organice los requisitos específicos para jerarquías múltiples detalladas a las necesidades específicas del sistema bajo la especificación.
Hay muchas anotaciones, métodos y herramientas de apoyo automatizadas disponibles para ayudar en la documentación de requisitos. La mayor parte, su utilidad es una función de organización. Por ejemplo, al organizar por el modo, máquinas de estado finitas o los mapas estatales pueden demostrar utilidad; al organizar por el objeto, el análisis objeto-orientado puede demostrar utilidad; al organizar por el rasgo, las

secuencias de estímulo-contestación pueden demostrar utilidad y al organizar por la jerarquía funcional, los datos fluyen según los diagramas y los diccionarios de datos pueden demostrar también utilidad.
En cualquiera de los contornos dados A.1 a través de A.8, esas secciones llamadas "Requisito Funcional" puede describirse en el idioma nativo (por ejemplo, inglés), en el pseudo código, en un idioma de definición de sistema, o en cuatro subdivisiones tituladas: La introducción, Entradas, Proceso, y Rendimientos.

3.4 Información de apoyo

La información de apoyo hace más fácil al SRS para usarse. Incluye a lo siguiente:

a) Tabla de contenidos;
b) Índice;
c) Apéndice.

3.4.1 Tabla de contenidos e índice

La tabla de contenidos e índice es bastante importante y debe seguir las prácticas de las composiciones generales.

3.4.2 Apéndices

Los apéndices no siempre son considerados parte del SRS real y no siempre son necesarios. Ellos pueden incluir:

a) Ejemplos de formatos de las entradas/salidas, las descripciones del análisis del costo que se estudiaron o resultados de estudios del usuario;

b) Apoyando o dando información a fondo que puede ayudar a los lectores del SRS;

c) Una descripción de los problemas a ser resuelto por el software;

d) las instrucciones del empaquetamiento especiales para el código y los medios de comunicación para reunir la seguridad, exportar la carga inicial u otros requisitos.
Cuando los apéndices son incluido, el SRS debe declarar explícitamente si o no los apéndices serán considerados parte de los requisitos.

Anexo A
(informativo)

Las plantillas de SRS

La Plantilla de A.1 de SRS Sección 3 organizada por el modo: Versión 1
3. Los requisitos específicos
   3.1 requisitos de las interfaces externas
      3.1.1 interfaz con el usuario
      3.1.2 interfaz con el hardware
      3.1.3 interfaz con el software

3.1.4 interfaces de comunicaciones
3.2 requisitos funcionales
3.2.1 modo 1
3.2.1.1 requisito 1.1 funcional
.
.
.
3.2.1.n requisito 1.n Funcional
3.2.2 modo 2
.
.
.
3.2.m Modo m
3.2.m.1 requisito Funcional m.1
.
.
.
3.2.m.n requisito Funcional m.n
3.3 Requisitos del desarrollo
3.4 Restricciones del diseño
3.5 Atributos de sistema de software
3.6 Otros requisitos

La Plantilla de A.2 de SRS Sección 3 organizada por el modo: Versión 2

3. Los requisitos específicos
3.1. Los requisitos funcionales
3.1.1 modo 1
3.1.1.1 interfaces externas
3.1.1.1.1 interfaz con el usuario
3.1.1.1.2 interfaz con el hardware
3.1.1.1.3 interfaz con el software
3.1.1.1.4 interfaces de comunicaciones
3.1.1.2 requisitos funcionales
3.1.1.2.1 requisito 1 funcional
.
.
3.1.1.2.n requisito Funcional n
3.1.1.3 Actuación
3.1.2 Modo 2
.
.
.
3.1.m Modo m
3.2 Restricciones del diseño
3.3 Atributos de sistema de software
3.4 Otros requisitos

La Plantilla de A.3 de SRS Sección 3 organizada por la clase del usuario

3. Los requisitos específicos
   3.1 Requisitos de la interface externa
      3.1.1 interfaz del usuario
      3.1.2 interfaz del hardware
      3.1.3 interfaz del software
      3.1.4 interfaces de comunicaciones
   3.2 Requisitos funcionales
      3.2.1 usuario clase 1
         3.2.1.1 requisito 1.1 funcional
.
.
.
         3.2.1.n requisito 1.n Funcional
      3.2.2 usuario clase 2
.
.
.
      3.2.m clase del Usuario m
      3.2.m.1 requisito Funcional m.1
.
.
.
      3.2.m.n requisito Funcional m.n
    3.3 requisitos de la actuación
    3.4 constreñimiento del plan
    3.5 atributos de sistema de software
    3.6 otros requisitos

La Plantilla de A.4 de SRS Sección 3 organizada por el objeto

3. Los requisitos específicos
   3.1 Requisitos de la interface externas
      3.1.1 interfaz con el usuario
      3.1.2 interfaz de hardware
      3.1.3 interfaz de software
      3.1.4 interfaces de comunicaciones
   3.2 Classes/Objects
      3.2.1 Class/Object 1
         3.2.1.1 atributos (directo o heredó)
            3.2.1.1.1 atributo 1
.
.
.
            3.2.1.1.n Atributo n
         3.2.1.2 funciones (los servicios, los métodos, directo o heredó)
            3.2.1.2.1 requisito 1.1 funcional
.
.
.

3.2.1.2.m requisito 1.m Funcional
3.2.1.3 Mensajes (las comunicaciones recibieron o enviaron)
3.2.2 Class/Object 2
.
.
.
3.2.p Class/Object p
3.3 Requisitos del desarrollo
3.4 Restricciones del diseño
3.5 Atributos de sistema de software
3.6 Otros requisitos

La Plantilla de A.5 de SRS Sección 3 organizada por el rasgo

3. Los requisitos específicos
3.1 Requisitos de la interface externas
3.1.1 Interfaz del usuario
3.1.2 Interfaz del hardware
3.1.3 Interfaz del software
3.1.4 interfaces de comunicaciones
3.2 Sistema ofrece
3.2.1 Sistema Rasgo 1
3.2.1.1 Introducción / Propósito de rasgo
3.2.1.2 Secuencia de estimulo / Respuesta
3.2.1.3 requisitos funcionales asociados
3.2.1.3.1 requisito 1 funcional
.
.
.
3.2.1.3.n requisito Funcional n
3.2.2 sistema rasgo 2
.
.
.
3.2.m rasgo del Sistema m
.
.
.
3.3 Requisitos de la actuación
3.4 Restricción del diseño
3.5 Atributos de sistema de software
3.6 Otros requisitos
.

La Plantilla de A.6 de SRS Sección 3 organizada por el estímulo

3. Los requisitos específicos
3.1 Requisitos de la interface externas
3.1.1 interfaz del usuario
3.1.2 interfaz del hardware
3.1.3 interfaz del software
3.1.4 interfaces de comunicaciones
3.2 Requisitos funcionales

3.2.1 Estímulo 1
3.2.1.1 Requisito 1.1 funcional
.
.
.
3.2.1.n Requisito 1.n Funcional
3.2.2 Estímulo 2
.
.
.
3.2.m Estímulo m
3.2.m.1 Requisito Funcional m.1
.
.
.
3.2.m.n Requisito Funcional m.n
3.3 Requisitos del desarrollo
3.4 Restricciones del diseño
3.5 Atributos del software del sistema
3.6 Otros requisitos

La Plantilla de A.7 de SRS Sección 3 organizada por la jerarquía funcional

3. Los requisitos específicos
3.1 Requisitos de la interface externos
3.1.1 Interfaz del usuario
3.1.2 Interfaz del hardware
3.1.3 Interfaz del software
3.1.4 Interfaces de comunicaciones
3.2 Requisitos funcionales
3.2.1 Fluido de la información
3.2.1.1 Fluijo de datos diagrama 1
3.2.1.1.1 Entidades de los datos
3.2.1.1.2 Procesos pertinentes
3.2.1.1.3 Topología
3.2.1.2 Flujo de datos diagrama 2
3.2.1.2.1 Entidades de los datos
3.2.1.2.2 Procesos pertinentes
3.2.1.2.3 Topología
.
.
.
3.2.1.n Flujo de datos diagrama n
3.2.1.n.1 Entidades de los Datos
3.2.1.n.2 Procesos Pertinentes
3.2.1.n.3 Topología
3.2.2 Descripciones del proceso
3.2.2.1 Proceso 1
3.2.2.1.1 Entidades de datos de entrada
3.2.2.1.2 Algoritmo o fórmula de proceso
3.2.2.1.3 Entidades de los datos afectado

3.2.2.2 Proceso 2
3.2.2.2.1 Entidades de datos de entrada
3.2.2.2.2 Algoritmo o fórmula de proceso
3.2.2.2.3 Entidades de los datos afectado
.
.
.
3.2.2.m Proceso m
3.2.2.m.1 Entidades de datos de Entrada
3.2.2.m.2 Algoritmo o fórmula de proceso
3.2.2.m.3 Entidades de los datos Afectado
3.2.3 Construcción de las especificaciones de los datos
3.2.3.1 Estructura 1
3.2.3.1.1 Tipo del registro
3.2.3.1.2 Elector presenta
3.2.3.2 Estructura 2
3.2.3.2.1 Tipo del registro
3.2.3.2.2 Elector presenta
.
.
.
3.2.3.p Estructura p
3.2.3.p.1 Tipo del Registro
3.2.3.p.2 Elector presenta
3.2.4 Diccionario de los datos
3.2.4.1 Datos elemento 1
3.2.4.1.1 Nombre
3.2.4.1.2 Representación
3.2.4.1.3 Units/Format
3.2.4.1.4 Precision/Accuracy
3.2.4.1.5 Rango
3.2.4.2 Datos elemento 2
3.2.4.2.1 Nombre
3.2.4.2.2 Representación
3.2.4.2.3 Units/Format
3.2.4.2.4 Precision/Accuracy
3.2.4.2.5 Rango
.
.
.
3.2.4.q Elemento de los Datos q
3.2.4.q.1 Nombre
3.2.4.q.2 Representación
3.2.4.q.3 Units/Format
3.2.4.q.4 Precision/Accuracy
3.2.4.q.5 Rango
.
3.3 Requisitos del desarrollo
3.4 Restricciones del diseño
3.5 Atributos del software del sistema

3.6 Otros requisitos

La Plantilla de A.8 de Sección de SRS 3 exhibición las organizaciones múltiples

3. Los requisitos específicos
   3.1 Requisitos de la interface externa
      3.1.1 Interfaz con el usuario
      3.1.2 Interfaz con el hardware
      3.1.3 Interfaz con el software
      3.1.4 Interfaces de comunicaciones
   3.2 Requisitos funcionales
      3.2.1 Usuario clase 1
         3.2.1.1 Rasgo 1.1
            3.2.1.1.1 Introducción / Propósito de rasgo
            3.2.1.1.2 secuencia de estímulos /Respuestas
            3.2.1.1.3 requisitos funcionales asociados
         3.2.1.2 Rasgo 1.2
            3.2.1.2.1 Introducción / Propósito de rasgo
            3.2.1.2.2 Secuencia de estímulos/ Respuestas
            3.2.1.2.3 Requisitos funcionales asociados
.
.
.
         3.2.1.m Rasgo 1.m
            3.2.1.m.1 Introducción /Propósito de rasgo
            3.2.1.m.2 Secuencia de estímulos /Respuestas
            3.2.1.m.3 Requisitos funcionales Asociados
      3.2.2 usuario clase 2
.
.
.
      3.2.n clase del Usuario n
.
.
.
3.3 Requisitos del desarrollo
3.4 Restricciones de diseño
3.5 Atributos del software del sistema
3.6 Otros requisitos