IEEE Std 830-1998
(Revisión de
IEEE Std 830-1993)

IEEE Prácticas Recomendadas para Especificaciones de Requisitos de Software

Sponsor
Comité de Estándares de Ingeniría de Software de la Sociedad de Computación IEEE.

Aprobado el 25 de Junio de 1998
Consejo de Normas de IEEE-SA

Resumen: Se describen el contenido y las cualidades de una buena especificación de requisitos de software (SRS, por sus siglas en inglés), y se presentan varios esquemas de SRS de muestra. Esta práctica recomendada tiene como objetivo especificar los requisitos del software a desarrollar, pero también puede aplicarse para ayudar en la selección de productos de software internos y comerciales. También se proporcionan pautas para cumplir con IEEE/EIA 12207.1-1997.
Palabras clave: contrato, cliente, prototipado, especificación de requisitos de software, proveedor, especificaciones de requisitos del sistema.

The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street, New York, NY 10017-2394, USA

ISBN 0-7381-0332-2

Los documentos de normas de IEEE se desarrollan dentro de las Sociedades de IEEE y los Comités Coordinadores de Normas de la Junta de Normas de la Asociación de Normas de IEEE (IEEE-SA). Los miembros de los comités sirven de manera voluntaria y sin compensación. No necesariamente son miembros del Instituto. Las normas desarrolladas dentro de IEEE representan un consenso de la amplia experiencia en el tema dentro del Instituto, así como de aquellas actividades fuera de IEEE que han expresado interés en participar en el desarrollo de la norma.

El uso de una Norma IEEE es completamente voluntario. La existencia de una Norma IEEE no implica que no haya otras formas de producir, probar, medir, comprar, comercializar o proporcionar otros bienes y servicios relacionados con el alcance de la Norma IEEE. Además, el punto de vista expresado en el momento en que se aprueba y emite una norma está sujeto a cambios provocados por avances en el estado del arte y comentarios recibidos de usuarios de la norma. Cada Norma IEEE está sujeta a revisión al menos cada cinco años para revisión o reafirmación. Cuando un documento tiene más de cinco años y no ha sido reafirmado, es razonable concluir que su contenido, aunque aún tenga algún valor, no refleja completamente el estado actual del arte. Se recomienda a los usuarios verificar que tienen la última edición de cualquier Norma IEEE.

Se agradecen los comentarios para la revisión de Normas IEEE de cualquier parte interesada, independientemente de la afiliación de membresía con IEEE. Las sugerencias para cambios en los documentos deben presentarse en forma de un cambio propuesto de texto, junto con comentarios de respaldo apropiados.

Interpretaciones: Ocasionalmente pueden surgir preguntas sobre el significado de partes de las normas en relación con aplicaciones específicas. Cuando la necesidad de interpretaciones se pone de manifiesto ante IEEE, el Instituto tomará medidas para preparar respuestas apropiadas. Dado que las Normas IEEE representan un consenso de todos los intereses involucrados, es importante asegurar que cualquier interpretación también haya recibido el acuerdo de un equilibrio de intereses. Por esta razón, IEEE y los miembros de sus sociedades y comités coordinadores de normas no pueden proporcionar una respuesta instantánea a solicitudes de interpretación, excepto en aquellos casos donde el asunto haya recibido consideración formal previa.

Comentarios sobre normas y solicitudes de interpretaciones deben dirigirse a:

Secre1ary, IEEE-SA S1andards Board
445 Hoes Lane
P.O. Box 1331
Pisca1away, NJ 08855-1331
USA

Nota: Se llama la atención sobre la posibilidad de que la implementación de esta norma pueda requerir el uso de materia cubierta por derechos de patente. Con la publicación de esta norma, no se toma posición con respecto a la existencia o validez de derechos de patente en relación con la misma. IEEE no será responsable de identificar patentes para las cuales pueda ser necesaria una licencia según una norma IEEE ni de realizar investigaciones sobre la validez legal o alcance de esas patentes que se le notifiquen.

Introducción

(Esta introducción no forma parte de IEEE Std 830-1998, Práctica Recomendada de IEEE para Especificaciones de Requisitos de Software).

Esta práctica recomendada describe enfoques recomendados para la especificación de requisitos de software. Se basa en un modelo en el que el resultado del proceso de especificación de requisitos de software es un documento de especificación claro y

a) A los clientes de software a describir con precisión lo que desean obtener;
b) A los proveedores de software a comprender exactamente lo que el cliente desea;
c) A los individuos a lograr los siguientes objetivos:
    1) Desarrollar un esquema estándar de especificación de requisitos de software (SRS) para sus propias organizaciones;
    2) Definir el formato y contenido de sus especificaciones de requisitos de software específicas;
    3) Desarrollar elementos de apoyo adicionales a nivel local, como una lista de verificación de calidad de SRS o un manual del escritor de SRS.

Para los clientes, proveedores y otras personas, una buena SRS debería proporcionar varios beneficios específicos, como los siguientes:

— *Establecer la base para el acuerdo entre los clientes y los proveedores sobre lo que el* producto de software debe hacer. La descripción completa de las funciones que debe realizar el software especificado en el SRS ayudará a los usuarios potenciales a determinar si el software especificado satisface sus necesidades o cómo se debe modificar el software para cumplir con sus necesidades.

— Reducir el esfuerzo de desarrollo. La preparación del SRS obliga a los diversos grupos interesados en la organización del cliente a considerar rigurosamente todos los requisitos antes de que comience el diseño, lo que disminuye la necesidad de posteriores rediseños, recodificaciones y reevaluaciones. Una revisión cuidadosa de los requisitos en el SRS puede revelar omisiones, malentendidos e inconsistencias en las primeras etapas del ciclo de desarrollo, cuando estos problemas son más fáciles de corregir.

— Proporcionar una base para estimar costos y plazos. La descripción del producto a desarrollar según se presenta en el SRS es una base realista para estimar los costos del proyecto y puede utilizarse para obtener aprobación de ofertas o estimaciones de precios.

— Proporcionar una base para la validación y verificación. Las organizaciones pueden desarrollar sus planes de validación y verificación de manera mucho más productiva a partir de un buen SRS. Como parte del contrato de desarrollo, el SRS proporciona una línea base contra la cual se puede medir el cumplimiento.

— Facilitar la transferencia. El SRS facilita la transferencia del producto de software a nuevos usuarios o nuevas máquinas. Los clientes encuentran más fácil transferir el software a otras partes de su organización, y los proveedores lo encuentran más sencillo para transferirlo a nuevos clientes.

— Servir como base para mejoras. Debido a que el SRS aborda el producto pero no el proyecto que lo desarrolló, el SRS sirve como base para la mejora posterior del producto terminado. Es posible que el SRS deba modificarse, pero proporciona una base para una evaluación continua de la producción.

Los lectores de este documento son remitidos al Anexo B para obtener pautas sobre cómo utilizar esta práctica recomendada para cumplir con los requisitos de IEEE/EIA 12207.1-1997, IEEE/EIA Guide - Industry Implementation of ISO/IEC 12207: 1995, Standard for Information Technology - Software life cycle processes - Life cycle data.

Participantes

Esta práctica recomendada fue preparada por el Grupo de Trabajo de Armonización de Datos del Ciclo de Vida del Comité de Normas de Ingeniería de Software de la IEEE Computer Society. En el momento en que esta práctica recomendada fue aprobada, el grupo de trabajo estaba compuesto por los siguientes miembros:

Leonard L. Tripp, *Chair*

Edward Byrne
Paul R. Croll
Perry DeWeese
Robin Fralick
Marilyn Ginsberg-Finner
John Harauz
Mark Henley
Dennis Lawrence
David Maibor
Ray Milovanovic
James Moore
Timo1hy Niesen
Dennis Rilling
Terry Rou1
Richard Schmid1
Norman F. Schneidewind
David Schul1z
Basil Sherlund
Pe1er Voldner
Ronald Wade

Las siguientes personas formaron parte del comité de votación:

Syed Ali
Theodore K. A1chinson
Mikhail Augus1on
Rober1 E. Barry
Leo Bel1racchi
H. Ronald Berlack
Richard E. Biehl
Michael A. Blackledge
Sandro Bologna
Juris Borzovs
Ka1hleen L. Briggs
M. Sco11 Buck
Michael Caldwell
James E. Cardow
Enrico A. Carrara
Lawrence Ca1chpole
Kei1h Chan
An1onio M. Cicu
Theo Clarke
Sylvain Clermon1
Rosemary Coleman
Virgil Lee Cooper
W. W. Geoff Cozens
Paul R. Croll
Gregory T. Daich
Geoffrey Darn1on
Taz Daugh1rey
Bos1jan K. Derganc
Perry R. DeWeese
James Do
Evelyn S. Dow
Carl Einar Drags1ed1
Sherman Eagles
Chris1of Eber1
Leo Egan
Richard E. Fairley
John W. Fendrich
Jay Fors1er
Kirby For1enberry
Eva Freund
Richard C. Fries
Roger U. Fujii
Adel N. Ghannam
Marilyn Ginsberg-Finner
John Gar1h Glynn
Julio Gonzalez-Sanz
L. M. Gun1her
David A. Gus1afson
Jon D. Hagar
John Harauz
Rober1 T. Harley
Herber1 Hech1
William Hefley
Manfred Hein
Mark Heinrich
Mark Henley
Debra Herrmann
John W. Horch
Jerry Huller
Pe1er L. Hung
George Jackelen
Frank V. Jorgensen
William S. Junk
George X. Kambic
Richard Karcich
Ron S. Kene11
Judi1h S. Kerner
Rober1 J. Kierzyk
Dwayne L. Knirk
Shaye Koenig
Thomas M. Kurihara
John B. Lane
J. Dennis Lawrence
Fang Ching Lim
William M. Lively
James J. Longbucco
Die1er Look
John Lord
S1an Magee
David Maibor
Harold Mains
Rober1 A. Mar1in
Tomoo Ma1subara
Mike McAndrew
Pa1rick D. McCray
Chris1opher McMacken
Jerome W. Mersky
Bre1 Michael
Alan Miller
Celia H. Modell
James W. Moore
Pavol Navra1
Myrna L. Olson
Indradeb P. Pal
Alex Polack
Pe1er T. Poon
Lawrence S. Przybylski
Kenne1h R. P1ack
Anne11e D. Reilly
Dennis Rilling
Andrew P. Sage
Helmu1 Sandmayr
S1ephen R. Schach
Hans Schaefer
Norman Schneidewind
David J. Schul1z
Lisa A. Selmon
Rober1 W. Shilla1o
David M. Siefer1
Carl A. Singer
James M. Sivak
Richard S. Sky
Nancy M. Smi1h
Melford E. Smyre
Harry M. Sneed
Alfred R. Sorkowi1z
Donald W. Sova
Luca Spo1orno
Julia S1esney
Fred J. S1rauss
Chris1ine Brown S1rysik
Toru Takeshi1a
Richard H. Thayer
Booker Thomas
Pa1ricia Trellue
Theodore J. Urbanowicz
Glenn D. Venables
Udo Voges
David D. Walden
Dolores Wallace
William M. Walsh
John W. Walz
Camille SWhi1e-Par1ain
Sco11 A. Whi1mire
P. A. Wolfgang
Paul R. Work
Na1alie C. Yopconka
Janusz Zalewski
Geraldine Zimmerman
Pe1er F. Zoll

Cuando la Junta de Normas de IEEE-SA aprobó esta práctica recomendada el 25 de junio de 1998, contaba con la siguiente membresía:

Richard J. Holleman, *Chair* **Donald N. Heirman,** *Vice Chair*

Judith Gorman, *Secretary*

Sa1ish K. Aggarwal
Clyde R. Camp
James T. Carlo
Gary R. Engmann
Harold E. Eps1ein
Jay Fors1er*
Thomas F. Garri1y
Ruben D. Garzon
James H. Gurney
Jim D. Isaak
Lowell G. Johnson
Rober1 Kennelly
E. G. "Al" Kiener
Joseph L. Koepfinger*
S1ephen R. Lamber1
Jim Logo1he1is
Donald C. Loughry
L. Bruce McClung
Louis-François Pau
Ronald C. Pe1ersen
Gerald H. Pe1erson
John B. Posey
Gary S. Robinson
Hans E. Weinrich
Donald W. Zipse

*Miembro Emérito

Valerie E. Zelen1y
Editor de Proyecto de Normas IEEE

Contenido

1. Panorama general 1

1.1 Alcance 1

2. Referencias. 2

3. Definiciones 2

4. Consideraciones para producir una buena SRS 3

4.1 Naturaleza de la SRS 3
4.2 Entorno de la SRS 3
4.3 Características de una buena SRS 4
4.4 Preparación conjunta de la SRS 8
4.5 Evolución de la SRS 8
4.6 Prototipado. 9
4.7 Incorporación del diseño en la SRS 9
4.8 Incorporación de los requisitos del proyecto en la SRS 10

5. Las partes de una SRS 10

5.1 Introducción (Sección 1 de la SRS). 11
5.2 Descripción General (Sección 2 de la SRS) 12
5.3 Requisitos Específicos (Sección 3 de la SRS) 15
5.4 Información de Apoyo 19

Anexo A (Informativo) Plantillas de SRS 21

Anexo B (Informativo) Pautas para el cumplimiento con IEEE/EIA 12207.1-1997 27

IEEE Prácticas Recomendadas para Especificaciones de Requisitos de Software

1. Visión general

Esta práctica recomendada describe enfoques recomendados para la especificación de requisitos de software. Se divide en cinco cláusulas. La Cláusula 1 explica el alcance de esta práctica recomendada. La Cláusula 2 enumera las referencias a otros estándares. La Cláusula 3 proporciona definiciones de términos específicos utilizados. La Cláusula 4 brinda información de fondo para redactar una buena SRS. La Cláusula 5 discute cada una de las partes esenciales de una SRS. Esta práctica recomendada también incluye dos anexos, uno que proporciona plantillas de formato alternativo y otro que ofrece pautas para cumplir con IEEE/EIA 12207.1-1997.

1.1 Alcance

Esta es una práctica recomendada para redactar especificaciones de requisitos de software. Describe el contenido y las cualidades de una buena especificación de requisitos de software (SRS) y presenta varios esquemas de SRS de muestra. Esta práctica recomendada tiene como objetivo especificar los requisitos del software a desarrollar, pero también puede aplicarse para ayudar en la selección de productos de software internos y comerciales. Sin embargo, su aplicación a software ya desarrollado podría ser contraproducente.

Cuando el software está integrado en algún sistema más grande, como equipo médico, entonces pueden surgir problemas más allá de los identificados en esta práctica recomendada y deberán abordarse.

Esta práctica recomendada describe el proceso de creación de un producto y el contenido del producto. El producto es una SRS. Esta práctica recomendada se puede utilizar para crear directamente una SRS o como modelo para un estándar más específico.

Esta práctica recomendada no identifica ningún método, nomenclatura o herramienta específica para preparar una SRS.

2. Referencias

Esta práctica recomendada deberá utilizarse en conjunto con las siguientes publicaciones.

ASTM E1340-96, S1andard Guide for Rapid Pro1o1yping of Compu1erized Sys1ems.[1]

IEEE S1d 610.12-1990, IEEE S1andard Glossary of Sof1ware Engineering Terminology.[2]

IEEE S1d 730-1998, IEEE S1andard for Sof1ware Quali1y Assurance Plans.

IEEE S1d 730.1-1995, IEEE Guide for Sof1ware Quali1y Assurance Planning.

IEEE S1d 828-1998, IEEE S1andard for Sof1ware Configura1ion Managemen1 Plans.[3]

IEEE S1d 982.1-1988, IEEE S1andard Dic1ionary of Measures 1o Produce Reliable Sof1ware.

IEEE S1d 982.2-1988, IEEE Guide for 1he Use of IEEE S1andard Dic1ionary of Measures 1o Produce Reliable Sof1ware.

IEEE S1d 1002-1987 (Reaff 1992), IEEE S1andard Taxonomy for Sof1ware Engineering S1andards.

IEEE S1d 1012-1998, IEEE S1andard for Sof1ware Verifica1ion and Valida1ion.

IEEE S1d 1012a-1998, IEEE S1andard for Sof1ware Verifica1ion and Valida1ion: Con1en1 Map 1o IEEE/EIA 12207.1-1997.[4]

IEEE S1d 1016-1998, IEEE Recommended Prac1ice for Sof1ware Design Descrip1ions.[5]

IEEE S1d 1028-1997, IEEE S1andard for Sof1ware Reviews.

IEEE S1d 1042-1987 (Reaff 1993), IEEE Guide 1o Sof1ware Configura1ion Managemen1.

IEEE P1058/D2.1, Draf1 S1andard for Sof1ware Projec1 Managemen1 Plans, da1ed 5 Augus1 1998.[6]

IEEE S1d 1058a-1998, IEEE S1andard for Sof1ware Projec1 Managemen1 Plans: Con1en1 Map 1o IEEE/EIA 12207.1-1997.[7]

IEEE S1d 1074-1997, IEEE S1andard for Developing Sof1ware Life Cycle Processes.

IEEE S1d 1233, 1998 Edi1ion, IEEE Guide for Developing Sys1em Requiremen1s Specifica1ions.[8]

---

[1]ASTM publica1ions are available from 1he American Socie1y for Tes1ing and Ma1erials, 100 Barr Harbor Drive, Wes1 Conshohocken, PA 19428-2959, USA.

[2]IEEE publica1ions are available from 1he Ins1i1u1e of Elec1rical and Elec1ronics Engineers, 445 Hoes Lane, P.O. Box 1331, Pisca1away, NJ 08855-1331, USA.

[3]As 1his s1andard goes 1o press, IEEE S1d 828-1998; IEEE S1d 1012a-1998; IEEE S1d 1016-1998; and IEEE S1d 1233, 1998 Edi1ion are approved bu1 no1 ye1 published. The draf1 s1andards are, however, available from 1he IEEE. An1icipa1ed publica1ion da1e is Fall 1998. Con1ac1 1he IEEE S1andards Depar1men1 a1 1 (732) 562-3800 for s1a1us informa1ion.

[4]See Foo1no1e 3.

[5]See Foo1no1e 3.

[6]Upon approval of IEEE P1058 by 1he IEEE-SA S1andards Board, 1his s1andard will be in1egra1ed wi1h IEEE S1d 1058a-1998 and published as IEEE S1d 1058, 1998 Edi1ion. Approval is expec1ed 8 December 1998.

[7]As 1his s1andard goes 1o press, IEEE S1d 1058a-1998 is approved bu1 no1 ye1 published. The draf1 s1andard is, however, available from 1he IEEE. An1icipa1ed publica1ion da1e is December 1998. Con1ac1 1he IEEE S1andards Depar1men1 a1 1 (732) 562-3800 for s1a1us informa1ion. See Foo1no1e 6.

[8]See Foo1no1e 3.

3. Definiciones

En general, las definiciones de los términos utilizados en esta práctica recomendada se ajustan a las definiciones proporcionadas en IEEE Std 610.12-1990. Las definiciones a continuación son términos clave tal como se utilizan en esta práctica recomendada.

3.1 contrato: Un documento legalmente vinculante acordado por el cliente y el proveedor. Esto incluye los requisitos técnicos y organizativos, el costo y el cronograma para un producto. Un contrato también puede contener información informal pero útil, como los compromisos o expectativas de las partes involucradas.

3.2 cliente: La persona o personas que pagan por el producto y generalmente (pero no necesariamente) deciden los requisitos. En el contexto de esta práctica recomendada, el cliente y el proveedor pueden ser miembros de la misma organización.

3.3 proveedor: La persona o personas que producen un producto para un cliente. En el contexto de esta práctica recomendada, el cliente y el proveedor pueden ser miembros de la misma organización.

3.4 usuario: La persona o personas que operan o interactúan directamente con el producto. El usuario(es) y el cliente(s) a menudo no son la misma(s) persona(s).

4. Consideraciones para producir una buena SRS

Esta cláusula proporciona información de fondo que debe tenerse en cuenta al redactar una SRS. Esto incluye lo siguiente:

a) Naturaleza de la SRS;
b) Entorno de la SRS;
c) Características de una buena SRS;
d) Preparación conjunta de la SRS;
e) Evolución de la SRS;
f) Prototipado;
g) Incorporación del diseño en la SRS;
h) Incorporación de los requisitos del proyecto en la SRS.

4.1 Naturaleza de la SRS

La SRS es una especificación para un producto de software específico, programa o conjunto de programas que realiza funciones particulares en un entorno específico. La SRS puede ser redactada por uno o varios representantes del proveedor, uno o varios representantes del cliente o por ambos. La Subcláusula 4.4 recomienda ambos.

Los problemas básicos que el(los) redactor(es) de la SRS deben abordar son los siguientes:

a) Funcionalidad. ¿Qué se supone que debe hacer el software?
b) Interfaces externas. ¿Cómo interactúa el software con las personas, el hardware del sistema, otros hardwares y otros softwares?
c) Rendimiento. ¿Cuál es la velocidad, disponibilidad, tiempo de respuesta, tiempo de recuperación de varias funciones del software, etc.?
d) Atributos. ¿Cuáles son las consideraciones de portabilidad, corrección, mantenibilidad, seguridad, etc.?
e) Restricciones de diseño impuestas en una implementación. ¿Existen estándares requeridos en vigencia, lenguaje de implementación, políticas para la integridad de la base de datos, límites de recursos, entorno(s) operativo(s), etc.?

El(los) redactor(es) de la SRS deben evitar incluir tanto requisitos de diseño como

requisitos de proyecto en la SRS. Para el contenido recomendado de una SRS, consulta

la Cláusula 5.

4.2 Entorno de la SRS

Es importante considerar el papel que juega la SRS (Especificación de Requisitos de Software) en el plan de proyecto total, que está definido en IEEE Std 610.12-1990. El software puede contener esencialmente toda la funcionalidad del proyecto o puede ser parte de un sistema más grande. En este último caso, típicamente habrá una SRS que establecerá las interfaces entre el sistema y su parte de software, y colocará requisitos de rendimiento y funcionalidad externos sobre la parte de software. Por supuesto, la SRS debe estar de acuerdo con y ampliar estos requisitos del sistema.

IEEE Std 1074-1997 describe los pasos en el ciclo de vida del software y las entradas aplicables para cada paso. Otros estándares, como los enumerados en la Cláusula 2, se relacionan con otras partes del ciclo de vida del software y, por lo tanto, pueden complementar los requisitos de software.

Dado que la SRS tiene un papel específico que desempeñar en el proceso de desarrollo de software, el(los) redactor(es) de la SRS deben tener cuidado de no salirse de los límites de ese papel. Esto significa que la SRS

a) Debería definir correctamente todos los requisitos de software. Un requisito de software puede existir debido a la naturaleza de la tarea a resolver o debido a una característica especial del proyecto.
b) No debería describir detalles de diseño o implementación. Estos deberían describirse en la etapa de diseño del proyecto.
c) No debería imponer restricciones adicionales al software. Estas se especifican adecuadamente en otros documentos, como un plan de aseguramiento de calidad del software.

Por lo tanto, una SRS correctamente redactada limita el rango de diseños válidos, pero no especifica ningún diseño particular.

4.3 Características de una buena SRS

Una SRS debe ser:

a) Correcta;
b) No Ambigua;
c) Completa;
d) Coherente;
e) Clasificado por importancia y/o estabilidad;
f) Verificable;
g) Modificable;
h) Rastreable.

4.3.1 Correcta

Una SRS es correcta si, y solo si, cada requisito declarado en ella es uno que el software debe cumplir. No hay herramienta o procedimiento que garantice la corrección. La SRS debe compararse con cualquier especificación superior aplicable, como una especificación de requisitos del sistema, con otra documentación del proyecto y con otras normas aplicables, para asegurarse de que esté de acuerdo. Alternativamente, el cliente o usuario puede determinar si la SRS refleja correctamente las necesidades reales. La trazabilidad facilita este procedimiento y lo hace menos propenso a errores (ver 4.3.8).

4.3.2 No Ambiguo

Una SRS es no ambigua si, y solo si, cada requisito declarado en ella tiene solo una interpretación. Como mínimo, esto requiere que cada característica del producto final se describa utilizando un único término único.

En casos en los que un término utilizado en un contexto particular podría tener múltiples significados, el término debe incluirse en un glosario donde su significado se haga más específico.

Una SRS es una parte importante del proceso de requisitos del ciclo de vida del software y se utiliza en el diseño, implementación, monitoreo del proyecto, verificación y validación, y en la capacitación, como se describe en IEEE Std 1074-1997. La SRS debe ser no ambigua tanto para quienes la crean como para quienes la utilizan. Sin embargo, estos grupos a menudo no tienen la misma formación y, por lo tanto, tienden a describir los requisitos de software de manera diferente. Representaciones que mejoren la especificación de requisitos para el desarrollador pueden ser contraproducentes, ya que disminuyen la comprensión para el usuario y viceversa.

Las subcláusulas 4.3.2.1 a 4.3.2.3 recomiendan cómo evitar la ambigüedad.

4.3.2.1 Problemas del lenguaje natural

Los requisitos suelen escribirse en lenguaje natural (por ejemplo, inglés). El lenguaje natural es inherentemente ambiguo. Un SRS en lenguaje natural debe ser revisado por una parte independiente para identificar el uso ambiguo del lenguaje para corregirlo.

4.3.2.2 Lenguajes de especificación de requisitos

Una forma de evitar la ambigüedad inherente al lenguaje natural es escribir la SRS en un lenguaje específico de especificación de requisitos. Sus procesadores de lenguaje detectan automáticamente muchos errores léxicos, sintácticos y semánticos.

Una desventaja en el uso de tales lenguajes es el tiempo requerido para aprenderlos. Además, muchos usuarios no técnicos los encuentran incomprensibles. Además, estos lenguajes tienden a ser mejores para expresar ciertos tipos de requisitos y abordar ciertos tipos de sistemas. Por lo tanto, pueden influir en los requisitos de maneras sutiles.

4.3.2.3 Herramientas de representación

En general, los métodos y lenguajes de requisitos y las herramientas que los respaldan se pueden clasificar en tres categorías generales: objetos, procesos 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 en jerarquías de funciones que se comunican mediante flujos de datos. Los enfoques de comportamiento describen el comportamiento externo del sistema en términos de alguna noción abstracta (como el cálculo de predicados), funciones matemáticas o máquinas de estados.

El grado en que estas herramientas y métodos pueden ser útiles en la preparación de una SRS depende del tamaño y complejidad del programa. No se hace ningún intento aquí de describir o respaldar alguna herramienta en particular.

Al usar cualquiera de estos enfoques, es mejor retener las descripciones en lenguaje natural. De esa manera, los clientes que no estén familiarizados con las notaciones aún pueden entender la SRS.

4.3.3 Completo

Una SRS es completa si, y solo si, incluye los siguientes elementos:

a) Todos los requisitos significativos, ya sea relacionados con la funcionalidad, el rendimiento, las restricciones de diseño, atributos o interfaces externas. En particular, se deben reconocer y tratar cualquier requisito externo impuesto por una especificación del sistema.

b) Definición de las respuestas del software a todas las clases realizables de datos de entrada en todas las clases realizables de situaciones. Es importante especificar las respuestas tanto para valores de entrada válidos como para inválidos.
c) Etiquetas completas y referencias a todas las figuras, tablas y diagramas en la SRS y definición de todos los términos y unidades de medida.

4.3.3.1 Uso de TBDs

Cualquier SRS que utilice la frase "por determinar" (TBD) no es una SRS completa. Sin embargo, en ocasiones es necesario el uso de TBD y debe estar acompañado por

a) Una descripción de las condiciones que causan el TBD (por ejemplo, por qué no se conoce una respuesta) para que la situación pueda resolverse.
b) Una descripción de lo que debe hacerse para eliminar el TBD, quién es responsable de su eliminación y para cuándo debe eliminarse.

4.3.4 Consistente

La consistencia se refiere a la consistencia interna. Si una SRS no está de acuerdo con algún documento de nivel superior, como una especificación de requisitos del sistema, entonces no es correcta (ver 4.3.1).

4.3.4.1 Consistencia interna

Una SRS es internamente consistente si, y solo si, ningún subconjunto de requisitos individuales descritos en ella entra en conflicto. Los tres tipos de conflictos probables en una SRS son los siguientes:

a) Las características especificadas de objetos del mundo real pueden entrar en conflicto. Por ejemplo,
   1) El formato de un informe de salida puede describirse en un requisito como tabular, pero en otro como textual.
   2) Un requisito puede establecer que todas las luces deben ser verdes, mientras que otro puede establecer que todas las luces deben ser 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 las multiplicará.
   2) Un requisito puede establecer que "A" siempre debe seguir a "B", mientras que otro puede requerir que "A y B" ocurran simultáneamente.
c) Dos o más requisitos pueden describir el mismo objeto del mundo real pero usar términos diferentes para ese objeto. Por ejemplo, la solicitud de entrada de un programa para un usuario puede llamarse "prompt" en un requisito y "cue" en otro. El uso de terminología y definiciones estándar promueve la consistencia.

4.3.5 Clasificado por importancia y/o estabilidad

Una SRS está clasificada por importancia y/o estabilidad si cada requisito en ella tiene un identificador que indica la importancia o estabilidad de ese requisito en particular.

Normalmente, no todos los requisitos que se relacionan con un producto de software son igualmente importantes. Algunos requisitos pueden ser esenciales, especialmente para aplicaciones críticas para la vida, mientras que otros pueden ser deseables.

Cada requisito en la SRS debe ser identificado para hacer estas diferencias claras y explícitas. Identificar los requisitos de la siguiente manera ayuda:

a) Hacer que los clientes consideren más cuidadosamente cada requisito, lo que a menudo aclara cualquier suposición oculta que puedan tener.
b) Hacer que los desarrolladores tomen decisiones de diseño correctas y dediquen niveles apropiados de esfuerzo a las diferentes partes del producto de software.

4.3.5.1 Grado de estabilidad

Un método para identificar requisitos utiliza la dimensión de la estabilidad. La estabilidad puede expresarse en términos del número de cambios esperados en cualquier requisito basado en la experiencia o el conocimiento de eventos futuros que afectan a la organización, funciones y personas respaldadas por el sistema de software.

4.3.5.2 Grado de necesidad

Otra forma de clasificar los requisitos es distinguir clases de requisitos como esenciales, condicionales y opcionales.

a) Esencial. Implica que el software no será aceptable a menos que estos requisitos se proporcionen de manera acordada.
b) Condicional. Implica que estos son requisitos que mejorarían el producto de software, pero que no lo harían inaceptable si están ausentes.
c) Opcional. Implica una clase de funciones que pueden o no ser valiosas. Esto le da al proveedor la oportunidad de proponer algo que exceda la SRS.

4.3.6 Verificable

Una SRS es verificable si, y solo si, cada requisito declarado en ella es verificable. Un requisito es verificable si, y solo si, existe algún proceso finito y rentable con el cual una persona o máquina puede verificar que el producto de software cumple con el requisito. En general, cualquier requisito ambiguo no es verificable.

Los requisitos no verificables incluyen afirmaciones como "funciona bien", "buena interfaz humana" y "debería suceder generalmente". Estos requisitos no se pueden verificar porque es imposible definir los términos "bien", "bueno" o "generalmente". La afirmación de que "el programa nunca entrará en un bucle infinito" no es verificable porque la prueba de esta calidad es teóricamente imposible.
Un ejemplo de una declaración verificable es

> *La salida del programa deberá generarse en un plazo de 20 segundos después del evento el 60% del tiempo; y deberá generarse en un plazo de 30 segundos después del evento el 100% del tiempo.*

Esta afirmación puede verificarse porque utiliza términos concretos y cantidades medibles.

Si no se puede idear un método para determinar si el software cumple con un requisito en particular, entonces ese requisito debe ser eliminado o revisado.

4.3.7 Modificable

Una SRS es modificable si, y solo si, su estructura y estilo son tales que cualquier cambio en los requisitos se pueda realizar fácil, completa y consistentemente, manteniendo la estructura y estilo. La modificabilidad generalmente requiere que una SRS

a) Tener una organización coherente y fácil de usar con una tabla de contenidos, un índice y referencias cruzadas explícitas;
b) No ser redundante (es decir, el mismo requisito no debería aparecer en más de un lugar en la SRS);
c) Expresar cada requisito por separado, en lugar de estar intercalado con otros requisitos.

La redundancia en sí misma no es un error, pero puede llevar fácilmente a errores. La redundancia ocasionalmente puede ayudar a que una SRS sea más legible, pero puede surgir un problema cuando el documento redundante se actualiza. Por ejemplo, un requisito puede modificarse en solo uno de los lugares donde aparece. La SRS se vuelve inconsistente. Siempre que la redundancia sea necesaria, la SRS debería incluir referencias cruzadas explícitas para facilitar la modificabilidad.

4.3.8 Trazable

Una SRS es trazable si el origen de cada uno de sus requisitos es claro y si facilita la referencia de cada requisito en la documentación futura de desarrollo o mejora. Se recomiendan los siguientes dos tipos de trazabilidad:

a) Trazabilidad hacia atrás (es decir, hacia las etapas anteriores del desarrollo). Esto depende de que cada requisito haga referencia explícita a su origen en documentos anteriores.
b) Trazabilidad hacia adelante (es decir, hacia todos los documentos generados por la SRS). Esto depende de que cada requisito en la SRS tenga un nombre o número de referencia único.

La trazabilidad hacia adelante de la SRS es especialmente importante cuando el producto de software entra en la fase de operación y mantenimiento. A medida que se modifican los documentos de código y diseño, es esencial poder determinar el conjunto completo de requisitos que pueden verse afectados por esas modificaciones.

4.4 Preparación conjunta de la SRS

El proceso de desarrollo de software debería comenzar con un acuerdo entre el proveedor y el cliente sobre lo que el software completo debe hacer. Este acuerdo, en forma de una SRS, debería prepararse conjuntamente. Esto es importante porque generalmente ni el cliente ni el proveedor están calificados para redactar una buena SRS por sí solos.

a) Los clientes generalmente no comprenden lo suficiente el proceso de diseño y desarrollo de software como para redactar una SRS utilizable.
b) Los proveedores generalmente no comprenden lo suficiente el problema del cliente y el campo de actuación como para especificar los requisitos para un sistema satisfactorio.

Por lo tanto, el cliente y el proveedor deberían trabajar juntos para producir una SRS bien redactada y completamente comprendida.

Existe una situación especial cuando un sistema y su software se definen simultáneamente. En este caso, la funcionalidad, interfaces, rendimiento y otros atributos y restricciones del software no están predefinidos, sino que se definen conjuntamente y están sujetos a negociación y cambios. Esto hace que sea más difícil, pero no menos importante, cumplir con las características establecidas en 4.3. En particular, una SRS que no cumple con los requisitos de su especificación de sistema principal es incorrecta.

Esta práctica recomendada no aborda específicamente el estilo, el uso del lenguaje o las técnicas de buena redacción. Sin embargo, es muy importante que una SRS esté bien escrita. Se pueden utilizar libros generales de redacción técnica como guía.

4.5 Evolución de la SRS

La SRS puede necesitar evolucionar a medida que avanza el desarrollo del producto de software. Puede ser imposible especificar algunos detalles en el momento en que se inicia el proyecto (por ejemplo, puede ser imposible definir todos los formatos de pantalla para un programa interactivo durante la fase de requisitos). Pueden surgir cambios adicionales a medida que se descubren deficiencias, limitaciones y errores en la SRS.

Dos consideraciones importantes en este proceso son las siguientes:

a) Los requisitos deben especificarse tan completamente y a fondo como se conozca en ese momento, incluso si se prevén revisiones evolutivas como inevitables. Debe señalarse el hecho de que son incompletos.
b) Debería iniciarse un proceso formal de cambio para identificar, controlar, seguir y informar sobre los cambios proyectados. Los cambios aprobados en los requisitos deben incorporarse en la SRS de manera que:
    1) Proporcione una traza de auditoría precisa y completa de los cambios;
    2) Permita la revisión de las partes actuales y descontinuadas de la SRS.

4.6 Prototipado

El prototipado se utiliza con frecuencia durante la fase de requisitos de un proyecto. Existen muchas herramientas que permiten crear un prototipo, que exhibe algunas características de un sistema, de manera rápida y sencilla. Consulte también ASTM E1340-96.

Los prototipos son útiles por las siguientes razones:

a) Es más probable que el cliente observe el prototipo y reaccione ante él que leer el SRS y reaccionar ante él. Por lo tanto, el prototipo proporciona retroalimentación rápida.
b) El prototipo muestra aspectos no anticipados del comportamiento del sistema. Así, produce no solo respuestas sino también nuevas preguntas. Esto ayuda a llegar a un cierre en el SRS.
c) Un SRS basado en un prototipo tiende a experimentar menos cambios durante el desarrollo, acortando así el tiempo de desarrollo.

Un prototipo debe utilizarse como una forma de obtener los requisitos de software. Algunas características, como el formato de pantalla o informe, se pueden extraer directamente del prototipo. Otros requisitos se pueden inferir mediante la realización de experimentos con el prototipo.

4.7 Incorporación del diseño en el SRS

Un requisito especifica una función o atributo externamente visible de un sistema. Un diseño describe un subcomponente específico de un sistema y/o sus interfaces con otros subcomponentes. Los redactores del SRS deben distinguir claramente entre identificar las restricciones de diseño necesarias y proyectar un diseño específico. Tenga en cuenta que cada requisito en el SRS limita las alternativas de diseño. Esto no significa, sin embargo, que cada requisito sea un diseño.

El SRS debería especificar qué funciones se realizarán en qué datos para producir qué resultados en qué ubicación para quién. El SRS debería centrarse en los servicios a realizar. Normalmente, el SRS no debe especificar elementos de diseño como los siguientes:

a) Particionar el software en módulos;
b) Asignar funciones a los módulos;
c) Describir el flujo de información o control entre módulos;
d) Elegir estructuras de datos.

4.7.1 Requisitos de diseño necesarios

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

a) Mantener ciertas funciones en módulos separados;
b) Permitir solo una comunicación limitada entre algunas áreas del programa;
c) Verificar la integridad de los datos para variables críticas.

Ejemplos de restricciones de diseño válidas son requisitos físicos, requisitos de rendimiento, estándares de desarrollo de software y estándares de aseguramiento de calidad del software.

Por lo tanto, los requisitos deben declararse desde un punto de vista puramente externo. Al utilizar modelos para ilustrar los requisitos, recuerde que el modelo solo indica el comportamiento externo y no especifica un diseño.

4.8 Incorporación de requisitos del proyecto en el SRS

El SRS debería abordar el producto de software, no el proceso de producción del producto de software.

Los requisitos del proyecto representan un entendimiento entre el cliente y el proveedor sobre asuntos contractuales relacionados con la producción de software y, por lo tanto, no deben incluirse en el SRS. Estos incluyen normalmente elementos como

a) Costo;
b) Cronogramas de entrega;
c) Procedimientos de informes;
d) Métodos de desarrollo de software;
e) Aseguramiento de calidad;
f) Criterios de validación y verificación;
g) Procedimientos de aceptación.

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

5. Las partes de un SRS

Esta cláusula discute cada una de las partes esenciales del SRS. Estas partes están dispuestas en la Figura 1 en un esquema que puede servir como ejemplo para escribir un SRS.

Si bien un SRS no tiene que seguir este esquema ni usar los nombres dados aquí para sus partes, un buen SRS debería incluir toda la información discutida aquí.

Tabla de Contenido

1. Introducción
   1.1 Propósito
   1.2 Alcance
   1.3 Definiciones, acrónimos y abreviaturas
   1.4 Referencias
   1.5 Descripción general
2. Descripción general
   2.1 Perspectiva del producto
   2.2 Funciones del producto
   2.3 Características del usuario
   2.4 Restricciones
   2.5 Suposiciones y dependencias
3. Requisitos específicos (Consulte 5.3.1 a 5.3.8 para explicaciones de posibles requisitos específicos. Consulte también el Anexo A para varias formas diferentes de organizar esta sección del SRS.)

Apéndice

Índice

Figura 1: Esquema del Prototipo de la SRS

5.1 Introducción (Sección 1 de la SRS)

La introducción de la SRS debería ofrecer una visión general de toda la SRS. Debería contener las siguientes subsecciones:

a) Propósito;
b) Alcance;
c) Definiciones, acrónimos y abreviaturas;
d) Referencias;
e) Visión general.

5.1.1 Propósito (1.1 del SRS)

Esta subsección debe

a) Delimitar el propósito del SRS;
b) Especificar la audiencia prevista para el SRS.

5.1.2 Alcance (1.2 del SRS)

Esta subsección debe

a) Identificar el o los productos de software que se producirán por nombre (por ejemplo, Host DBMS, Generador de informes, etc.).
b) Explicar lo que hará y, si es necesario, lo que no hará el producto de software.
c) Describir la aplicación del software que se está especificando, incluyendo los beneficios, objetivos y metas relevantes.
d) Ser coherente con declaraciones similares en especificaciones de nivel superior (por ejemplo, la especificación de requisitos del sistema), si existen.

5.1.3 Definiciones, acrónimos y abreviaturas (1.3 del SRS)

Esta subsección debe proporcionar las definiciones de todos los términos, acrónimos y abreviaturas necesarios para interpretar adecuadamente el SRS. Esta información puede proporcionarse haciendo referencia a uno o más apéndices en el SRS o haciendo referencia a otros documentos.

5.1.4 Referencias (1.4 del SRS)

Esta subsección debe

a) Proporcionar una lista completa de todos los documentos referenciados en otras partes del SRS;
b) Identificar cada documento por título, número de informe (si corresponde), fecha y organización editora;
c) Especificar las fuentes de las cuales se pueden obtener las referencias.

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

5.1.5 Resumen (1.5 del SRS)

Esta subsección debe

a) Describir lo que el resto del SRS contiene;
b) Explicar cómo está organizado el SRS.

5.2 Descripción general (Sección 2 del SRS)

Esta sección del SRS debe describir los factores generales que afectan al producto y sus requisitos. Esta sección no establece requisitos específicos. En cambio, proporciona un contexto para esos requisitos, que se definen en detalle en la Sección 3 del SRS, y los hace más fáciles de entender.

Esta sección generalmente consta de seis subsecciones, de la siguiente manera:

a) Perspectiva del producto;
b) Funciones del producto;
c) Características del usuario;
d) Restricciones;
e) Suposiciones y dependencias;
f) Distribución de requisitos.

5.2.1 Perspectiva del producto (2.1 del SRS)

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

Un diagrama de bloques que muestre los componentes principales del sistema más grande, las interconexiones y las interfaces externas puede ser útil.

Esta subsección también debería describir cómo opera el software dentro de varias restricciones. Por ejemplo, estas restricciones podrían incluir

a) Interfaces del sistema;
b) Interfaces de usuario;
c) Interfaces de hardware;
d) Interfaces de software;
e) Interfaces de comunicación;
f) Memoria;
g) Operaciones;
h) Requisitos de adaptación al sitio.

5.2.1.1 Interfaces del sistema

Esto debería enumerar cada interfaz del sistema e identificar la funcionalidad del software para cumplir con el requisito del sistema y la descripción de la interfaz para que coincida con el sistema.

5.2.1.2 Interfaces de usuario

Esto debería especificar lo siguiente:

a) Las características lógicas de cada interfaz entre el producto de software y sus usuarios. Esto incluye las características de configuración (por ejemplo, formatos de pantalla requeridos, disposición de páginas o ventanas, contenido de informes o menús, o disponibilidad de teclas de función programables) necesarias para cumplir con los requisitos de software.

b) Todos los aspectos de optimización de la interfaz con la persona que debe utilizar el sistema. Esto puede incluir simplemente una lista de recomendaciones y prohibiciones sobre cómo aparecerá el sistema para el usuario. Un ejemplo podría ser un requisito para la opción de mensajes de error largos o cortos. Al igual que todos los demás, estos requisitos deben ser verificables, por ejemplo, "un mecanógrafo de grado 4 puede realizar la función X en Z minutos después de 1 hora de capacitación" en lugar de "un mecanógrafo puede realizar la función X". (Esto también puede especificarse en los Atributos del Sistema de Software en una sección titulada Facilidad de Uso).

5.2.1.3 Interfaces de hardware

Esto debe especificar las características lógicas de cada interfaz entre el producto de software y los componentes de hardware del sistema. Esto incluye características de configuración (número de puertos, conjuntos de instrucciones, etc.). También aborda cuestiones como qué dispositivos se deben admitir, cómo se deben admitir y los protocolos. Por ejemplo, el soporte de terminales puede especificar el soporte de pantalla completa en lugar de soporte de línea por línea.

5.2.1.4 Interfaces de software

Esto debe especificar el uso de otros productos de software requeridos (por ejemplo, un sistema de gestión de datos, un sistema operativo o un paquete matemático) e interfaces con otros sistemas de aplicaciones (por ejemplo, la conexión entre un sistema de cuentas por cobrar y un sistema de contabilidad general). Para cada producto de software requerido, se debe proporcionar lo siguiente:

— Nombre;
— Mnemotecnia;
— Numero de especificacion
— Número de Versión
— Fuente.

Para cada interfaz, se debe proporcionar lo siguiente:

- Discusión sobre el propósito del software de interfaz en relación con este producto de software.
- Definición de la interfaz en términos de contenido y formato del mensaje. No es necesario detallar cualquier interfaz bien documentada, pero se requiere una referencia al documento que define la interfaz.

5.2.1.5 Interfaces de comunicación

Esto debería especificar las diversas interfaces de comunicación, como protocolos de red local, etc.

5.2.1.6 Restricciones de memoria

Esto debería especificar las características y límites aplicables a la memoria principal y secundaria.

5.2.1.7 Operaciones

Esto debería especificar las operaciones normales y especiales requeridas por el usuario, como

a) Los diversos modos de operación en la organización del usuario (por ejemplo, operaciones iniciadas por el usuario);
b) Períodos de operaciones interactivas y períodos de operaciones no supervisadas;
c) Funciones de soporte para el procesamiento de datos;
d) Operaciones de respaldo y recuperación.

NOTA: Esto a veces se especifica como parte de la sección de Interfaces de Usuario.

5.2.1.8 Requisitos de adaptación al sitio

Esto debería:

a) Definir los requisitos para cualquier secuencia de datos o inicialización específica para un sitio, misión o modo operativo dado (por ejemplo, valores de la cuadrícula, límites de seguridad, etc.).
b) Especificar las características relacionadas con el sitio o la misión que deben modificarse para adaptar el software a una instalación particular.

5.2.2 Funciones del producto (Sección 2.2 del SRS)

Esta subsección del SRS debería proporcionar un resumen de las funciones principales que el software llevará a cabo. Por ejemplo, un SRS para un programa de contabilidad puede utilizar esta parte para abordar el mantenimiento de cuentas de clientes, la preparación de estados de cuenta y facturas sin mencionar la vasta cantidad de detalles que cada una de esas funciones requiere.

A veces, el resumen de funciones necesario para esta parte se puede tomar directamente de la sección de la especificación de nivel superior (si existe) que asigna funciones específicas al producto de software. Cabe señalar que, por claridad y simplicidad, las funciones deben organizarse de una manera que tenga sentido para el usuario y el cliente, no para el desarrollador de software. Cada función mencionada en esta sección debe ser rastreable hasta una descripción más detallada en la Sección 3.

a) Las funciones deben organizarse de manera que la lista de funciones sea comprensible para el cliente o cualquier otra persona que lea el documento por primera vez.

b) Se pueden utilizar métodos textuales o gráficos para mostrar las diferentes funciones y sus relaciones. Este tipo de diagrama no tiene la intención de mostrar un diseño del producto, sino simplemente muestra las relaciones lógicas entre variables.

5.2.3 Características del usuario (2.3 del SRS)

Esta subsección del SRS debe describir las características generales de los usuarios previstos del producto, incluido el nivel educativo, la experiencia y la experiencia técnica. No debe utilizarse para establecer requisitos específicos, sino más bien para proporcionar las razones por las cuales se especifican ciertos requisitos específicos más adelante en la Sección 3 del SRS.

5.2.4 Restricciones (2.4 del SRS)

Esta subsección del SRS debe proporcionar una descripción general de cualquier otro elemento que limite las opciones del desarrollador. Esto incluye

a) Políticas regulatorias;
b) Limitaciones de hardware (por ejemplo, requisitos de temporización de señales);
c) Interfaces con otras aplicaciones;
d) Operación en paralelo;
e) Funciones de auditoría;
f) Funciones de control;
g) Requisitos de lenguaje de alto nivel;
h) Protocolos de handshake de señales (por ejemplo, XON-XOFF, ACK-NACK);
i) Requisitos de confiabilidad;
j) Criticidad de la aplicación;
k) Consideraciones de seguridad y protección.

5.2.5 Suposiciones y dependencias (2.5 del SRS)

Esta subsección del SRS debe enumerar cada uno de los factores que afectan los requisitos establecidos en el SRS. Estos factores no son restricciones de diseño en el software, sino cualquier cambio en ellos que pueda 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 de software. Si, de hecho, el sistema operativo no está disponible, el SRS tendría que cambiar en consecuencia.

5.2.6 Distribución de requisitos (2.6 del SRS)

Esta subsección del SRS debe identificar los requisitos que pueden posponerse hasta futuras versiones del sistema.

5.3 Requisitos específicos (Sección 3 del SRS)

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

a) Los requisitos específicos deben declararse de conformidad con todas las características descritas en 4.3.
b) Los requisitos específicos deben hacer referencia cruzada a documentos anteriores relacionados.
c) Todos los requisitos deben ser identificables de manera única.
d) Se debe prestar atención cuidadosa a la organización de los requisitos para maximizar la legibilidad.

Antes de examinar formas específicas de organizar los requisitos, es útil entender los diversos elementos que componen los requisitos, como se describe en 5.3.1 a 5.3.7.

5.3.1 Interfaces externas

Esto debería ser una descripción detallada de todas las entradas y salidas del sistema de software. Debería complementar las descripciones de interfaz en 5.2 y no repetir información allí.

Debe incluir tanto el contenido como el formato de la siguiente manera:

a) Nombre del elemento;
b) Descripción del propósito;
c) Fuente de entrada o destino de salida;
d) Rango válido, precisión y/o tolerancia;
e) Unidades de medida;
f) Temporización;
g) Relaciones con otras entradas/salidas;
h) Formatos/organización de pantalla;
i) Formatos/organización de ventana;
j) Formatos de datos;
k) Formatos de comandos;
l) Mensajes de finalización.

5.3.2 Funciones

Los requisitos funcionales deben definir las acciones fundamentales que deben ocurrir en el software al aceptar y procesar las entradas, así como al procesar y generar las salidas. Por lo general, se enumeran como declaraciones de "deberá" que comienzan con "El sistema deberá..."

Estos incluyen

a) Comprobaciones de validez en las entradas
b) Secuencia exacta de operaciones
c) Respuestas a situaciones anormales, incluyendo
    1) Desbordamiento
    2) Facilidades de comunicación
    3) Manejo de errores de recuperación
d) Efecto de los parámetros
e) Relación de las salidas con las entradas, incluyendo
    1) Secuencias de entrada/salida
    2) Fórmulas para la conversión de entrada a salida

Puede ser apropiado dividir los requisitos funcionales en subfunciones o subprocesos. Esto no implica que el diseño del software también se dividirá de esa manera.

5.3.3 Requisitos de rendimiento

Esta subsección debe especificar tanto los requisitos numéricos estáticos como dinámicos impuestos al software o a la interacción humana con el software en su conjunto. Los requisitos numéricos estáticos pueden incluir lo siguiente:

a) La cantidad de terminales que se deben admitir;
b) La cantidad de usuarios simultáneos que se deben admitir;
c) Cantidad y tipo de información que se debe manejar.

Los requisitos numéricos estáticos a veces se identifican en una sección separada titulada Capacidad.

Los requisitos numéricos dinámicos pueden incluir, por ejemplo, la cantidad de transacciones y tareas, y la cantidad de datos a procesar en ciertos períodos de tiempo tanto para condiciones de carga normales como para condiciones de carga máxima.

Todos estos requisitos deben expresarse en términos medibles.

Por ejemplo,

El 95% de las transacciones deberá procesarse en menos de 1 segundo.

en lugar de,

Un operador no deberá esperar a que se complete la transacción.

NOTA: Los límites numéricos aplicados a una función específica generalmente se especifican como parte de la descripción de ese párrafo de procesamiento de la función.

5.3.4 Requisitos lógicos de la base de datos

Esto debería especificar los requisitos lógicos para cualquier información que se vaya a colocar en una base de datos. Esto puede incluir lo siguiente:

a) Tipos de información utilizados por diversas funciones;
b) Frecuencia de uso;
c) Capacidades de acceso;
d) Entidades de datos y sus relaciones;
e) Restricciones de integridad;
f) Requisitos de retención de datos.

5.3.5 Restricciones de diseño

Esto debe especificar restricciones de diseño impuestas por otros estándares, limitaciones de hardware, etc.

5.3.5.1 Cumplimiento de estándares

Esta subsección debe especificar los requisitos derivados de estándares o regulaciones existentes. Pueden incluir lo siguiente:

a) Formato de informe;
b) Nomenclatura de datos;
c) Procedimientos contables;
d) Rastreo de auditoría.

Por ejemplo, esto podría especificar el requisito de que el software trace la actividad de procesamiento. Se necesitan tales trazas para que algunas aplicaciones cumplan con los estándares regulatorios o financieros mínimos. Un requisito de rastreo de auditoría puede, por ejemplo, indicar que todos los cambios en una base de datos de nóminas deben registrarse en un archivo de trazas con valores antes y después.

5.3.6 Atributos del sistema de software

Existen varios atributos del software que pueden servir como requisitos. Es importante que se especifiquen los atributos requeridos para que su logro pueda ser verificado objetivamente. Los subpárrafos 5.3.6.1 a 5.3.6.5 proporcionan una lista parcial de ejemplos.

5.3.6.1 Confiabilidad

Esto debería especificar los factores necesarios para establecer la confiabilidad requerida del sistema de software en el momento de la entrega.

5.3.6.2 Disponibilidad

Esto debería especificar los factores necesarios para garantizar un nivel de disponibilidad definido para todo el sistema, como checkpoints, recuperación y reinicio.

5.3.6.3 Seguridad

Esto debería especificar los factores que protegen el software contra el acceso, uso, modificación, destrucción o divulgación accidental o maliciosa. Los requisitos específicos en esta área podrían incluir la necesidad de

a) Utilizar ciertas técnicas criptográficas;
b) Mantener conjuntos de datos de registro o historial específicos;
c) Asignar ciertas funciones a diferentes módulos;
d) Restringir comunicaciones entre algunas áreas del programa;
e) Verificar la integridad de los datos para variables críticas.

5.3.6.4 Mantenibilidad

Esto debería especificar atributos del software que se relacionen con la facilidad de mantenimiento del propio software. Puede haber algún requisito para cierta modularidad, interfaces, complejidad, etc. No se deben incluir requisitos aquí solo porque se considere que son buenas prácticas de diseño.

5.3.6.5 Portabilidad

Esto debería especificar atributos del software que se relacionen con la facilidad de trasladar el software a otras máquinas anfitrionas y/o sistemas operativos. Esto puede incluir lo siguiente:

a) Porcentaje de componentes con código dependiente del anfitrión;
b) Porcentaje de código que es dependiente del anfitrión;
c) Uso de un lenguaje portátil probado;
d) Uso de un compilador o subconjunto de lenguaje específico;
e) Uso de un sistema operativo específico.

5.3.7 Organización de los requisitos específicos

Para cualquier sistema que no sea trivial, los requisitos detallados tienden a ser extensos. Por esta razón, se recomienda dar una cuidadosa consideración para organizarlos de una manera óptima para su comprensión. No hay una organización óptima única para todos los sistemas. Diferentes clases de sistemas se prestan a diferentes organizaciones de requisitos en la Sección 3 del SRS. Algunas de estas organizaciones se describen en 5.3.7.1 hasta 5.3.7.7.

5.3.7.1 Modo del sistema

Algunos sistemas se comportan de manera bastante diferente según el modo de operación. Por ejemplo, un sistema de control puede tener diferentes conjuntos de funciones según su modo: entrenamiento, normal o emergencia. Al organizar esta sección por modo, se debe utilizar el esquema en A.1 o A.2. La elección depende de si las interfaces y el rendimiento dependen del modo.

5.3.7.2 Clase de usuario

Algunos sistemas ofrecen diferentes conjuntos de funciones a diferentes clases de usuarios. Por ejemplo, un sistema de control de ascensores presenta diferentes capacidades a pasajeros, trabajadores de mantenimiento y bomberos. Al organizar esta sección por clase de usuario, se debe utilizar el esquema en A.3.

5.3.7.3 Objetos

Los objetos son entidades del mundo real que tienen un equivalente dentro del sistema. Por ejemplo, en un sistema de monitoreo de pacientes, los objetos incluyen pacientes, sensores, enfermeras, habitaciones, médicos, medicamentos, etc. Asociados con cada objeto hay un conjunto de atributos (de ese objeto) y funciones (realizadas por ese objeto). Estas funciones también se llaman servicios, métodos o procesos. Al organizar esta sección por objeto, se debe utilizar el esquema en A.4. Tenga en cuenta que conjuntos de objetos pueden compartir atributos y servicios. Estos se agrupan como clases.

5.3.7.4 Característica

Una característica es un servicio externamente deseado por el sistema que puede requerir una secuencia de entradas para lograr el resultado deseado. Por ejemplo, en un sistema telefónico, las características incluyen llamada local, desvío de llamadas y conferencia telefónica. Generalmente, cada característica se describe en una secuencia de pares estímulo-respuesta. Al organizar esta sección por característica, se debe utilizar el esquema en A.5.

5.3.7.5 Estímulo

Algunos sistemas se pueden organizar mejor describiendo sus funciones en términos de estímulos. Por ejemplo, las funciones de un sistema automático de aterrizaje de aeronaves pueden organizarse en secciones para pérdida de energía, cizalladura del viento, cambio repentino de balanceo, velocidad vertical excesiva, etc. Al organizar esta sección por estímulo, se debe utilizar el esquema en A.6.

5.3.7.6 Respuesta

Algunos sistemas se pueden organizar mejor describiendo todas las funciones en apoyo a la generación de una respuesta. Por ejemplo, las funciones de un sistema de personal pueden organizarse en secciones correspondientes a todas las funciones asociadas con la generación de cheques de pago, todas las funciones asociadas con la generación de una lista actual de empleados, etc. Se debe utilizar el esquema en A.6 (con todas las ocurrencias de estímulo reemplazadas por respuesta).

5.3.7.7 Jerarquía funcional

Cuando ninguno de los esquemas organizativos anteriores resulta útil, la funcionalidad general se puede organizar en una jerarquía de funciones organizadas por entradas comunes, salidas comunes o acceso común a datos internos. Los diagramas de flujo de datos y los diccionarios de datos se pueden utilizar para mostrar las relaciones entre y entre las funciones y los datos. Al organizar esta sección por jerarquía funcional, se debe utilizar el esquema en A.7.

5.3.8 Comentarios adicionales

Cuando se contempla un nuevo SRS, más de una de las técnicas organizativas dadas en 5.3.7.7 puede ser apropiada. En tales casos, organice los requisitos específicos para múltiples jerarquías adaptadas a las necesidades específicas del sistema bajo especificación. Por ejemplo, consulte A.8 para una organización que combina clase de usuario y característica. Cualquier requisito adicional puede colocarse en una sección separada al final del SRS.

Existen muchas notaciones, métodos y herramientas de soporte automatizado disponibles para ayudar en la documentación de requisitos. En su mayoría, su utilidad es una función de la organización. Por ejemplo, cuando se organiza por modo, las máquinas de estados finitos o los diagramas de estados pueden resultar útiles; al organizar por objeto,

el análisis puede resultar útil; al organizar por característica, las secuencias de estímulo-respuesta pueden resultar útiles; y al organizar por jerarquía funcional, los diagramas de flujo de datos y los diccionarios de datos pueden resultar útiles.

En cualquiera de los esquemas dados en A.1 a través de A.8, aquellas secciones llamadas "Requisito funcional i" pueden describirse en el lenguaje nativo (por ejemplo, inglés), en seudocódigo, en un lenguaje de definición de sistema o en cuatro subsecciones tituladas: Introducción, Entradas, Procesamiento y Salidas.

5.4 Información de apoyo

La información de apoyo facilita el uso del SRS. Incluye lo siguiente:

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

5.4.1 Índice y tabla de contenido

El índice y la tabla de contenido son bastante importantes y deben seguir prácticas de composición general.

5.4.2 Apéndices

Los apéndices no siempre se consideran parte del SRS real y no siempre son necesarios. Pueden incluir

a) Formatos de entrada/salida de ejemplo, descripciones de estudios de análisis de costos o resultados de encuestas de usuarios;
b) Información de apoyo o antecedentes que pueda ayudar a los lectores del SRS;
c) Una descripción de los problemas que debe resolver el software;
d) Instrucciones de empaque especiales para el código y los medios para cumplir con requisitos de seguridad, exportación, carga inicial u otros requisitos.

Cuando se incluyen apéndices, el SRS debe indicar explícitamente si se deben considerar o no parte de los requisitos.

Anexo A

(informativo)

Plantillas de SRS

A.1 Plantilla de la Sección 3 del SRS organizada por modo: Versión 1

3. Requisitos Especificos
3.1 Requisitos de interfaz externa
- 3.1.1 Interfaces de usuario
- 3.1.2 Interfaces de hardware
- 3.1.3 Interfaces de software
- 3.1.4 Interfaces de comunicación

3.2 Requisitos funcionales
- 3.2.1 Modo 1
  - 3.2.1.1 Requisito funcional 1.1
  - .
  - .
  - .
  - 3.2.1.n Requisito funcional 1.n
- 3.2.2 Modo 2
- .
- .
- .
- *3.2.m* Modo *m*
  - 3.2.m.1 Requisitos funcionales m.1
  - .
  - .
  - .
  - *3.2.m.n Requisitos funcionales m.n*

3.3 Requisitos de rendimiento
3.4 Restricciones de diseño
3.5 Atributos del sistema de software
3.6 Otros requisitos

A.2 Plantilla de la Sección 3 del SRS organizada por modo: Versión 2

3. Requisitos Específicos
3.1. Requisitos Funcionales
- 3.1.1 Modo 1
  - 3.1.1.1 Interfaces Externas
    - 3.1.1.1.1 Interfaces de usuario
    - 3.1.1.1.2 Interfaces de hardware
    - 3.1.1.1.3 Interfaces de software
    - 3.1.1.1.4 Interfaces de comunicación
  - 3.1.1.2 Requisitos Funcinales
    - 3.1.1.2.1 Requisito Funcional 1
    - .
    - .
    - .

3.1.1.2.*n* Requisito Funcional *n*
3.1.1.3 Rendimiento
3.1.2 Modo 2
.
.
.
3.1.*m* Modo *m*
3.2 Restricciones de diseño
3.3 Atributos del sistema de software
3.4 Otros requisitos

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

3. Requisitos Especificos
3.1 Requisitos de interfaz externa
3.1.1 Interfaces de usuario
3.1.2 Interfaces de hardware
3.1.3 Interfaces de software
3.1.4 Interfaces de comunicación
3.2 Requisitos Funcionales
3.2.1 Usuario clase 1
3.2.1.1 Requisito funcional 1.1
.
.
.
3.2.1.n Requisito funcional 1.n
3.2.2 Usuario clase 2
.
.
.
3.2.m Usuario clase m
3.2.m.1 Requisito funcional m.1
.
.
.
3.2.m.n Requisito funcional m.n
3.3 Requisitos de rendimiento
3.4 Restricciones de diseño
3.5 Atributos del sistema de software
3.6 Otros requisitos

A.4 Plantilla de la Sección 3 de la SRS organizada por objecto

3. Requisitos Especifícos
3.1 Requisitos de interfaz externa
3.1.1 Interfaces de usuario
3.1.2 Interfaces de hardware
3.1.3 Interfaces de software
3.1.4 Interfaces de comunicación
3.2 Clases/Obje1os
3.2.1 Clase/Objeto 1

3.2.1.1 Atributos (directos o heredados)
3.2.1.1.1 A1ributo 1
.
.
.
3.2.1.1.*n* A1ribu1o *n*
3.2.1.2 Funciones (servicios, me1odos, directos o heredados)
3.2.1.2.1 Requisito Funcional 1.1 .
.
.
3.2.1.2.*m* Requisito Funcional 1.*m*

3.2.1.3 Mensajes (comunicaciones recibidas o enviadas)
3.2.2 Clase/Objeto 2
.
.
.
3.2.*p* *Clase/Objeto p*
3.3 Requisitos de rendimiento
3.4 Restricciones de diseño
3.5 Atributos del sistema de software
3.6 Otros requisitos

A.5 Plantilla de la Sección 3 de la SRS organizada por características

3. Requisitos Especifícos
3.1 Requisitos de interfaz externa
3.1.1 Interfaces de usuario
3.1.2 Interfaces de hardware
3.1.3 Interfaces de software
3.1.4 Interfaces de comunicación
3.2 Características del Sistema
3.2.1 Característica 1
3.2.1.1 Introducción/Propósito de la característica
3.2.1.2 Secuencia Estímulo/Respuesta
3.2.1.3 Requisitos funcionales asociados
3.2.1.3.1 Requisito funcional 1
.
.
.
3.2.1.3.n Requisito funcional n
3.2.2 Característica 2
.
.
.
3.2.*m* Característica *m*
.
.
.
3.3 Requisitos de rendimiento
3.4 Restricciones de diseño
3.5 Atributos del sistema de software
3.6 Otros requisitos

A.6 Plantilla de la Sección 3 de la SRS organizada por estímulos

3. Requisitos Especifícos
3.1 Requisitos de interfaz externa
3.1.1 Interfaces de usuario
3.1.2 Interfaces de hardware
3.1.3 Interfaces de software
3.1.4 Interfaces de comunicación
3.2 Requisitos Funcionales
3.2.1 Estímulo 1
3.2.1.1 Requisito Funcional 1.1
.
.
.
3.2.1.*n* Requisito Funcional 1.*n*
3.2.2 Es1imulo 2
.
.
.
3.2.m Estimulo m
3.2.m.1 Requisito Funcional *m*.1
.
.
.
3.2.*m*.*n* Requisito Funcional *m*.*n*
3.3
3.4 Requisitos de rendimiento
3.5 Restricciones de diseño
3.6 Atributos del sistema de software
Otros requisitos

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

3. Requisitos Especifícos
3.1 Requisitos de interfaz externa
3.1.1 Interfaces de usuario
3.1.2 Interfaces de hardware
3.1.3 Interfaces de software
3.1.4 Interfaces de comunicación
3.2 Requisitos Funcionales
3.2.1 Flujo de información
3.2.1.1 Diagrama de flujo de datos 1
3.2.1.1.1 Entidades de datos
3.2.1.1.2 Procesos pertinentes
3.2.1.1.3 Topología
3.2.1.2 Diagrama de flujo de datos 2
3.2.1.2.1 Entidades de datos
3.2.1.2.2 Procesos pertinentes
3.2.1.2.3 Topología
.
.
.
3.2.1.n Diagrama de flujo de datos *n*

3.2.1.n.1 Entidades de datos
3.2.1.n.2 Procesos pertinentes
3.2.1.n.3 Topología

3.2.2 Descripción de Procecsos

3.2.2.1 Proceso 1
3.2.2.1.1 Entidades de datos de entrada
3.2.2.1.2 Algoritmo o fórmula del proceso
3.2.2.1.3 Entidades de datos afectadas

3.2.2.2 Proceso 2
3.2.2.2.1 Entidades de datos de entrada
3.2.2.2.2 Algoritmo o fórmula del proceso
3.2.2.2.3 Entidades de datos afectadas

.
.
.

3.2.2.m Proceso *m*
3.2.2.m.1 Entidades de datos de entrada
3.2.2.m.2 Algoritmo o fórmula del proceso
3.2.2.m.3 Entidades de datos afectadas

3.2.3 Especificaciones de construcción de datos
3.2.3.1 Construcción 1
3.2.3.1.1 Tipo de registro
3.2.3.1.2 Campos constituyentes
3.2.3.2 Construcción 2
3.2.3.2.1 Tipo de registro
3.2.3.2.2 Campos constituyentes

.
.
.

3.2.3.p Cons1rucción *p*
3.2.3.p.1 Tipo de resgistro
3.2.3.p.2 Campos constituyentes

3.2.4 Diccionario de datos

3.2.4.1 Elemento de datos 1
3.2.4.1.1 Nombre
3.2.4.1.2 Representación
3.2.4.1.3 Unidades/Formato
3.2.4.1.4 Precisión/Exactitud
3.2.4.1.5 Rango

3.2.4.2 Elemento de datos 1
3.2.4.2.1 Nombre
3.2.4.2.2 Representación
3.2.4.2.3 Unidades/Formato
3.2.4.2.4 Precisión/Exactitud
3.2.4.2.5 Rango

.
.
.

3.2.4.q Elemento de datos q
3.2.4.q.1 Nombre
3.2.4.q.2 Representación
3.2.4.q.3 Unidades/Formato
3.2.4.q.4 Precisión/Exactitud
3.2.4.q.5 Rango

3.3 Interfaces de usuario
3.4 Interfaces de hardware
3.5 Interfaces de software
3.6 Otros requisitos

A.8 Plantillas de la Sección 3 de la SRS mostrando múltiples organizaciones

3. Requisitos Especifícos

3.1 Requisitos de interfaz externa
- 3.1.1 Interfaces de usuario
- 3.1.2 Interfaces de hardware
- 3.1.3 Interfaces de software
- 3.1.4 Interfaces de comunicación

3.2 Requisitos Funcionales
- 3.2.1 Clase de Usuario 1
  - 3.2.1.1 Característica 1.1
    - 3.2.1.1.1 Introducción/Propósito de la característica
    - 3.2.1.1.2 Secuencia de Estímulo/Respuesta
    - 3.2.1.1.3 Requisitos funcionales asociados
  - 3.2.1.2 Característica 1.2
    - 3.2.1.2.1 Introducción/Propósito de la característica
    - 3.2.1.2.2 Secuencia de Estímulo/Respuesta
    - 3.2.1.2.3 Requisitos funcionales asociados
  - .
  - .
  - .
  - *3.2.1.m* Característica 1.*m*
    - 3.2.1.m.1 Introducción/Propósito de la característica
    - 3.2.1.m.2 Secuencia de Estímulo/Respuesta
    - 3.2.1.m.3 Requisitos funcionales asociados
- 3.6.1 Clase de usuario 2
- .
- .
- .
- *3.2.n* Clase de usuario *n*
- .
- .
- .

3.3 Requisitos de rendimiento
3.4 Restricciones de diseño
3.5 Atributos del sistema de software
3.6 Otros requisitos

Annex B

(informative)

Guidelines for compliance with IEEE/EIA 12207.1-1997

B.1 Overview

The Sof1ware Engineering S1andards Commi11ee (SESC) of 1he IEEE Compu1er Socie1y has endorsed 1he policy of adop1ing in1erna1ional s1andards. In 1995, 1he in1erna1ional s1andard, ISO/IEC 12207, Informa1ion 1echnology–Sof1ware life cycle processes, was comple1ed. The s1andard es1ablishes a common framework for sof1ware life cycle processes, wi1h well-defined 1erminology, 1ha1 can be referenced by 1he sof1ware indus1ry.

In 1995 1he SESC evalua1ed ISO/IEC 12207 and decided 1ha1 1he s1andard should be adop1ed and serve as 1he basis for life cycle processes wi1hin 1he IEEE Sof1ware Engineering Collec1ion. The IEEE adap1a1ion of ISO/IEC 12207 is IEEE/EIA 12207.0-1996. I1 con1ains ISO/IEC 12207 and 1he following addi1ions: improved compliance approach, life cycle process objec1ives, life cycle da1a objec1ives, and erra1a.

The implemen1a1ion of ISO/IEC 12207 wi1hin 1he IEEE also includes 1he following:

— IEEE/EIA 12207.1-1997, IEEE/EIA Guide for Informa1ion Technology–Sof1ware life cycle processes–Life cycle da1a;

— IEEE/EIA 12207.2-1997, IEEE/EIA Guide for Informa1ion Technology–Sof1ware life cycle processes–Implemen1a1ion considera1ions; and

– Addi1ions 1o 11 SESC s1andards (i.e., IEEE S1ds 730, 828, 829, 830, 1012, 1016, 1058, 1062, 1219, 1233, 1362) 1o define 1he correla1ion be1ween 1he da1a produced by exis1ing SESC s1andards and 1he da1a produced by 1he applica1ion of IEEE/EIA 12207.1-1997.

NOTE–Al1hough IEEE/EIA 12207.1-1997 is a guide, i1 also con1ains provisions for applica1ion as a s1andard wi1h specific compliance requiremen1s. This annex 1rea1s 12207.1-1997 as a s1andard.

B.1.1 Scope and purpose

Bo1h IEEE S1d 830-1998 and IEEE/EIA 12207.1-1997 place requiremen1s on a Sof1ware Requiremen1s Descrip1ion Documen1. The purpose of 1his annex is 1o explain 1he rela1ionship be1ween 1he 1wo se1s of requiremen1s so 1ha1 users producing documen1s in1ended 1o comply wi1h bo1h s1andards may do so.

B.2 Correlation

This clause explains 1he rela1ionship be1ween IEEE S1d 830-1998 and IEEE/EIA 12207.0-1996 and IEEE/EIA 12207.1-1997 in 1he following areas: 1erminology, process, and life cycle da1a.

B.2.1 Terminology correlation

Bo1h 1his recommended prac1ice and IEEE/EIA 12207.0-1996 have similar seman1ics for 1he key 1erms of sof1ware, requiremen1s, specifica1ion, supplier, developer, and main1ainer. This recommended prac1ice uses

1he 1erm “cus1omer” where IEEE/EIA 12207.0-1996 uses “acquirer,” and 1his recommended prac1ice uses “user” where IEEE/EIA 12207.0-1996 uses “opera1or.”

B.2.2 Process correlation

IEEE/EIA 12207.0-1996 uses a process-orien1ed approach for describing 1he defini1ion of a se1 of require-men1s for sof1ware. This recommended prac1ice uses a produc1-orien1ed approach, where 1he produc1 is a Sof1ware Requiremen1s Descrip1ion (SRD). There are na1ural process s1eps, namely 1he s1eps 1o crea1e each por1ion of 1he SRD. These may be correla1ed wi1h 1he process requiremen1s of IEEE/EIA 12207.0-1996. The difference is 1ha1 1his recommended prac1ice is focused on 1he developmen1 of sof1ware requiremen1s whereas IEEE/EIA 12207.0-1996 provides an overall life cycle view and men1ions Sof1ware Requiremen1s Analysis as par1 of i1s Developmen1 Process. This recommended prac1ice provides a grea1er level of de1ail on wha1 is involved in 1he prepara1ion of an SRD.

B.2.3 Life cycle data correlation

IEEE/EIA 12207.0-1996 1akes 1he viewpoin1 1ha1 1he sof1ware requiremen1s are derived from 1he sys1em requiremen1s. Therefore, i1 uses 1he 1erm, “descrip1ion” ra1her 1ha1 “specifica1ion” 1o describe 1he sof1ware requiremen1s. In a sys1em in which sof1ware is a componen1, each requiring i1s own specifica1ion, 1here would be a Sys1em Requiremen1s Specifica1ion (SRS) and one or more SRDs. If 1he 1erm Sof1ware Require-men1s Specifica1ion had been used, 1here would be a confusion be1ween an SRS referring 1o 1he sys1em or sof1ware requiremen1s. In 1he case where 1here is a s1and-alone sof1ware sys1em, IEEE/EIA 12207.1-1997 s1a1es “If 1he sof1ware is a s1and-alone sys1em, 1hen 1his documen1 should be a specifica1ion.”

B.3 Content mapping

This clause provides de1ails bearing on a claim 1ha1 an SRS complying wi1h 1his recommended prac1ice would also achieve “documen1 compliance” wi1h 1he SRD described in IEEE/EIA 12207.1-1997. The requiremen1s for documen1 compliance are summarized in a single row of Table 1 of IEEE/EIA 12207.1-1997. Tha1 row is reproduced in Table B.1 of 1his recommended prac1ice.

Table B.1—Summary of requirements for an SRD excerpted from Table 1 of IEEE/EIA 12207.1-1997

Information item	IEEE/EIA 12207.0-1996 Clause	Kind	IEEE/EIA 12207.1-1997 Clause	References
Sof1ware Requiremen1s Descrip1ion	5.1.1.4, 5.3.4.1, 5.3.4.2	Descrip1ion (See no1e for 6.22.1 of IEEE/EIA 12207.1-1997.)	6.22	IEEE S1d 830-1998; EIA/IEEE J-STD-016, F.2.3, F.2.4; MIL-STD 961D. Also see ISO/IEC 5806, 5807, 6593, 8631, 8790, and 11411 for guidance on use of no1a1ions.

The requiremen1s for documen1 compliance are discussed in 1he following subclauses:

— B.3.1 discusses compliance wi1h 1he informa1ion requiremen1s no1ed in column 2 of Table B.1 as prescribed by 5.1.1.4, 5.3.4.1, and 5.3.4.2 of IEEE/EIA 12207.0-1996.

— B.3.2 discusses compliance wi1h 1he generic con1en1 guideline (1he “kind” of documen1) no1ed in column 3 of Table B.1 as a “descrip1ion”. The generic con1en1 guidelines for a “descrip1ion” appear in 5.1 of IEEE/EIA 12207.1-1997.

— B.3.3 discusses compliance wi1h 1he specific requiremen1s for a Sof1ware Requiremen1s Descrip1ion no1ed in column 4 of Table B.1 as prescribed by 6.22 of IEEE/EIA 12207.1-1997.

— B.3.4 discusses compliance wi1h 1he life cycle da1a objec1ives of Annex H of IEEE/EIA 12207.0-1996 as described in 4.2 of IEEE/EIA 12207.1-1997.

B.3.1 Compliance with information requirements of IEEE/EIA 12207.0-1996

The informa1ion requiremen1s for an SRD are 1hose prescribed by 5.1.1.4, 5.3.4.1, and 5.3.4.2 of IEEE/EIA 12207.0-1996. The requiremen1s are subs1an1ively iden1ical 1o 1hose considered in B.3.3 of 1his recommended prac1ice.

B.3.2 Compliance with generic content guidelines of IEEE/EIA 12207.1-1997

According 1o IEEE/EIA 12207.1-1997, 1he generic con1en1 guideline for an SRD is generally a descrip1ion, as prescribed by 5.1 of IEEE/EIA 12207.1-1997. A complying descrip1ion shall achieve 1he purpose s1a1ed in 5.1.1 and include 1he informa1ion lis1ed in 5.1.2 of IEEE/EIA 12207.1-1997.

The purpose of a descrip1ion is:

> IEEE/EIA 12207.1-1997, subclause 5.1.1: Purpose: Describe a planned or ac1ual func1ion, design, performance, or process.

An SRD complying wi1h 1his recommended prac1ice would achieve 1he s1a1ed purpose.

Any descrip1ion or specifica1ion complying wi1h IEEE/EIA 12207.1-1997 shall sa1isfy 1he generic con1en1 requiremen1s provided in 5.1.2 of 1ha1 s1andard. Table B.2 of 1his recommended prac1ice lis1s 1he generic con1en1 i1ems and, where appropria1e, references 1he clause of 1his recommended prac1ice 1ha1 requires 1he same informa1ion.

Table B.2—Coverage of generic description requirements by IEEE Std 830-1998

IEEE/EIA 12207.1-1997 generic content	Corresponding clauses of IEEE Std 830-1998	Additions to requirements of IEEE Std 830-1998
a) Da1e of issue and s1a1us	—	Da1e of issue and s1a1us shall be provided.
b) Scope	5.1.1 Scope	—
c) Issuing organiza1ion	—	Issuing organiza1ion shall be iden1ified.
d) References	5.1.4 References	—
e) Con1ex1	5.1.2 Scope	—
f) No1a1ion for descrip1ion	4.3 Charac1eris1ics of a good SRS	—
g) Body	5. The par1s of an SRS	—
h) Summary	5.1.1. Overview	—
i) Glossary	5.1.3 Defini1ions	—
j) Change his1ory	—	Change his1ory for 1he SRD shall be provided or referenced.

B.3.3 Compliance with specific content requirements of IEEE/EIA 12207.1-1997

The specific con1en1 requiremen1s for an SRD in IEEE/EIA 12207.1-1997 are prescribed by 6.22 of IEEE/EIA 12207.1-1997. A complian1 SRD shall achieve 1he purpose s1a1ed in 6.22.1 of IEEE/EIA 12207.1-1997.

The purpose of 1he SRD is:

> IEEE/EIA 12207.1-1997, subclause 6.22.1: Purpose: Specify 1he requiremen1s for a sof1ware i1em and 1he me1hods 1o be used 1o ensure 1ha1 each requiremen1 has been me1. Used as 1he basis for design and qualifica1ion 1es1ing of a sof1ware i1em.

An SRS complying wi1h 1his recommended prac1ice and mee1ing 1he addi1ional requiremen1s of Table B.3 of 1his recommended prac1ice would achieve 1he s1a1ed purpose.

An SRD complian1 wi1h IEEE/EIA 12207.1-1997 shall sa1isfy 1he specific con1en1 requiremen1s provided in 6.22.3 and 6.22.4 of 1ha1 s1andard. Table B.3 of 1his recommended prac1ice lis1s 1he specific con1en1 i1ems and, where appropria1e, references 1he clause of 1his recommended prac1ice 1ha1 requires 1he same informa-1ion.

An SRD specified according 1he requiremen1s s1a1ed or referenced in Table B.3 of 1his recommended prac-1ice shall be evalua1ed considering 1he cri1eria provided in 5.3.4.2 of IEEE/EIA 12207.0-1996.

Table B.3—Coverage of specific SRD requirements by IEEE Std 830-1998

IEEE/EIA 12207.1-1997 specific content	Corresponding clauses of IEEE Std 830-1998	Additions to requirements of IEEE Std 830-1998
a) Generic descrip1ion informa1ion	See Table B.2	—
b) Sys1em iden1ifica1ion and overview	5.1.1 Scope	—
c) Func1ionali1y of 1he sof1ware i1em including: – Performance requiremen1s – Physical charac1eris1ics – Environmen1al condi1ions	5.3.2 Func1ions 5.3.3 Performance requiremen1s	Physical charac1eris1ics and environmen1al condi1ions should be provided.
d) Requiremen1s for in1erfaces ex1ernal 1o sof1ware i1em	5.3.1 Ex1ernal in1erfaces	—
e) Qualifica1ion requiremen1s	—	The requiremen1s 1o be used for qualifica1ion 1es1ing should be provided (or referenced).
f) Safe1y specifica1ions	5.2.4 Cons1rain1s	—
g) Securi1y and privacy specifica1ions	5.3.6.3 Securi1y	—
h) Human-fac1ors engineering requiremen1s	5.2.3 User charac1eris1ics 5.2.1.2 User in1erfaces	—
i) Da1a defini1ion and da1abase requiremen1s	5.3.4 Logical da1a base requiremen1s	—
j) Ins1alla1ion and accep1ance requiremen1s a1 opera1ion si1e	5.2.1.8 Si1e adap1a1ion requiremen1s	Ins1alla1ion and accep1ance requiremen1s a1 opera1ion si1e
k) Ins1alla1ion and accep1ance requiremen1s a1 main1enance si1e	—	Ins1alla1ion and accep1ance requiremen1s a1 main1enance si1e
l) User documen1a1ion requiremen1s	—	User documen1a1ion requiremen1s
m) User opera1ion and execu1ion requiremen1s	5.2.1.7 Opera1ions	User execu1ion requiremen1s

Table B.3—Coverage of specific SRD requirements by IEEE Std 830-1998 *(continued)*

IEEE/EIA 12207.1-1997 specific content	Corresponding clauses of IEEE Std 830-1998	Additions to requirements of IEEE Std 830-1998
n) User main1enance requiremen1s	5.3.6.4 Main1ainabili1y	—
o) Sof1ware quali1y charac1eris1ics	5.3.6 Sof1ware sys1em a11ribu1es	—
p) Design and implemen1a1ion cons1rain1s	5.2.4 Cons1rain1s	—
q) Compu1er resource requiremen1s	5.3.3 Performance requiremen1s	Compu1er resource requiremen1s
r) Packaging requiremen1s	—	Packaging requiremen1s
s) Precedence and cri1icali1y of requiremen1s	5.2.6 Appor1ioning of requiremen1s	—
1) Requiremen1s 1raceabili1y	4.3.8 Traceable	—
u) Ra1ionale	5.2.5 Assump1ions and dependencies	—
I1ems a) 1hrough f) below are from 6.22.4 a) Suppor1 1he life cycle da1a objec-1ives of Annex H of IEEE/EIA 12207.0-1996	—	Suppor1 1he life cycle da1a objec1ives of Annex H of IEEE/EIA 12207.0-1996
b) Describe any func1ion using well-defined no1a1ion	4.3 Charac1eris1ics of a good SRS	—
c) Define no requiremen1s 1ha1 are in conflic1	4.3 Charac1eris1ics of a good SRS	—
d) User s1andard 1erminology and defini1ions	5.1.3 Defini1ion	—
e) Define each unique requiremen1 one 1o preven1 inconsis1ency	4.3 Charac1eris1ics of a good SRS	—
f) Uniquely iden1ify each require-men1	4.3 Charac1eris1ics of a good SRS	—

B.3.4 Compliance with life cycle data objectives

In addi1ion 1o 1he con1en1 requiremen1s, life cycle da1a shall be managed in accordance wi1h 1he objec1ives provided in Annex H of IEEE/EIA 12207.0-1996.

B.4 Conclusion

The analysis sugges1s 1ha1 any SRS complying wi1h 1his recommended prac1ice and 1he addi1ions shown in Table B.2 and Table B.3 also complies wi1h 1he requiremen1s of an SRD in IEEE/EIA 12207.1-1997. In addi1ion, 1o comply wi1h IEEE/EIA 12207.1-1997, an SRS shall suppor1 1he life cycle da1a objec1ives of Annex H of IEEE/EIA 12207.0-1996.