CONTENIDO RECOMENDADO PARA UN SRS SEGÚN LA NORMA IEEE 830-1998

1. **Introducción**

Esta sección establece el propósito y el contexto del documento. Incluye:

- 1.1 Propósito: Describe el objetivo del documento, qué software se está especificando y para quién está destinado (desarrolladores, clientes, usuarios, etc.).

- 1.2 Alcance: Define los límites del sistema, incluyendo qué hará el software, sus principales funcionalidades y lo que no está incluido.

- 1.3 Definiciones, acrónimos y abreviaturas: Proporciona un glosario con términos técnicos, acrónimos y abreviaturas usados en el documento para garantizar claridad.

- 1.4 Referencias: Lista documentos, estándares o recursos relacionados que sirvan de base o apoyo (por ejemplo, IEEE 830-1998, normativas legales, etc.).

- 1.5 Visión general: Describe la estructura del resto del documento, explicando cómo está organizado y qué contiene cada sección.

2. **Descripción General**

Esta sección ofrece una visión general del sistema sin entrar en detalles específicos de los requisitos. Incluye:

- 2.1 Perspectiva del producto: Describe cómo encaja el software dentro de un sistema más grande (si aplica) y sus relaciones con otros sistemas o productos.

- 2.2 Funciones del producto: Resume las principales funciones que el sistema debe realizar, desde la perspectiva del usuario.

- 2.3 Características de los usuarios: Identifica los tipos de usuarios del sistema, su nivel de experiencia, formación y actividades principales.

- 2.4 Restricciones Detalla limitaciones que afectan el diseño o desarrollo, como hardware, sistemas operativos, lenguajes de programación, normativas o metodologías específicas.

- 2.5 Suposiciones y dependencias: Enumera factores asumidos (por ejemplo, disponibilidad de un sistema operativo) que, de cambiar, podrían afectar los requisitos.

- 2.6 Requerimientos futuros (opcional): Describe posibles requisitos que podrían surgir en el futuro, pero no son parte del alcance actual.

3. **Requisitos Específicos**

Esta sección detalla los requisitos del sistema de forma precisa y verificable. Puede organizarse de diferentes maneras (por ejemplo, por tipo de usuario, por objetos, por funciones, etc.). Incluye:

- 3.1 Requerimientos de interfaces externas:

  - Interfaces de usuario: Describe las interfaces gráficas, comandos o interacciones con el usuario.

  - Interfaces de hardware: Detalla los requisitos para interactuar con hardware específico.

  - Interfaces de software: Especifica conexiones con otros sistemas o aplicaciones (por ejemplo, APIs, bases de datos).

  - Interfaces de comunicación: Define protocolos de red o comunicación (por ejemplo, TCP/IP, FTP).

- 3.2 Requerimientos funcionales: Describe las funciones específicas que el sistema debe realizar, expresadas en términos de "el sistema debe...". Por ejemplo:

  - Autenticación de usuarios.

  - Gestión de datos (crear, leer, actualizar, eliminar).

  - Generación de reportes.

  - Cada requisito debe ser claro, no ambiguo, verificable y rastreable.

- 3.3 Requerimientos no funcionales:

  - Rendimiento: Especificaciones sobre velocidad, capacidad o tiempo de respuesta.

  - Seguridad: Requisitos para proteger datos o accesos (por ejemplo, encriptación, autenticación).

  - Portabilidad: Capacidad del software para operar en diferentes plataformas.

  - Confiabilidad: Nivel de estabilidad o tolerancia a fallos.

  - Usabilidad: Facilidad de uso para los usuarios.

  - Mantenibilidad: Facilidad para realizar modificaciones o actualizaciones.

- 3.4 Restricciones de diseño: Limitaciones específicas en la implementación, como el uso de un lenguaje de programación o frameworks concretos.

- 3.5 Atributos del sistema: Otras características, como disponibilidad, escalabilidad o compatibilidad.

- 3.6 Otros requisitos: Cualquier requisito que no encaje en las categorías anteriores, como requisitos legales, culturales o políticos.

4. **Apéndices**

Incluye información adicional que no forma parte del núcleo del SRS, pero que es relevante, como:

- Diagramas (casos de uso, diagramas de flujo, etc.).

- Modelos de datos.

- Información suplementaria sobre el sistema o el proceso de desarrollo.

5. **Índice**

Un índice o tabla de contenido para facilitar la navegación, especialmente en documentos largos. También puede incluir una tabla de referencias cruzadas para rastrear requisitos.

Características de un buen SRS según IEEE 830

Un SRS debe cumplir con las siguientes características:

- Correcto: Refleja fielmente las necesidades del cliente y las especificaciones del producto.

- No ambiguo: Usa un lenguaje claro y preciso para evitar interpretaciones múltiples.

- Completo: Incluye todos los requisitos relevantes solicitados por el cliente.

- Consistente: Evita contradicciones entre requisitos o definiciones.

- Rastreable: Cada requisito tiene un identificador único para seguirlo a lo largo del ciclo de vida del proyecto.

- Modificable: Está estructurado para facilitar cambios sin introducir inconsistencias.

- Verificable: Los requisitos son medibles o comprobables (por ejemplo, "el sistema debe procesar 100 transacciones por segundo" es verificable).

- Clasificado por importancia/estabilidad: Prioriza requisitos según su relevancia o probabilidad de cambio.

Notas adicionales

- Formato: El estándar no exige seguir estrictamente su estructura, pero toda la información relevante debe estar presente. Las secciones no aplicables pueden marcarse como "No Aplicable" (NA) con una justificación.

- Herramientas: Aunque se puede usar un procesador de texto como Microsoft Word, herramientas de gestión de requisitos como Visure Solutions facilitan la trazabilidad y gestión de cambios.[](https://visuresolutions.com/software-requirement-specification-srs-tips-template/)

- Importancia: El SRS es un documento clave para alinear expectativas entre clientes y desarrolladores, reducir retrabajos y servir como base para la validación y verificación del software.