

# Documentación e informe de requisitos

## **Breve descripción:**

A través de este recurso educativo se exponen los conceptos para la apropiación de las fases iniciales en las soluciones de transformación digital, la documentación, la ingeniería de requisitos, los registros y los acuerdos, además se abordan los paradigmas de los modelos de ciclo de vida del software.

---

**Julio 2023**

## Tabla de contenido

Introducción.....	1
1. Aplicaciones web .....	3
1.1. Componentes.....	3
1.2. Plataformas.....	5
1.3. Bases de datos .....	7
1.4. Backend de autenticación.....	10
1.5. Redundancia .....	12
2. Ciclo de vida del software .....	14
2.1. Fases.....	15
2.2. Paradigmas de los modelos de ciclo de vida del software .....	16
3. Fase de definición de requisitos .....	27
4. Requisitos.....	29
4.1. Importancia de los requisitos .....	29
4.2. Clasificación .....	31
5. Ingeniería de requisitos .....	33
6. Informe de requisitos .....	37
6.1. Elementos del documento.....	37
6.2. IREB (International Requirements Engineering Board).....	39

6.3. Estándar IEEE .....	41
6.4. SCRUM (historias de usuario) .....	43
6.5. Criterios de aceptación .....	46
7. Buenas prácticas de documentación.....	49
7.1. Normas APA .....	49
7.2. Buenas prácticas de redacción de requisitos.....	51
Síntesis .....	54
Material complementario.....	55
Glosario.....	56
Referencias bibliográficas .....	57
Créditos.....	58

## Introducción

En este componente se abordarán los conceptos y los fundamentos para realizar la documentación e informe de requisitos con base en las buenas prácticas, además se abordarán los conceptos acerca del ciclo de vida del software y las aplicaciones web.

Vea un video que contextualiza al respecto:

### Video 1. Documentación e informe de requisitos



#### [Enlace de reproducción del video](#)

#### **Síntesis del video: Documentación e informe de requisitos**

En la actualidad se hace necesario que las empresas cuenten con procesos para la ingeniería de requisitos, garanticen el registro, la documentación y los informes que

los sustenten y así poder obtener una mejor comprensión de las necesidades del cliente, el contexto y el problema.

Todo esto da como resultado, la generación de una debida documentación de los requisitos, la arquitectura, los aspectos técnicos y lo necesario para el usuario final.

En el componente formativo se abordan las fases para la definición de requisitos, su importancia, la clasificación, las técnicas para realizar el informe y las buenas prácticas de documentación.

Además, se profundizará en los paradigmas de los modelos de ciclo de vida del software, las aplicaciones web y sus componentes, los cuales son necesarios para la mejora de procesos empresariales y el avance tecnológico.

## **1. Aplicaciones web**

En la historia de la computación la web sin duda fue el acontecimiento que marcó el hito más importante para la popularidad y el aumento del uso de la internet, que ya venía naciendo entre los años 60 y 70 del siglo pasado, pero no había tenido relevancia más allá de los proyectos científicos, las grandes corporaciones, los académicos y los militares; sin embargo, no fue hasta década y media después que con la creación de la World Wide Web publicada en 1990 por el inglés Tim Berners-Lee permitió su popularidad y su funcionalidad, pues aumentó el uso a través de los navegadores (browser).

Las aplicaciones web son, entonces, los servicios que se desarrollan en la web, implicando el despliegue de un conjunto de software, tecnologías y plataformas que soportan su funcionamiento, comunicación y transmisión.

Los diversos servicios web intercambian datos entre sí para ofrecer servicios más sofisticados y específicos. Esto sucede bajo una arquitectura básica, donde los proveedores brindan servicios remotos que podrían ser solicitados por usuarios a través de llamados que llegan al cliente, todo por medio de la web.

### **1.1. Componentes**

Existen diversos elementos que son importantes de entender, si bien no son nuevos, son la fundamentación tecnológica que soporta el funcionamiento de los aplicativos basados en la web y su funcionamiento en la internet.

A continuación se mencionan los aspectos más relevantes que todo profesional de la digitalización y la transformación tecnológica debe conocer y estudiar, donde cada

elemento implica una arquitectura y una capa en el funcionamiento de la internet a través de la web.

- **WWW.** Son las iniciales de Word Wide Web, por lo general se colocan estas tres letras para indicar al navegador que se trata de un aplicativo web, que se despliega mediante el protocolo HTTP.
- **HTTP.** Son las siglas de Hypertext Transfer Protocol que significa protocolo de comunicación para la transferencia de hipertexto, usado en las aplicaciones web.
- **IP.** Son las siglas de Internet Protocol que significa Protocolo de internet, es un protocolo de comunicación, de la capa de red, que se responsabiliza de las direcciones IP y determina las rutas de salida y llegada de los paquetes de datos que van y vienen. Cada página web cuenta con una dirección IP única, desde la línea de comando puede conocerla mediante el comando ping a la página web.
- **DNS.** Es el sistema traduce los nombres de los dominios, por ejemplo, `www.sena.edu.co` a direcciones IP aptas para la lectura por parte de las máquinas (ejemplo: 186.113.6.24). Desde la línea de comandos puede usar la herramienta `nslookup` seguido de la dirección web.
- **URL.** Son las siglas de Uniform Resource Locator que significa Localizador de recursos uniforme, se trata de la dirección única que se asigna a cada recurso de una web. Esta URL la usan los navegadores y los usuarios para llegar a una web o a un lugar específico de esta.
- **HTML.** Son las siglas de HyperText Markup Language que significa lenguaje de marcado de hipertexto, es el lenguaje 'nativo' que se utiliza en

los sitios web y lo traduce el explorador o navegador para mostrar los contenidos. Actualmente, la web no solo usa HTML, se complementa con estilos (CSS), JavaScript, XML y otros recursos de programación.

- **XML.** Son las siglas de Extensible Markup Language que significa lenguaje de marcado extensible, se emplea para enviar o recibir información de otros sistemas y para integrar las plataformas de información. Es un lenguaje que comparte datos de un sistema a otro.
- **Navegador web o browser.** Es la aplicación tipo cliente que permite el acceso y la carga de los sitios web, interpreta la información enviada por servidores y permite la visualización al usuario final. Actualmente los navegadores más populares en su orden son: Google Chrome, Apple Safari, Microsoft Edge, Mozilla Firefox y Opera entre otros.

## 1.2. Plataformas

Son espacios en la internet que ejecutan diversas aplicaciones, servicios o programas, estos permiten la virtualización de servicios especializados, ahorrando tiempo de desarrollo y optimizando los recursos, pues las organizaciones solo deben concentrarse en su acción misional sin preocupaciones por las herramientas digitales o contratar personas especializadas, e incluso contar con mayor flexibilidad al momento de cambiar de proveedor si no satisface los requerimientos o las necesidades.

Las plataformas web podrían clasificarse según el uso requerido de la siguiente manera:



- **Plataformas sociales.** Quizás las más usadas masivamente, se relacionan directamente con las redes sociales, y su funcionalidad es guardar y gestionar la información referente a las interacciones.
- **Plataformas educativas.** Se trata de plataformas LMS (Learning Management System - Sistema de gestión del aprendizaje) y otras, que se concentran en gestionar contenido educativo y usuarios con sus roles correspondientes, en estas se crean cursos, materiales de formación, se gestionan exámenes, certificados y todo lo relacionado tanto con lo académico como con lo administrativo.
- **Comercio electrónico (e-commerce).** Son plataformas dedicadas a las ventas, a través de estas las personas pueden averiguar, cotizar y comprar desde internet, permitiendo que la masificación de transacciones y compras online venga en aumento en Latinoamérica y especialmente en Colombia.
- **Computación en la nube (Cloud computing).** Este tipo de plataformas son ecosistemas digitales donde conviven con otras plataformas de negocio. Ofrecen múltiples servicios TI con arquitecturas basadas en la nube a través de servidores remotos, cuyos propietarios, por lo general, son los proveedores de estos servicios.

Para cada necesidad existe una o muchas plataformas web que permiten desarrollar fácilmente lo que se requiera, ya sea para el desarrollo web o de aplicaciones, marketing, diseño e imagen, diversión y juegos, apuestas, gestión de proyectos, etc.

Todas las plataformas han tenido un crecimiento importante, las opciones en el mercado son múltiples y con muy buenas prestaciones, antes de iniciar proyectos web enmarcados en una actividad específica, no está de más realizar investigaciones previas a los proyectos para definir las mejores opciones para las organizaciones.

Generalmente, lo que se proyecta o plantea desarrollar en las empresas ya existe en una plataforma que lo hace de manera rápida y especializada, además suele ser más económico contratar una plataforma web especializada que crear una propia.

### **1.3. Bases de datos**

*Son colecciones digitales y la manera más adecuada para almacenar datos en un sistema de información, debido a sus diversas características de seguridad, capacidad de recuperación ante fallos, gestión centralizada, estandarización del lenguaje de consulta y funcionalidades avanzadas; las bases de datos son un elemento fundamental en el entorno informático, actualmente tienen una aplicación en todos los ámbitos y los campos.*

**- Pulido, Escobar & Núñez, 2019.**

Los principios y las fundamentaciones de las bases de datos se usan desde hace décadas, estas han venido evolucionando y permitiendo el avance de lo que es la informática actual.

Aún se mantienen las bases de datos relacionales con características y métodos propios con tablas normalizadas; pero ya han aparecido otros tipos de arreglos y

almacenamientos tales como bases de datos NoSQL, dimensionales, entre otras, que permiten el desarrollo de nuevas maneras de gestionar la información, teniendo en cuenta el big data, los modelos analíticos y el gran volumen de datos y la diversidad que actualmente se genera.

Las arquitecturas de uso predominantes se encuentran implementadas bajo modelos de computación en la nube, por lo que la infraestructura física y lógica de los servidores pasa a ser adquirida a través de un tercero o proveedor de servicios tecnológicos.

Los sistemas de gestión de bases de datos predominantes en el mercado son:

- **SQL Server.** Del proveedor Microsoft, gestiona datos relacionales basados en el lenguaje Transact-SQL, de gran capacidad para gestionar muchos usuarios y grandes cantidades de datos de manera simultánea, con buen soporte técnico y, el costo de las licencias es medianamente alto.
- **Oracle.** Del proveedor Oracle Corporation, este tipo de base de datos cuenta con gran capacidad para gestionar muchos usuarios y grandes cantidades de datos de manera simultánea. Es altamente confiable, con buen soporte técnico, el costo de la licencia es relativamente alto, pero es un sistema muy estable y utilizado por grandes empresas.
- **MySQL.** Es un tipo de base de datos relacional, multihilo y multiusuario, es la base de datos más usada a nivel mundial, tiene diversas licencias (tanto comercial como no paga), la más popular es la de código abierto y uso libre; no obstante existen múltiples marcas que ofrecen MySQL promoviendo la infraestructura en la nube como Google, AWS, Oracle, entre otras.

- **PostgreSQL.** Es un tipo de base de datos relacional orientada a objetos, con licencia de código abierto y uso libre. Presenta estabilidad y capacidad de gestionar grandes volúmenes de datos.
- **Cassandra.** Es un tipo de base de datos NoSQL, tiene un nivel muy elevado de compatibilidad para el clúster que abarca múltiples centros de datos con la replicación asíncrona sin maestro, lo que permite operaciones de baja latencia para todos los clientes, impulsando el uso de big data en las empresas, además permite procesos en tiempo real con cientos de fuentes de datos diferentes.
- **Mongo DB.** Es un tipo de base de datos NoSQL, de código abierto, que en lugar de guardar los datos en tablas se orienta a documentos en estructuras de datos BSON (similar a JSON), estos esquemas hacen que la integración con aplicaciones sea más rápida. Ideal para soluciones de big data.

En términos generales, todas las marcas (en especial las pagas), prometen grandes ventajas para usar sus motores de bases de datos, entre las características más importantes a la hora de seleccionar un sistema de gestión de datos son: estabilidad, prueba a fallos, capacidad de redundancia, infraestructura con alta disponibilidad, consolas administrativas, conectividad con otros sistemas de información, versatilidad en las plataformas, precios, entre otras características.

## 1.4. Backend de autenticación

Este hace referencia al principio del funcionamiento de la web, donde un usuario solicita peticiones o servicios desde un navegador en un equipo de cómputo (celular, desktop u otro dispositivo) y estas solicitudes viajan por la red, para que un servidor envíe la respuesta a dicha petición; en la práctica esta dinámica de solicitudes y respuestas de peticiones se hacen de forma veloz, durante todo el tiempo que se usa la internet.

Es aquí donde un usuario denominado “cliente” y un servicio provisto por un “servidor” intercambian información mediante “peticiones” y “respuestas” entre ambos, dando cuenta del funcionamiento web. Por el lado del cliente, quien está representado por las máquinas locales de los usuarios a través de navegadores, protocolos de comunicación y códigos de lenguaje fáciles de traducir por los sistemas del cliente (HTML, CSS, JSON, etc.), y por el lado del servidor que consta de elementos como la programación, la gestión de datos, los archivos y todo el funcionamiento interno de los servicios y peticiones.

En este sentido, para el desarrollo web existen diversos profesionales especializados, aquellos que se concentran en crear para el Backend (del lado del servidor) y quienes se especializan en Frontend (del lado del cliente). Algunos profesionales tienen ambas habilidades técnicas y se denominan Full Stack.

En el siguiente recurso interactivo se ilustran los elementos que conforman el Backend y el Frontend para mejor comprensión:

### a. Frontend

- **Programación en el Front.** Si bien esta parte podría parecer ambigua y quizás confusa para diferenciar entre el Back y el Front, es necesario mencionar que existen algunos lenguajes del lado del cliente, es decir, lenguajes que se desarrollan principalmente desde los navegadores con lógica de cliente, por lo general, hacen peticiones y conexiones con elementos del Back. Entre estos lenguajes se tienen HTML, JavaScript y CSS.
- **Interfaz web.** En esta se presentan y establecen todos los elementos visuales y ambientes gráficos de los recursos web.
- **Validaciones.** En este aspecto el Frontend se responsabiliza de cargar validaciones al momento de ingresar a zonas privadas que requieren autenticación de usuarios. Envía contraseñas y usuarios que viajan encriptadas y aceptan o no las validaciones.
- **Consumo de APIs.** Una APIs “interfaz de programación de aplicaciones” es un mecanismo para que dos software se integren, puedan comunicarse y compartir recursos entre sí, por ejemplo, si se desarrolla una aplicación que tenga que ver con mapas se puede usar la API de Google Maps u otras disponibles para que usen los recursos disponibles de otros sistemas.

### b. Backend

- **Programación de APIs.** Programar las funcionalidades de integración para usar los recursos por otra aplicación, ya sea propia o de terceros requiere definir y programar la estructura para que pueda ser usada.

- **Programación en Backend.** Se traduce en el uso de lenguajes que se ejecutan en los servidores; lo que quiere decir que todo el trabajo del procesamiento y cálculos de datos e información se ejecutan en el servidor en el cual está alojado el servicio.
- **Bases de datos.** Desde el cliente a través de las interfaces del Frontend se realiza una petición, por ejemplo, una búsqueda; esta petición se realiza al servicio de base de datos (del Backend), el proceso de búsqueda y carga de procesamiento se ejecuta en el servidor y este envía al cliente los resultados o mensajes de error en caso de presentarse.

## 1.5. Redundancia

En el marco de la Ingeniería, la Informática y la web, la redundancia se consolida como un respaldo para garantizar las mínimas caídas posibles de los servicios, de esta manera si una infraestructura o servicio falla entraría a operar el redundante; otra razón para implementar la redundancia es la conservación de la información, evitar pérdidas o incluso prevenciones contra ataques cibernéticos.

Los tipos de redundancia más comunes son:

- **Redundancia de servidores.** Uno de los valores agregados que ofrecen los proveedores de computing cloud es la de garantizar servicios redundantes, de esta manera los datos (bases de datos) y procesos (programas en línea y archivos) que estén en un proveedor falla entrarían

de manera automática al respaldo, a suplir mientras el servidor principal vuelve a subir los servicios.

- **Redundancia de red.** Para algunas empresas es de vital importancia mantener la conexión, por lo que algunas organizaciones deciden tener redundancia de proveedores de internet, ya sea para mantener velocidades más eficientes, separar servicios o simplemente como respaldo en caso de perder conexión con la salida principal.
- **Redundancia de poder.** Algunas entidades desean asegurar que sus servicios se mantengan en línea, para este objetivo invierten en la implementación de respaldos de alimentación eléctrica, evitando cortes en caso de que la red principal falle; estos respaldos tienen múltiples arquitecturas y es un tema que debe consultarse con expertos en ingeniería eléctrica, sistemas de protección y redundancia de energía.



## 2. Ciclo de vida del software

También conocido como (SDLC o Systems Development Life Cycle), es el proceso que se sigue para construir y hacer evolucionar un determinado software. El ciclo de vida permite iniciar una serie de fases mediante las cuales se procede a la validación y al desarrollo del software garantizando que se cumplan los requisitos para la aplicación y verificación de los procedimientos de desarrollo; para ello, se utilizan métodos del ciclo del software, que indican distintos pasos a seguir para el desarrollo de un producto.

Si bien existen diferentes ciclos de desarrollo de software, la normativa ISO/IEC/IEEE 12207:2017 establece que: [Es] un marco común para los procesos del ciclo de vida de los programas informáticos, con una terminología bien definida, a la que pueda remitirse la industria del software. Contiene procesos, actividades y tareas aplicables durante la adquisición, el suministro, el desarrollo, el funcionamiento, el mantenimiento o la eliminación de sistemas, productos y servicios informáticos. Estos procesos del ciclo de vida se llevan a cabo mediante la participación de los interesados, con el objetivo final de lograr la satisfacción del cliente (s.p.).

A continuación, se indican cuáles son los elementos que integran un ciclo de vida:

- **Fases.** Una fase es un conjunto de actividades relacionadas con un objetivo en el desarrollo del proyecto. Se construye agrupando tareas (actividades elementales) que pueden compartir un tramo determinado del tiempo de vida de un proyecto. La agrupación temporal de tareas impone requisitos temporales correspondientes a la asignación de recursos (humanos, financieros o materiales).

- **Entregables.** Son los productos intermedios que generan las fases. Pueden ser materiales o inmateriales (documentos, software). Los entregables permiten evaluar la marcha del proyecto mediante comprobaciones de su adecuación o no a los requisitos funcionales y de condiciones de realización previamente establecido.

## 2.1. Fases

- a. **Fase planificación.** En esta primera fase se realiza el planteamiento del problema, se definen alcances y objetivos del software.  
**Objetivos:** estudio de viabilidad, realizar planificación detallada.
- b. **Fase análisis (definición de requisitos).** Esta fase busca definir los requisitos que son los que dirigirán el desarrollo del proyecto de software.  
**Objetivos:** conocer los requisitos, asegurar que los requisitos son alcanzables, formalizar acuerdo con el cliente.
- c. **Fase diseño.** En esta fase se estudian posibles opciones de implementación para el software que hay que construir, estructura general del mismo.  
**Objetivos:** identificar soluciones tecnológicas, asignar recursos materiales, proponer identificar y seleccionar, establecer métodos de validación, ajustar especificaciones.
- d. **Fase pruebas.** Esta fase busca detectar fallos cometidos en las etapas anteriores para corregirlos.

**Objetivos:** realizar los ajustes necesarios para corregir posibles errores o inconsistencias.

- e. **Fase mantenimiento.** En esta fase se realizan tres puntos referenciados: mantenimiento correctivo, mantenimiento adaptativo y mantenimiento perfectivo.

**Objetivos:** operación asegurar que el uso del proyecto es el que se pretendía, mantenimiento.

## 2.2. Paradigmas de los modelos de ciclo de vida del software

Con la finalidad de proporcionar una metodología común entre el cliente y la empresa de software, se utilizan los modelos de ciclos de vida o paradigmas de desarrollo de software para plasmar las etapas y la documentación necesaria, de manera que cada fase se valide antes de continuar con la siguiente.

Un modelo de ciclo de vida de software es una vista de las actividades que ocurren durante el desarrollo de software e intenta determinar el orden de las etapas involucradas y los criterios de transición asociados entre estas.

Según la norma 1074 IEEE se define al ciclo de vida del software como: “una aproximación lógica a la adquisición, el suministro, el desarrollo, la explotación y el mantenimiento del software”.

La ISO/IEC 12207 Information Technology / Software Life Cycle Processes señala que es:

Un marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de

software, abarcando la vida del sistema desde la definición de los requisitos hasta la finalización de su uso (2008).

Existen modelos preestablecidos con los cuales se puede elaborar un proyecto; a continuación, se mencionan los diferentes paradigmas de modelos de ciclo de vida para desarrollar software.

- **Paradigma tradicional.** Los paradigmas tradicionales se identifican, fundamentalmente, por ser lineales, es decir se trata de completar cada proceso de principio a fin hasta que quede listo para avanzar a la segunda fase del ciclo del software.  
Si bien es verdad que las metodologías actuales están basadas en lo que fueron los paradigmas tradicionales, hoy en día se ha evolucionado, sin embargo, los paradigmas tradicionales se mantienen.
- **Desventaja.** Pérdida de tiempo si se encuentran errores en una fase avanzada porque al devolverse se debe pasar nuevamente por todas las fases y reestructurar de acuerdo con las modificaciones.
- **Paradigma orientado a objetos.** Las etapas de desarrollo de software en el paradigma orientado a objetos, se conforma principalmente por la creación de clases, análisis de requisitos y el diseño. Con este paradigma se pretende que el código fuente sea reutilizable para otros proyectos.
- **Paradigma de desarrollo ágil.** El objetivo de este paradigma es el desarrollo de proyectos en poco tiempo, se simplifican procesos tediosos, se agilizan las fases del desarrollo y las interacciones se hacen en corto tiempo.

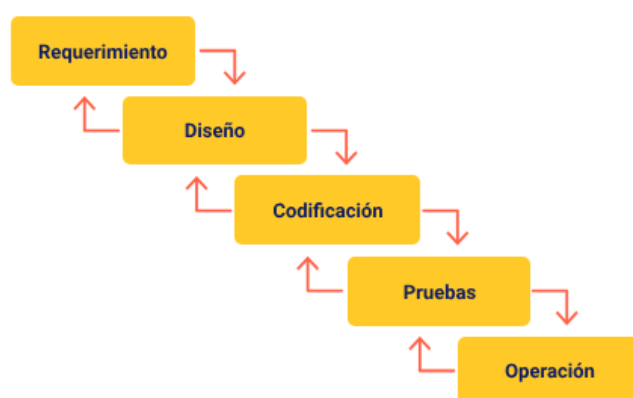
Una de las principales diferencias con los paradigmas anteriores es que el cliente se ve involucrado en el proyecto durante el desarrollo de este, así, el cliente sugiere mejoras, propone ideas y se mantiene al tanto del desarrollo del producto, a diferencia del paradigma tradicional y el orientado objeto donde el cliente únicamente está al principio.

A continuación, se revisan los modelos del paradigma tradicional más utilizados.

**Modelo en cascada.** Uno de los primeros modelos de ciclo de vida del desarrollo fue establecido por W. Royce en 1970 y es conocido como el “modelo de cascada” (waterfall model).

En su concepción básica, cada una de las actividades genera, como salidas, productos y modelos que son utilizados como entradas para el proceso subsiguiente; esto supone que una actividad debe terminarse (por lo menos, en algún grado) para empezar la siguiente.

**Figura 1.** Pasos en la algoritmia básica

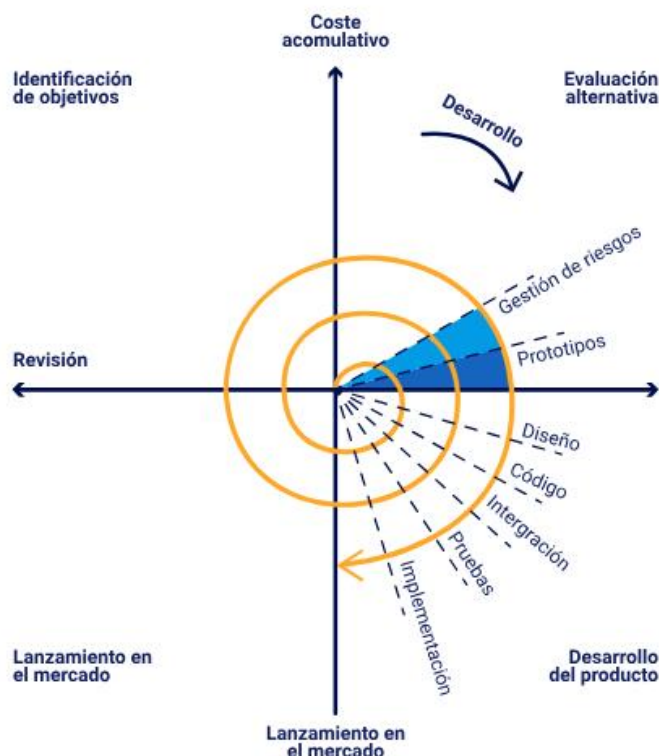


**Modelo espiral.** Fue diseñado por Boehm en el año 1988 y se basa en una serie de ciclos repetitivos para ir ganando madurez en el producto final. El espiral se repite

las veces que sea necesario hasta que el cliente o el usuario obtenga la satisfacción de sus necesidades.

En este modelo hay 4 actividades que envuelven a las etapas: planificación, análisis de riesgo, implementación y evaluación. Una de sus principales ventajas es que los riesgos van disminuyendo conforme avanzan los ciclos o interacciones.

**Figura 2.** Modelo espiral del ciclo de vida del desarrollo



### Síntesis de la figura: Modelo espiral del ciclo de vida del desarrollo

La espiral parte del centro de un plano cartesiano y va girando conforme a las manecillas del reloj de manera creciente pasando por cada fase.

Parte del recuadro de Evaluación alternativa, pasando por el desarrollo del producto, luego, continúa en el lanzamiento al mercado, sigue por una fase revisión, se identifican objetivos, después el coste acumulativo e inicia de nuevo en la evaluación alternativa y continúa su desarrollo.

En cada fase a su vez, existen microactividades. Por ejemplo, cuando se llega al Desarrollo del producto, se deben ejecutar: procesos de diseño, código, integración, pruebas e implementación.

**Modelo interactivo o por prototipos.** Este modelo consiste en un procedimiento que permite al equipo de desarrollo diseñar y analizar una aplicación que represente el sistema que será implementado (McCracken y Jackson, 1982).

**Objetivos.** Son un medio eficaz para aclarar los requisitos de los usuarios e identificar las características de un sistema que deben cambiarse o añadirse. Mediante el prototipo se puede verificar la viabilidad del diseño de un sistema.

- a. Son un medio eficaz para aclarar los requisitos de los usuarios e identificar las características de un sistema que deben cambiarse o añadirse.
- b. Mediante el prototipo se puede verificar la viabilidad del diseño de un sistema.

Las etapas del modelo son:

- a. Colecta y refinamiento de los requerimientos y proyecto rápido.
  - Análisis.
  - Especificación del prototipo.
- b. Diseño rápido.

- c. Construcción del prototipo.
- d. Evaluación del prototipo por el cliente.
- e. Refinamiento del prototipo.
  - Diseño técnico.
  - Programación y test.
  - Operación y mantenimiento.
- f. Producto de ingeniería.

**Figura 3.** Modelo de iterativo o por prototipos



Con respecto a los modelos del ciclo de vida del paradigma ágil, estos se caracterizan por estar basados en etapas del ciclo de vida del software tradicional, pero combinándolas con algunas técnicas, al respecto se pueden revisar los siguientes:



**Modelo Scrum.** Este modelo se basa en el desarrollo incremental, es decir conforme pasen las fases y la iteración mayor será el tamaño del proyecto que se está desarrollando.

Los procesos que utiliza son:

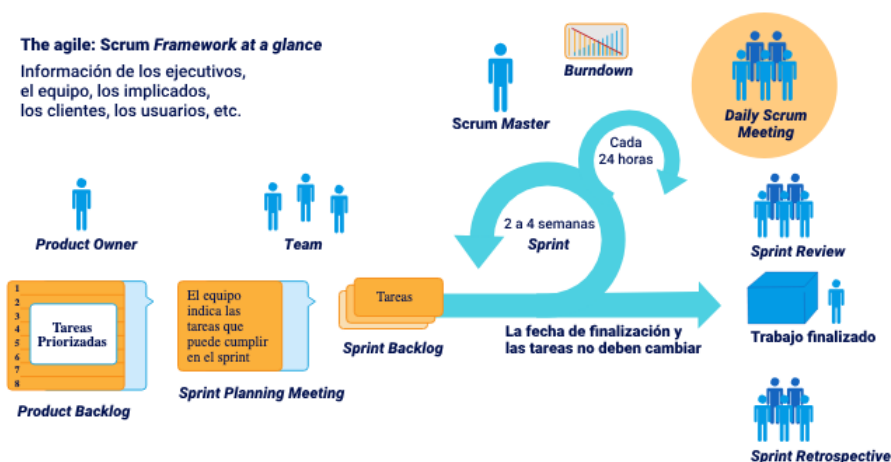
- a. Product Backlog.
- b. Sprint Backlog.
- c. Sprint Planning Meeting.
- d. Daily Scrum.
- e. Sprint Review.
- f. Sprint Retrospective.

El Scrum consiste en realizar un análisis de los requerimientos del sistema (Product Backlog), señalar cuáles serán los objetivos a corto o mediano plazo dentro de un sprint, o sea, la fase de desarrollo. Posteriormente, los desarrolladores harán lo suyo, se realizarán algunas pruebas y se retroalimentará de acuerdo con lo conseguido al terminar la última fase.

### **Ventajas**

- **Gestión regular de las expectativas del usuario:** los usuarios participan y proponen soluciones
- **Resultados anticipados:** no es necesario esperar hasta el final para ver resultados
- **Flexibilidad y adaptación:** se adapta a cualquier contexto, área o sector.
- **Gestión sistemática de riesgos:** los problemas son gestionados en el mismo momento de su aparición.

**Figura 4. Modelo ágil Scrum**



### Síntesis de la figura: Modelo ágil Scrum

Los pasos de un modelo Scrum inicia con el Product Backlog, allí se priorizan las tareas. Luego, pasa al Sprint Planning Meeting en donde el equipo indica las tareas que puede cumplirse en el sprint. Después, sigue el Sprint Backlog en el que se organizan las tareas y se establecen las fechas de finalización. El proceso de Sprint puede durar de 2 a 4 semanas. En este tiempo, se involucran la etapa Daily Scrum. una vez finalizado el trabajo se da paso al Sprint Review y al Sprint Retrospective.

**Modelo Kanban.** David J. Anderson (reconocido como el líder de pensamiento de la adopción del Lean/Kanban para el trabajo de conocimiento), formuló el método Kanban como una aproximación al proceso evolutivo e incremental y al cambio de sistemas para las organizaciones de trabajo. El método está enfocado en llevar a cabo

las tareas pendientes y los principios más importantes pueden ser divididos en cuatro principios básicos y seis prácticas.

El modelo Kanban es uno de los modelos más visuales de las metodologías ágiles; este consiste en la creación de un tablero con etiquetas, donde se seccionan cada una de las fases de su desarrollo, además se clasifican de acuerdo con los equipos de trabajo y se les asignan objetivos a corto, mediano y largo plazo.

Mediante la metodología japonesa Kanban se:

- Define el flujo de trabajo.
- Establecen las fases del ciclo de producción.
- Stop starting, start finishing.
- Tiene un control.

**Figura 5.** Modelo ágil Kanban



**Modelo XP o programación extrema.** La programación extrema o eXtreme Programming (XP) es un enfoque de la ingeniería de software formulado por Kent

Beck, autor del primer libro sobre este tema: Extreme Programming Explained: Embrace Change (1999). Esta metodología es adaptable según las necesidades y requerimientos a implementar, además, el cliente se encuentra involucrado en el proceso de desarrollo lo que hace que el producto pueda ser terminado en un menor tiempo.

- a. Tipo de desarrollo iterativo e incremental.
- b. Pruebas unitarias.
- c. Trabajo en equipo.
- d. Trabajo junto al cliente.
- e. Corrección de errores.
- f. Reestructuración del código.
- g. El código es de todos.
- h. El código simple es la clave.

Características principales de la programación extrema:

**Figura 6. Modelo XP**



### **Síntesis de la figura: Modelo XP**

La programación extrema consiste en una metodología de cuadro grandes pasos que a su vez contiene fases a tener en cuenta. Estos son:

- a. Planificación. Fase: historia del usuario
- b. Diseño. Fases: diseño simple y prototipos.
- c. Codificación. Fases: programación en parejas y pruebas unitarias integración continua.
- d. Pruebas. Fases: Lanzamiento, pruebas de aceptación

### 3. Fase de definición de requisitos

En esta primera fase del ciclo de vida del software, también llamada fase de análisis, se recopila, se examina y se formulan los requisitos del cliente, así como la verificación de las posibles restricciones que se puedan aplicar.

Por eso, la etapa de análisis en el ciclo de vida del software corresponde al proceso a través del cual se intenta descubrir qué es lo que realmente se necesita y se llega a una comprensión adecuada de los requerimientos del sistema (las características que el sistema debe poseer).

La etapa de análisis es esencial debido a que sin esta no se sabe con precisión qué es lo que se necesita y ningún proceso de desarrollo permitirá obtenerlo. El problema que se presenta es que al inicio del desarrollo el cliente no sepa exactamente lo que necesita; por tanto, se debe averiguar con ayuda de distintas técnicas.

De otra parte, la inestabilidad de los requerimientos de un sistema es inevitable, pues se estima que 25% de los requerimientos iniciales de un sistema cambian antes de que el sistema comience a utilizarse. Por ello, muchas prácticas resultan efectivas para gestionar adecuadamente los requerimientos de un sistema y, en cierto modo, controlar su evolución.

En la siguiente tabla, se describen las actividades y los artefactos que se realizan en la fase de **definición de requisitos**.

**Tabla 1.** Actividades y artefactos de la fase de definición de requisitos

Actividades	Artefactos
Definición del alcance del proyecto.	N/A
Identificación del negocio.	Modelo del negocio.
Toma de requerimientos.	Análisis y realización de casos de uso.
Estudio de procesos de negocio.	Modelo de procesos y actividades de negocio.
Calendarización del proyecto.	Cronograma del proyecto.

## 4. Requisitos

*Un requisito es una condición o capacidad que necesita el usuario para resolver un problema o conseguir un objetivo determinado (IEEE, 1990).*

Los requisitos comunican las expectativas de los consumidores de productos software; de otra parte, los requisitos pueden ser obvios o estar ocultos, conocidos o desconocidos, esperados o inesperados, desde el punto de vista del cliente.

### 4.1. Importancia de los requisitos

Los requisitos cobran importancia dentro del ciclo de vida del software, puesto que:

- a. Establecen el alcance del trabajo subsecuente, pueden definir estrategias de desarrollo, riesgos, tomar decisiones de negocio (viabilidad de negocio), de proyecto (tiempo, recursos), de sistema (arquitectura).
- b. Indican al equipo del proyecto qué requieren los usuarios (necesidades de negocio).
- c. El éxito o fracaso de un proyecto está altamente influenciado por la calidad de los requisitos y el proceso para gestionarlos durante el desarrollo de un producto.

En la siguiente figura se pueden revisar las características que los requisitos deben cumplir de acuerdo con Pfleeger (2002).

- **Necesario.** Si se tiene alguna duda acerca de la necesidad del requerimiento, se puede preguntar “¿Qué sería lo peor de no incluirlo?” Si



no se encuentra una respuesta o cualquier consecuencia, entonces es probable que no sea un requerimiento necesario.

- **Completo.** Un requerimiento está completo si no necesita ampliar detalles en su redacción, es decir, si se proporciona la información suficiente para su comprensión.
- **Consistente.** Un requerimiento es consistente si no es contradictorio con otro requerimiento.
- **Correcto.** Acuerdo entre dos partes. Contiene una sola idea.
- **Factible.** El requerimiento deberá de ser totalmente factible y dentro de presupuesto, calendario y otras restricciones, si se tiene alguna duda de su factibilidad, hay que investigar, generar pruebas de concepto para saber su complejidad y factibilidad, si aun así el requerimiento es no factible, hay que revisar la visión del sistema y replantear el requerimiento.
- **Modificable.** Los cambios en los requisitos deben hacerse de manera sistemática, y debe tenerse en cuenta su impacto en otros requisitos.
- **Priorizado.** Ategorizar el requerimiento nos ayuda a saber el grado de necesidad del mismo: esencial/crítico, deseado, opcional verificable.
- **Verificable.** Si un requerimiento no se puede comprobar, entonces, ¿cómo se sabe si se cumplió con él o no? Debe ser posible verificarlo ya sea por inspección, análisis de prueba o demostración. Cuando se escriba un requerimiento, se deberán determinar los criterios de aceptación.
- **Rasteable.** La especificación se debe organizar de tal forma que cada función del sistema se pueda rastrear hasta su conjunto de requerimientos correspondiente. Facilita las pruebas y la validación del diseño.

- **Claro.** Un requerimiento es conciso si es fácil de leer y entender, su redacción debe ser simple y clara para quienes lo consulten en un futuro.

## 4.2. Clasificación

Los requerimientos se pueden definir de distintas maneras, la primera clasificación se encuentra relacionada con el nivel de descripción con la que cuentan estos y dentro de este tipo de clasificación se encuentran:

- **Requerimientos de usuario.** Son declaraciones, en lenguaje natural y en diagramas, de los servicios que se espera que el sistema proporcione y de las restricciones bajo las cuales debe funcionar.
- **Requerimientos de sistema.** Estos requerimientos establecen con detalle las funciones, servicios y restricciones operativas del sistema. El documento de requerimientos del sistema deberá ser preciso, y definir exactamente lo que se va a desarrollar.

En la siguiente clasificación se observa la que se da a los requerimientos del sistema, la cual se encuentra dividida con base en lo que se va a describir, las clasificaciones son:

- **Requerimientos funcionales.** Son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que este debe reaccionar a entradas particulares; o también pueden declarar explícitamente lo que el sistema no debe hacer.
- **Requerimientos no funcionales.** Son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo, sobre

el proceso de desarrollo y estándares. Dentro de estos requerimientos se encuentra todo lo referente a la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento.

En la siguiente tabla se presentan algunos ejemplos sobre requisitos funcionales y no funcionales.

**Tabla 2.** Requisitos

Funcionales	No funcionales
Se debe ingresar cédula, nombre y teléfono de cada cliente.	Las consultas deben resolverse en menos de 3 segundos.
Se requiere un listado de clientes por zona.	El lenguaje de programación debe ser Java.

**Nota.** Se puede ampliar el tema de requisitos funcionales y no funcionales en los videos que se proponen dentro del material complementario.

## 5. Ingeniería de requisitos

Las siguientes son definiciones de ingeniería de requisitos de algunos autores.

- La ingeniería de requisitos es la disciplina para desarrollar una especificación completa, consistente y no ambigua, la cual servirá como base para acuerdos comunes entre todas las partes involucradas y en dónde se describen las funciones que realizará el sistema. (Boehm, 1979).
- La ingeniería de requisitos es el proceso de estudiar las necesidades del usuario para llegar a una definición de requisitos de sistema, hardware o software. (IEEE, 1990).
- La ingeniería de requisitos puede considerarse como un proceso de descubrimiento y comunicación de las necesidades de clientes y usuarios y la gestión de los cambios de dichas necesidades. (Amador, 2000)

El término IR “ingeniería de requisitos” ha surgido para englobar los procesos de desarrollo y gestión de requisitos en el ciclo de vida del software, el primer término (ingeniería) se enfoca en las actividades de obtención, análisis, especificación y validación de los requisitos que permitirá alcanzar los objetivos del negocio y el segundo (requisitos) está centrado en la administración de los mismos y tiene como propósito central la gestión de los cambios y la trazabilidad, de esta forma la IR proporciona el mecanismo apropiado para:

- Entender lo que el cliente quiere.
- Analizar las necesidades.
- Evaluar la factibilidad.
- Negociar una solución razonable.

- Especificar la solución sin ambigüedades.
- Validar la especificación.
- Administrar los requisitos conforme éstos se transforman en un sistema operacional.

### **Etapas de la ingeniería de requisitos**

Hay cuatro (4) etapas en un proceso usual de ingeniería de requisitos y que son utilizadas para el desarrollo de un producto único, a saber: elicitación, análisis, especificación y validación de los requisitos.

Un requerimiento está completo si no necesita ampliar detalles en su redacción, es decir, si se proporciona la información suficiente para su comprensión.

**Elicitación.** Actividad involucrada en el descubrimiento de los requisitos del sistema. Aquí los analistas deben trabajar junto con el cliente para descubrir el problema que el sistema debe resolver, los diferentes servicios que el sistema debe prestar y las restricciones que se pueden presentar.

Los principales objetivos que se deben alcanzar son los siguientes:

- Conocer el dominio del problema, de forma tal que los analistas puedan entenderse con los clientes y usuarios y sean capaces de transmitir dicho conocimiento al resto del equipo.
- Descubrir necesidades reales entre clientes y usuarios, haciendo énfasis en aquellas que la mayor parte de las veces se asumen y toman por implícitas.
- Consensuar los requisitos entre los propios clientes y usuarios hasta obtener una visión común de los mismos.

**Análisis.** Sobre la base de la obtención realizada previamente, comienza esta fase la cual tiene como propósito descubrir problemas con los requisitos del sistema identificados hasta el momento, para ello se basa en los siguientes objetivos:

- Detectar conflicto en los requisitos que suelen provenir de distintas fuentes y presentar contradicciones o ambigüedades debido a su naturaleza informal.
- Profundizar en el conocimiento del dominio del problema puede facilitar el proceso de construir un producto útil para clientes y usuarios (Durán, 2000).
- En esta fase, el analista proporciona un sistema de retroalimentación que refina el entendimiento conseguido en la etapa de obtención.

**Especificación.** Aquí se documentan los requisitos acordados con el cliente, en un nivel apropiado de detalle. En la práctica, esta etapa se realiza conjuntamente con el análisis, por lo que se puede decir que la especificación es el “pasar en limpio” el análisis realizado previamente aplicando técnicas y/o estándares de documentación, como la notación UML (Lenguaje de Modelado Unificado), que es un estándar para el modelado orientado a objetos, por lo que los casos de uso y la obtención de requisitos basada en los casos de uso se utilizan cada vez más para la obtención de requisitos.

**Validación.** Por último, la validación garantiza que los requisitos, una vez analizados y resueltos los posibles conflictos, correspondan realmente a las necesidades de clientes y usuarios, para evitar que, a pesar de que el producto final sea técnicamente correcto, no sea satisfactorio. La validación puede llevar al analista a reescribir algunas especificaciones de requisitos y, en otros casos, a obtener nuevos, producto de la aparición de necesidades que hasta entonces estaban ocultas, para

volver a evaluar el análisis inicial, o para corregir y perfeccionar el conjunto de requisitos documentados.

## 6. Informe de requisitos

Los requerimientos deben ser identificados, descritos y documentados de forma estructurada y normalizada, por esto, es importante diferenciar los requisitos de otros elementos, por ejemplo, declaraciones como “el programa debe escribirse en Python” no son un requerimiento, pues es una decisión de implementación. Pero algo como “un usuario puede retirar de la biblioteca hasta 10 libros diferentes al tiempo” sí es un requerimiento porque define algo que el usuario puede hacer y el sistema debe dejarlo cumpliendo restricciones y reglas de negocio.

Después de haber aplicado todas las herramientas e instrumentos para la adquisición de requisitos, necesidades del cliente y problemas a resolver, llega el momento de elaborar documentos que representen de la manera más exacta y detallada posible los elementos que la solución digital irá a abarcar.

Antes de asumir el tema específico para la documentación de requisitos se estudiarán algunos estándares que deben aplicarse a cualquier documento e informe como los elementos generales de un documento y algunos marcos referenciales orientados a la recolección de requisitos tales como IREB, IEEE y metodologías ágiles.

### 6.1. Elementos del documento

El propósito de la documentación de requisitos es comunicar y reflejar la información obtenida por el analista de requisitos, por lo cual se documenta lo investigado, estudiado y analizado. La redacción debe ser clara y completa, con un lenguaje que no dé lugar a interpretaciones erróneas, pues está dirigido a personas que incluso no tienen que ver con la organización, si bien contienen elementos técnicos,



debe ser entendible para todos y para cualquier profesional del área técnica que se menciona. Debe incluir los métodos y proponer las soluciones.

Independiente de la metodología seleccionada todos los documentos e informes deben contener como mínimo los siguientes elementos:

- **Objetivo general.** Expresa el fin del reporte; está relacionado estrechamente con la formulación del problema.
- **Objetivos específicos.** Se refiere a los propósitos y las tareas más definidas con las cuales se llega al objetivo general, se deben enumerar y presentar el alcance de los objetivos específicos de manera ordenada para lograr el resultado del objetivo general.
- **Introducción.** Permite contextualizar y generar expectativa del informe; se puede resumir el tema en forma global sin profundizar en los detalles (¿qué?), se debe indicar la importancia del informe (¿para qué?) y la manera en cómo está abordado el problema planteado (¿cómo?).
- **Contenido.** El desarrollo del informe debe presentar todos los detalles tratados, en este apartado se encuentra el desarrollo como tal del informe y la documentación que se entregará.
- **Conclusiones.** Se resume de manera muy sintetizada todo lo expuesto en el contenido, se puede desarrollar a partir de los objetivos planteados y si se cumplieron, en el caso específico de proyectos tecnológicos se debe mencionar la utilidad para la continuación de las fases de desarrollo e implementación.
- **Recomendaciones.** En este ítem es necesario tener conocimiento técnico y contexto específico de los problemas y las situaciones del proyecto (que

está detallado en el contenido). A partir del análisis de los problemas y las alternativas de solución dar un concepto técnico que se ajuste a la solución de los problemas y cumplimiento de los requerimientos, que tenga viabilidad técnica y económica para que sea posible, y tener la claridad del cómo, las implicaciones y los costos.

## **6.2. IREB (International Requirements Engineering Board)**

Es una organización sin ánimo de lucro que provee el esquema de certificación profesional en ingeniería de requisitos (CPRE), el consejo está compuesto por representantes y líderes en IR (Requisitos Internacionales), que provienen de la academia, la investigación, la empresa y la consultoría. La página web oficial es [www.ireb.org](http://www.ireb.org)

Esta surgió en Alemania en el año 2006; sin embargo, es un comité con alcance internacional, su objetivo es destacar la importancia de los requerimientos como una disciplina completa y mostrar su valor agregado a la industria del software.

Podría denominarse que la IREB es un movimiento que procura valorizar la importancia de los requerimientos, y que los sectores económicos y la industria dispongan de este componente, de la categoría y de la valoración que merece.

La asociación estimula la estandarización y el uso de metodologías de la ingeniería de requerimientos que está enmarcado en la ingeniería del software.

En el medio actual, profesionales certificados en IREB no son comunes, por lo que podría ser una opción y una oportunidad para la certificación, a continuación se

listan los aspectos que soportan la importancia de emplear técnicas en la fase de análisis y en la descripción de requerimientos:

- Se estima que el 47 % de los fracasos en los proyectos se debe a la gestión deficiente de los requerimientos.
- Se considera que al menos el 20 % de los defectos en desarrollo de software tienen sus orígenes en los requerimientos o descripciones erróneas en los requerimientos; muchas veces las fallas no son un asunto de programación o de otra naturaleza, sino deficiencias en las fases iniciales.
- Encontrar y corregir los defectos en el software después de entregado el proyecto es mucho más costoso que hacerlo en la fase de requerimientos.

La Certificación CPRE (Certified Professional on Requirements Engineering) es una certificación gestionada y entregada por IREB dirigida a los profesionales en:

- **Análisis de negocio.** Profesionales como administradores y analistas de procesos que se enfocan en mejorar y optimizar áreas de negocios.
- **Requerimientos.** Expertos en análisis de requisitos y planteamientos de soluciones informáticas.
- **Pruebas.** Expertos en testeos de aplicaciones y soluciones tecnológicas, evalúan si el proyecto sí cumple con lo pactado o prometido, emplean técnicas y métodos establecidos según el tipo de prueba que desean ejecutar.

### 6.3. Estándar IEEE

El Instituto de Ingenieros Eléctricos y Electrónicos es la organización internacional más grande que se encarga de promover avances científicos en las áreas electrónica, eléctrica, informática, energética y demás áreas relacionadas, además es la sociedad que regula y estandariza las normas internacionales de estas disciplinas, este surgió en los Estados Unidos desde la década de los 60, lo que ha permitido que la estandarización de los requisitos de software estén basados en las Normas IEEE.

La especificación de requisitos de software (ERS) hace referencia a una descripción que debe indicar algo y esta especificación debe ser entendida por el cliente, no puede llevar elementos de implementación o de desarrollo de soluciones técnicas, y a partir de una especificación surgen otras especificaciones más detalladas de producto en la descripción documentada.

La metodología IEEE 830 presenta la siguiente estructura:

#### a. Introducción.

- **Propósito.** Se especifica para qué se realiza, cuáles son los objetivos y el cliente.
- **Ámbito del sistema.** Se puede relacionar el nombre del sistema futuro. Describe lo que hará y lo que no hará, se indican los beneficios y las metas. Se incluyen documentos relacionados (legales, ambientales, de contexto).
- **Personal involucrado.** Nombre de las personas involucradas, roles, categoría y responsabilidades.
- **Definiciones, abreviaturas y lenguajes corporativos**

- **Referencias.** Listar los documentos tomados como base o referencia usados o consultados.

- **Resumen**

#### **b. Descripción general**

- **Perspectiva del producto.** Relación del futuro sistema con otros sistemas actuales de la organización. Si el proyecto hace parte de un proyecto mayor.
- **Funciones del producto.** Resumen de las funciones principales del futuro sistema.
- **Características de los usuarios.** Describir generalidades de los usuarios del producto a desarrollar; nivel educativo, rol en el sistema y actividades a realizar dentro de la empresa.
- **Restricciones.** Se refiere a las limitaciones que el nuevo producto tendrá. Se mencionan las políticas de la empresa, las limitaciones del hardware, las interfaces con otras aplicaciones, los lenguajes de desarrollo y otros elementos.
- **Suposiciones y dependencias.** Se describen los factores que sí cambian y afectan directamente los requisitos.

#### **c. Requisitos específicos**

- **Requerimientos funcionales (RF).** Expresan el funcionamiento del sistema y sus interacciones; definen qué debe hacer el sistema. Deben ser comprensibles y en lenguaje natural. Definen los requisitos obligatorios y los requisitos deseables.

- **Requerimientos no funcionales (RNF).** Describen las características generales y las restricciones al sistema (como entornos gráficos, velocidad de procesamiento), estos también deben listarse e identificarse. Por lo general los RF están asociados a uno o varios RNF.

#### **6.4. SCRUM (historias de usuario)**

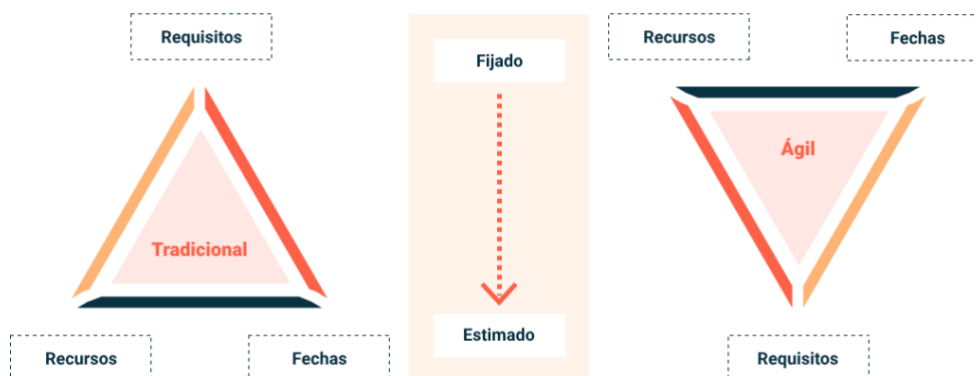
Está enmarcado en una de las metodologías ágiles, en el entorno actual es el más empleado por las empresas y su estrategia de desarrollo está basado en este modelo.

A diferencia de lo expuesto en IREB se prefiere realizar la toma de requisitos de manera tradicional a través de las reuniones con los clientes, expresar en qué consiste el requisito, sin que haya detalles minuciosos y luego estimar el requisito, priorizarlo y plasmarlo en historias de usuario.

Las metodologías tradicionales proponen fijar los requisitos con un alto nivel de detalle al inicio del proyecto y a partir de estos se hace una estimación de los costos y de la fecha de entrega. Esto podría implicar fácilmente inconvenientes, pues con frecuencia los clientes no tienen claridad sobre sus necesidades y en las metodologías tradicionales un cambio a mitad del proyecto podría resultar en problemas y trabas que implican costos, ya sea en los resultados finales o en incrementos presupuestales.

La metodología ágil como SCRUM propone un cambio de paradigma, que puede identificar a continuación:

**Figura 7.** Ejemplo del cambio de paradigma



Las metodologías ágiles como SCRUM y otras proponen partir de un presupuesto y de unas fechas de entrega, a partir de ahí se trabaja para implementar la funcionalidad más valiosa para el cliente en cada momento, trabajando de esta manera el alcance será flexible.

En el siguiente video se presenta cómo el marco de trabajo SCRUM aporta en el proceso de especificación de requisitos:

**Video 2.** SCRUM y la especificación de requisitos



[Enlace de reproducción del video](#)

### Síntesis del video: SCRUM y la especificación de requisitos

El marco de trabajo SCRUM está soportado en un proceso de construcción interactivo e incremental evolutivo en el que se identifican tres roles principales:

- El **equipo de trabajo o team**, conformado por los desarrolladores, diseñadores, personal de calidad y de infraestructura, requerido para la construcción del producto de software.
- El **scrum master** que realizan funciones parecidas a las del director del Proyecto pero más enfocados en garantizar que el equipo de trabajo tenga todas las herramientas y recursos necesarios para el desarrollo de su trabajo.
- El **dueño del producto o autor pauner** que se convierte en un representante del cliente y quien es el único encargado de la gestión de requisitos del proyecto.

SCRUM establece el concepto de sprint para referirse a una integración que contempla tiempos fijos entre dos y cuatro semanas dependiendo del equipo de trabajo. Durante este tiempo, se incluye la planeación del sprint donde se definen los requerimientos a desarrollar en ese periodo de tiempo: una fase de construcción del producto y, finalmente, un proceso de despliegue para poder hacer la respectiva demostración de lo construido al final de la iteración en reuniones de revisión.

En este marco de trabajo se redefine el concepto de requerimiento hecho y normalmente va mucho más allá de construir el código; por lo general, se incluyen procesos de validación con pruebas unitarias y pruebas de integración.



## 6.5. Criterios de aceptación

Resumen el cumplimiento del requisito o de la funcionalidad del sistema o proyecto, hacen parte de la historia de usuario en la metodología SCRUM. Es la parte en la que el cliente evalúa y se establece si el desarrollo de la parte del proyecto es o no aceptada; se propone de manera correcta estos criterios, se evitan controversias en los proyectos y se dejan claros los alcances y las funcionalidades.

Características principales de los criterios de aceptación:


- Se deben redactar con frases concretas y que concluyan las historias de los usuarios.
- Enriquece la historia y hace posible las pruebas.
- Los resultados de las pruebas solo deben tener dos estados: correcto o incorrecto.
- Se debe asegurar el entendimiento por parte de todo el equipo para que exista unanimidad en la consideración si es o no aceptada y finalizada.

Tipos de criterio de aceptación:

- **Condiciones:** cómo se reacciona frente a las opciones de comportamiento y ante situaciones que no se cumplen.
- **Funcionales:** es lo que se espera ver o lo que hace la solución.
- **No funcional:** hace referencia a la accesibilidad, la seguridad, el performance, el soporte, etc.
- **Usabilidad:** da cuenta de la facilidad de uso, la eficiencia, la navegabilidad, los errores y la experiencia de usuario.

Si bien el formato o plantillas son muy diversas, esto depende de los líderes o de la compañía, deben tener unos elementos mínimos, a continuación se presenta un ejemplo de una tarjeta o documento de historia de usuario y sus componentes:

**Figura 8.** Ejemplo de una historia de usuario



<b>Historia de usuario:</b> 32 - Catálogo de productos	
<b>Como:</b> proveedor <b>Quiero:</b> poder entrar a mis productos <b>Para poder:</b> componer un catálogo para ofrecerlos por la Web	
<b>Validación:</b> - Dar de alta productos - Comprobar que salen en la Web - Comprobar que están todos - Modificar el producto y verificar que se actualiza	<b>Valor:</b> 200
	<b>Prioridad:</b> 1
	<b>Estimación:</b> 16h

### Síntesis del video: Ejemplo de una historia de usuario

Ficha en la que se establecen datos como:

**ID del usuario y título:** 32 – catálogo de productos

**Descripción:**

- **Como:** proveedor.
- **Quiero:** poder entrar a mis productos.
- **Para poder:** componer un catálogo para ofrecerlos por la web.

**Criterios de aceptación. Validación:**

- Dar de alta productos.

- Comprobar que salen en la web.
- Comprobar que están todos.
- Modificar el producto y verificar que se actualiza.

**Ponderación.** Valor: 200

**Prioridad:** 1

**Estimación:** 16 horas

Los criterios de aceptación se expresan a modo de validación y dan luz sobre las pruebas que se deben realizar por parte del equipo de pruebas y aprobación de dicha historia.

## **7. Buenas prácticas de documentación**

El cuidado de los detalles y las formas de presentar la documentación en los proyectos no es un asunto menor; implica escribir con claridad, buen estilo estético del texto, contenidos bien redactados, respeto por los estándares de derechos de autor y por las normas ortográficas, todo independiente de la metodología o corriente de desarrollo de los proyectos tecnológicos.

El elemento fundamental y principal de las relaciones comerciales y laborales surge a partir de la confianza; un documento mal elaborado, con errores elementales y sin referencias podría ser un aspecto negativo para la generación de desconfianza.

Por lo cual se deben aplicar las buenas prácticas de documentación en todo documento que se elabore, incluyendo aquellos que están en las metodologías de desarrollo, ya sea SCRUM, IEEE u otra.

### **7.1. Normas APA**

Buscando una estandarización en las referencias bibliográficas, las Normas APA han sido el estándar más empleado para las citaciones de las fuentes complementarias o marcos referenciales de los documentos, APA son las iniciales de American Psychological Association (Asociación Americana de Psicología) <https://apastyle.apa.org> que fue la organización que creó el estándar desde hace décadas atrás en 1929 para el campo de la Psicología, pero que actualmente muchas ramas del conocimiento lo emplean como esquema documental.

A continuación se presentan algunas indicaciones para el cumplimiento de las Normas APA:

- **Citas.** Hace referencia a un tema, construcción o concepto, pueden ser también antecedentes que contribuyen al escrito propio.
- **Citas en paráfrasis.** Es usar palabras y expresiones propias de un texto o idea de otro autor, en este caso también se debe citar al autor, indicando apellidos y el año, por ejemplo: (Bonasera, 2012); y entregando el detalle completo de la fuente en las referencias bibliográficas.
- **Citas textuales cortas (40 o menos palabras).** Se deben emplear usando comillas e incluir el número de la página de referencia: “texto corto” (González, 2008, p. 34), segundo ejemplo, Según González (2008), “texto corto” (p. 34). Puede incluir la forma Según González (2008, p. 34), “texto corto”.
- **Citas textuales con más de 40 palabras.** Estas citas deben estar separadas del texto propio usando tabulador con sangría de 1,27 cm desde el margen izquierdo, no se usan comillas. Un aspecto diferenciador de las citas cortas es que el punto final va antes del paréntesis que abre la cita. En caso de que la citación tenga un segundo párrafo, la primera línea de este debe tabularse otros 1,27 cm.
- **Citas de autores corporativos.** Cuando se toman referencias de entidades y organizaciones como (SENA, ICBF, IBM, AWS, etc.), se recomienda la primera vez usar los nombres completos sin abreviar, por ejemplo: Según la Organización de las Naciones Unidas para la Educación, la Ciencia y la Cultura (UNESCO, 2012), la educación debe establecer...

- **Referencia de artículos y revistas digitales con DOI (Digital Object Identifier).** Debe tener el apellido, inicial del autor, año de publicación, título del artículo con volumen, en cursiva, páginas y dirección DOI, por ejemplo: Ortega, Y., Hernández, M. (2019). Concepción dinámica familiar: Análisis desde la percepción de un grupo de niños de cuatro años. *Cultura, Educación y Sociedad*, 10, 63-72.

Se invita a que amplíe este concepto y tenga como referencia el “Instructivo uso del estilo APA 7ª edición” del SENA:

**Nota.** Para profundizar en las instrucciones del uso de estilo APA, citas bibliográficas, referencias bibliográficas; cómo aprovechar Microsoft Word para el uso de citas y referencias, y algunos sitios web recomendados, se sugiere revisar el siguiente recurso web. [Enlace Normas APA](#)

## 7.2. Buenas prácticas de redacción de requisitos

Se recomienda aplicar algunos puntos clave a la hora generar un buen documento enfocado a la comunicación, registro de proyectos de transformación digital y de requisitos; presentar textos a los clientes o equipos de trabajo con mal manejo de redacción y poca claridad puede tener implicaciones negativas cuando el proyecto esté en fases posteriores.

Por esta razón, se presentan algunos consejos prácticos para la construcción de documentos bien elaborados:

- **Defina la metodología que seguirá.** Seleccione la línea metodológica que va a usar, ocasionalmente lo define el cliente o la empresa que dirige los

proyectos, pero en caso de no hacerlo use las técnicas como la IEEE, SCRUM, entre otras.

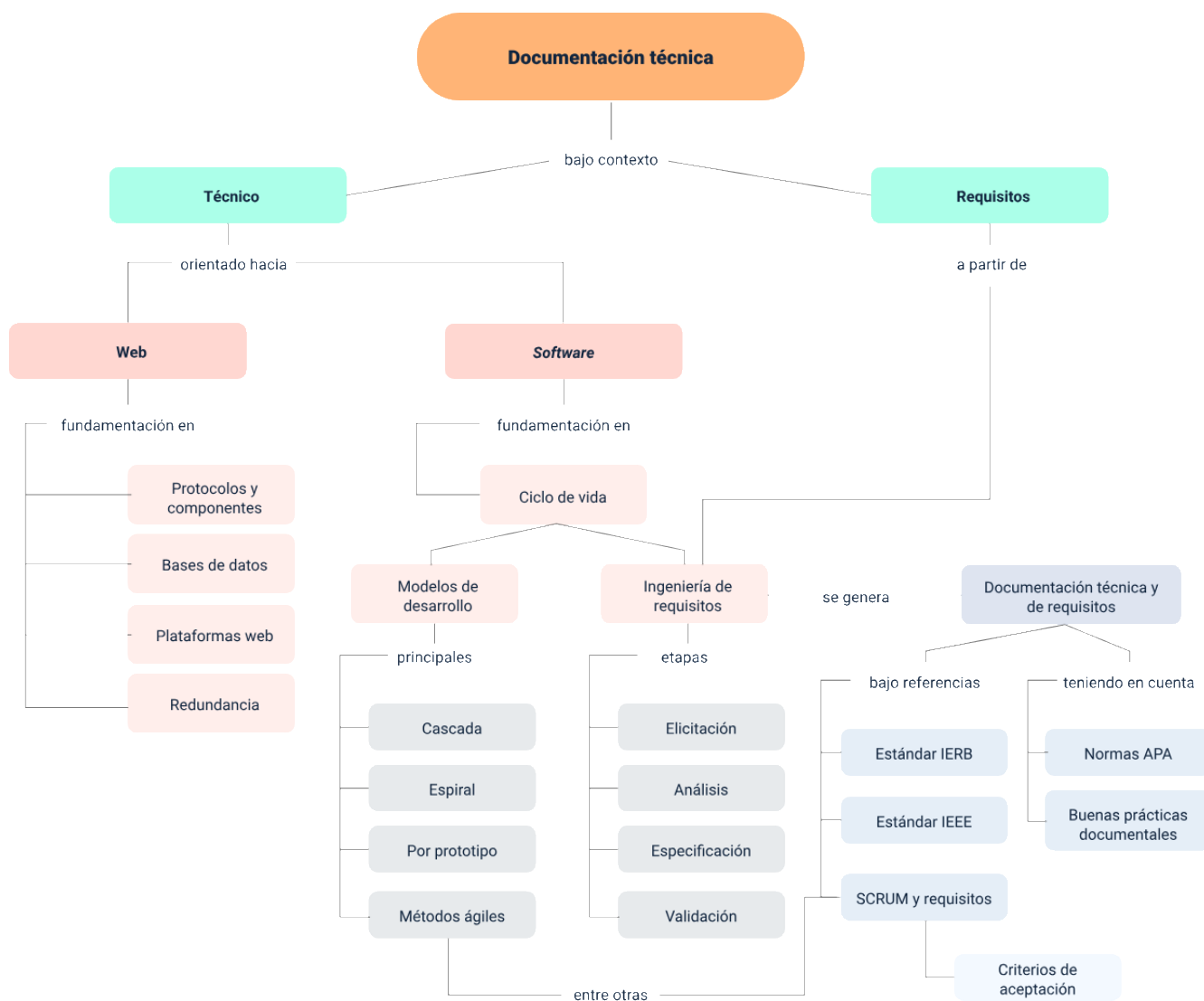
- **Tenga claridad de los insumos e instrumentos.** Antes de la documentación debe tener instrumentos aplicados en la organización como encuestas, entrevistas, diagramas, mapas de procesos y tomas de requerimientos. Ya debe tener una idea clara de la organización y de los procesos.
- **Empiece con los objetivos.** Se deben definir los objetivos y las metas, estos darán claridad y orden al documento. Cuando se construyen documentos, estos deben ser claros para la correcta interpretación de cualquier lector, tanto los clientes como el equipo de trabajo.
- **La forma importa.** La ortografía y la forma de redactar son primordiales a la hora de documentar, esto da cuenta de la seriedad, la técnica y aporta a la confianza en la gestión de los requerimientos en la organización.
- **Cite las referencias.** Es muy importante valorar las referencias, los marcos teóricos y técnicos con los que el proyecto se soporta. Recuerde que menos es más, evite redundar, procure escribir conciso, con pocas palabras puede dar claridad y se logrará más fácil el objetivo.
- **Claridad.** Redacte para todos, no emplee términos muy técnicos o poco usuales para descrestar, apóyese en recursos gráficos para describir mejor la idea; los recursos como mapas mentales, infografías, diagramas, tablas, etc. Hacen la lectura más simple y permite un mejor entendimiento.
- **Revise varias veces al final.** Cada vez que se relea un texto se va perfeccionando más, si tiene la oportunidad apóyese en otra persona que

lea el documento para determinar la claridad de los requisitos, y que pueda brindar sugerencias.



## Síntesis

A continuación, se presenta el diagrama que representa el resumen de las temáticas que están desarrolladas en el componente formativo.



## Material complementario

Tema	Referencia	Tipo de material	Enlace del recurso
Aplicaciones web	Ferrer, J. (2014). Aplicaciones web. RA-MA.	Libro	<a href="https://login.bdigital.sena.edu.co/login?url=https://www.ebooks7-24.com%2f%3fil%3d12589">https://login.bdigital.sena.edu.co/login?url=https://www.ebooks7-24.com%2f%3fil%3d12589</a>
Ciclo de vida del software	Hernández, M. y Baquero, L. (2020). Ciclo de vida de desarrollo ágil de software seguro. Fundación Universitaria Los Libertadores.	Libro	<a href="https://www-ebooks7-24-com.bdigital.sena.edu.co/?il=22372">https://www-ebooks7-24-com.bdigital.sena.edu.co/?il=22372</a>
Requisitos	iTunes U - UAEH. (2019). Tipos de requerimientos	Video	<a href="https://www.youtube.com/watch?v=PUyfeZsUSg">https://www.youtube.com/watch?v=PUyfeZsUSg</a>
Requisitos	CavernaTech. (2019). Requisitos funcionales y no funcionales de software.	Video	<a href="https://www.youtube.com/watch?v=Lv7XbZtnQ6A">https://www.youtube.com/watch?v=Lv7XbZtnQ6A</a>
Ingeniería de requisitos	Pressman, R. (2010). Ingeniería del software: un enfoque práctico. McGraw-Hill.	Libro	<a href="https://www-ebooks7-24-com.bdigital.sena.edu.co/?il=686">https://www-ebooks7-24-com.bdigital.sena.edu.co/?il=686</a>
Ingeniería de requisitos	Sommerville, I. (2011). Ingeniería de software. Pearson Educación.	Libro	<a href="https://www-ebooks7-24-com.bdigital.sena.edu.co/s tage.aspx?il=3313&amp;pg=&amp;ed=">https://www-ebooks7-24-com.bdigital.sena.edu.co/s tage.aspx?il=3313&amp;pg=&amp;ed=</a>

## Glosario

**Ágil:** comprende un conjunto de tareas o acciones que se utilizan para producir y mantener productos, así como para lograr los objetivos del proceso. La actividad incluye los procedimientos, los estándares, las políticas y los objetivos para crear y modificar un conjunto de productos de trabajo.

**Capa de red:** es la capa tres en del modelo OSI, se encarga de permitir la conexión entre dispositivos que están ubicados en redes diferentes. Es la capa de direccionamiento.

**Ciclo de vida** software: aplicación de metodologías para el desarrollo del sistema software (AECC, 1986).

**Método:** indica cómo construir técnicamente el software. Se incluyen técnicas de modelado y otras técnicas descriptivas.

**Metodología:** colección de métodos para resolver un tipo de problemas.

**Protocolo de comunicación:** conjunto de reglas que permiten la comunicación entre dos o más nodos en servicios específicos. En telemática los protocolos más usados son HTTP, FTP, SMTP, DNS, etc.

**Requerimiento:** se refiere a la petición que se hace de algo que se solicita.

**Requisito:** condición que debe cumplir algo, en general el requisito cumple con lo que se necesita con el requerimiento.

## Referencias bibliográficas

Boehm, B. (1979). A Spiral Model of Software Development and Enhancement. ACM Software Engineering Notes, 11(4), 22-42.

Durán, A. y Bernárdez, B. (2001). Metodología para el análisis de requisitos de sistemas software. DOCPLAYER. <https://docplayer.es/9147696-Metodologia-para-el-analisis-de-requisitos-de-sistemas-software.html>

Heras del Dedo, R. y Álvarez, A. (2017). Métodos ágiles: Scrum, Kanban, Lean. Difusora Larousse - Anaya Multimedia.

ISO/IEC 12207. (2008). Systems and software engineering - Software life cycle processes. ISO. <https://www.iso.org/obp/ui/#iso:std:iso-iec:12207:ed-2:v1:en>

Penzenstadler, B. (s.f.). Requirements Engineering. CSU Long Beach. California State University Long Beach <https://bit.ly/3rtBKXN>

Pfleeger, S. (2002). Ingeniería del software. Teoría y práctica. Prentice Hall.

Porfirio, C. (s. f.). Técnicas de priorización: el desafío de conseguir un orden para las funcionalidades. Atsisistemas - Consultoría it blog. <https://www.atsistemas.com/es/blog/tcnicas-de-priorizacin-el-desafio-de-conseguir-un-orden-para-las-funcionalidades>

Rivadeneira, M. (2014). Metodologías ágiles enfocadas al modelado de requerimientos. Informes Científicos Técnicos - UNPA, 5(1), 1-29. <https://doi.org/10.22305/ict-unpa.v5i1.66>

Sommerville, I. (2011). Ingeniería del software. Addison-Wesley.

## Créditos

Nombre	Cargo	Regional y Centro de Formación
Claudia Patricia Aristizábal	Líder del Ecosistema	Dirección General
Rafael Neftalí Lizcano Reyes	Responsable de Línea de Producción	Centro Industrial del Diseño y la Manufactura - Regional Santander
Jaime Hernán Tejada	Experto Temático	Centro de la Industria, la Empresa y los Servicios CIES - Norte de Santander
Giovanna Andrea Escobar Ospina	Diseñador Instruccional	Centro de la Industria, la Empresa y los Servicios CIES - Norte de Santander
Silvia Milena Sequeda Cárdenas	Asesora Metodológica y Pedagógica	Centro de Diseño y Metrología - Regional Distrito Capital
Julia Isabel Roberto	Corrección de Estilo	Centro de Diseño y Metrología - Regional Distrito Capital
Francisco José Lizcano Reyes	Desarrollador Full-Stack	Centro Industrial del Diseño y la Manufactura - Regional Santander
Yerson Fabian Zarate Saavedra	Diseñador de Contenidos Digitales	Centro Industrial del Diseño y la Manufactura - Regional Santander
María Carolina Tamayo López	Locución	Centro Industrial del Diseño y la Manufactura - Regional Santander
Oleg Litvin	Animador y Productor Audiovisual	Centro Industrial del Diseño y la Manufactura - Regional Santander
Wilson Andrés Arenales Cáceres	Storyboard e Ilustración	Centro Industrial del Diseño y la Manufactura - Regional Santander
Camilo Bolaño	Actividad didáctica	Centro Industrial del Diseño y la Manufactura - Regional Santander

Nombre	Cargo	Regional y Centro de Formación
Zuleidy María Ruiz Torres	Validación de Recursos Educativos Digitales	Centro Industrial del Diseño y la Manufactura - Regional Santander
Luis Gabriel Urueta Alvarez	Validación de Recursos Educativos Digitales	Centro Industrial del Diseño y la Manufactura - Regional Santander
Daniel Ricardo Mutis Gómez	Evaluador para Contenidos Inclusivos y Accesibles	Centro Industrial del Diseño y la Manufactura - Regional Santander