Fundamentos de análisis y diseño de software

Breve descripción:

En este componente formativo se abordan los saberes de ingeniería de requisitos, la fase de elicitación de requisitos y el análisis de requisitos (priorización, descomposición funcional, matriz de trazabilidad) y estándares, y/o guías existentes para la especificación formal de los mismos dependiendo del tipo de marco de trabajo usado (tradición o ágil).

Mayo 2024

Grupo de Ejecución de la Formación Virtual

Tabla de contenido

Introducción 6

1. Caracterización de procesos 7

1.1. Teoría general de sistemas 7

1.2. Enfoque de sistemas 8

1.3. Definiendo los sistemas 8

Clasificación de sistemas 9

Sistemas de información 12

Datos e información 14

1.4. Análisis de los procesos a nivel de negocio 14

Identificar los procesos 17

Determinar el equipo de trabajo 20

Generar los diagramas de los procesos de negocio 20

Realizar un diagnóstico de cada proceso 21

Puntos de mejora 21

Nuevo proceso mejorado 21

2. Ingeniería de requisitos 25

2.1. Ciclo de vida del software 28

Fases 29

Paradigmas de los modelos de ciclo de vida del software 30

Modelo Scrum 36

Modelo Kanban 38

Modelo XP o programación extrema 39

2.2. Fase de definición de requisitos.....41

2.3. Requisitos.....42

Importancia de los requisitos 42

Clasificación.....44

3. La fase de elicitación de requisitos 46

3.1. Planeación.....46

Identificar las fuentes de requerimientos 47

Identificar interesados del producto.....49

Matriz de stakeholders 52

3.2. Técnicas e instrumentos para elicitar requisitos.....56

Entrevista 57

Encuesta.....60

Observación.....62

Sesiones grupales.....64

3.3. Herramientas para captura de requisitos.....66

Diagrama de casos de uso 66

Historias de usuario 69

Storyboard 71

3.4. Herramientas de modelado 72

4. Técnicas de análisis y especificación de requisitos 77

4.1. Técnicas de análisis de requisitos 77

Priorización de requisitos 77

Técnica de clasificación de lista 78

Técnica de puntos de historia y valor del negocio 79

Técnica urgente 80

Técnica MoSCoW 82

Juicio de expertos 83

Matriz de priorización 84

Matriz de trazabilidad 85

Descomposición funcional 87

4.2. Especificación de requisitos 88

Estándar IEEE 830 88

Estándar IEEE 29148:2018 90

La especificación de requisitos a través de marcos de trabajo ágiles 92

Scrum y la especificación de requisitos 93

Kanban y la especificación de requisitos 97

Síntesis 100

Material complementario 101

Glosario 103

Referencias bibliográficas 106

Créditos 110

Introducción

Bienvenidos al estudio del presente componente formativo, en el cual se estudiará la caracterización de procesos, para continuar con la ingeniería de requisitos, entendiéndola como el conjunto de actividades y tareas del proceso de desarrollo de sistemas software, donde se definen que las características de un sistema software satisfagan las necesidades de negocio de clientes y usuarios y que se integre con éxito en el entorno en el que se trabaje.

Por este motivo, estudiaremos el ciclo de vida del software, sus fases y los diferentes paradigmas de los modelos de vida del software, haciendo énfasis en los requisitos, su importancia y clasificación.

Luego trataremos el tema de la recolección de datos, el cual se refiere al uso de una gran diversidad de técnicas y herramientas que pueden ser utilizadas por el analista para desarrollar los sistemas.

Continuaremos con el análisis de requisitos, documento donde se describe lo que se debe hacer, es el estudio de las necesidades. Allí, se estudiarán los principios básicos sobre los que se fundamenta el proceso de análisis de requisitos.

Conoceremos diferentes formas (planillas y estándares) para realizar el proceso de documentación de requisitos, así como diferentes metodologías ágiles que nos ayudarán en la especificación de requisitos.

Bienvenido, y le deseamos muchos éxitos en el proceso de aprendizaje.

1. Caracterización de procesos

La definición de procesos ayuda al entendimiento del funcionamiento de una organización, el análisis, moldeamiento, diagramación y mejora continua, son actividades esenciales en la estructura de una empresa que quiere aplicar desde un orden lógico las actividades diarias.

1.1. Teoría general de sistemas

Es un campo de la ciencia que pretende examinar las propiedades que definen a los sistemas; es decir, categorías formadas por partes interrelacionadas que llevan al cumplimiento de una acción u objetivo.

Esta teoría se constituyó como un nuevo aporte dentro del conjunto de paradigmas científicos, para el abordaje de problemas que se basan en la correspondencia entre los elementos que conforman los sistemas. Antes de esta propuesta se pensaba que los sistemas, desde el punto de vista de su conformación global, se asemejaban a la sumatoria de sus elementos integrantes y que podían ser entendidos, disgregándolos para realizar un estudio individual de cada componente.

Desde sus inicios, la teoría general de sistemas se ha utilizado en diferentes áreas de la ciencia, la ingeniería, las matemáticas, la biología, la política, las ciencias sociales, las ciencias de la computación, la economía y otras ciencias exactas y sociales, primordialmente en el marco del análisis de las correlaciones que se presentan en sus componentes y su interoperabilidad.

1.2. Enfoque de sistemas

Un puntal básico, unido a los respectivos análisis (simulación, teoría de comportamientos, teoría de colas, teoría de juegos), que se pueden aplicar para la resolución de problemas usando el método científico, es el enfoque de sistemas:

"Se trata de comprender el funcionamiento de una organización u objeto de estudio desde una perspectiva holística e integradora, en donde lo importante son las relaciones entre los componentes. Se llama holismo al punto de vista que se interesa más por el todo que por la suma de las partes. El enfoque sistémico no concibe la posibilidad de explicar un elemento si no es precisamente en su relación con el todo. Metodológicamente, por tanto, el enfoque sistémico es lo opuesto al individualismo metodológico, aunque esto no implique necesariamente que estén en contradicción". (Centro Tic Junta de Andalucía, 2021).

1.3. Definiendo los sistemas

"Sistema es un conjunto organizado de elementos que interactúan entre sí con estructura lógica o que son interdependientes, formando un todo complejo, identificable y distinto. Por los elementos de un sistema se entiende no solo su conformación física, sino las funciones que estos desempeñan. Algún conjunto de elementos de un sistema puede ser considerado un subsistema si mantienen una relación entre sí que los hace también un conjunto identificable y distinto". Ludwig Von Bertalanffy (1968).

Tales sistemas se caracterizan por su capacidad de recepción de elementos, los cuales son denominados entradas; a su vez, se pueden tipificar como tipos de energía, información a manera de mensajes, datos y señales, también recursos físicos. Estas

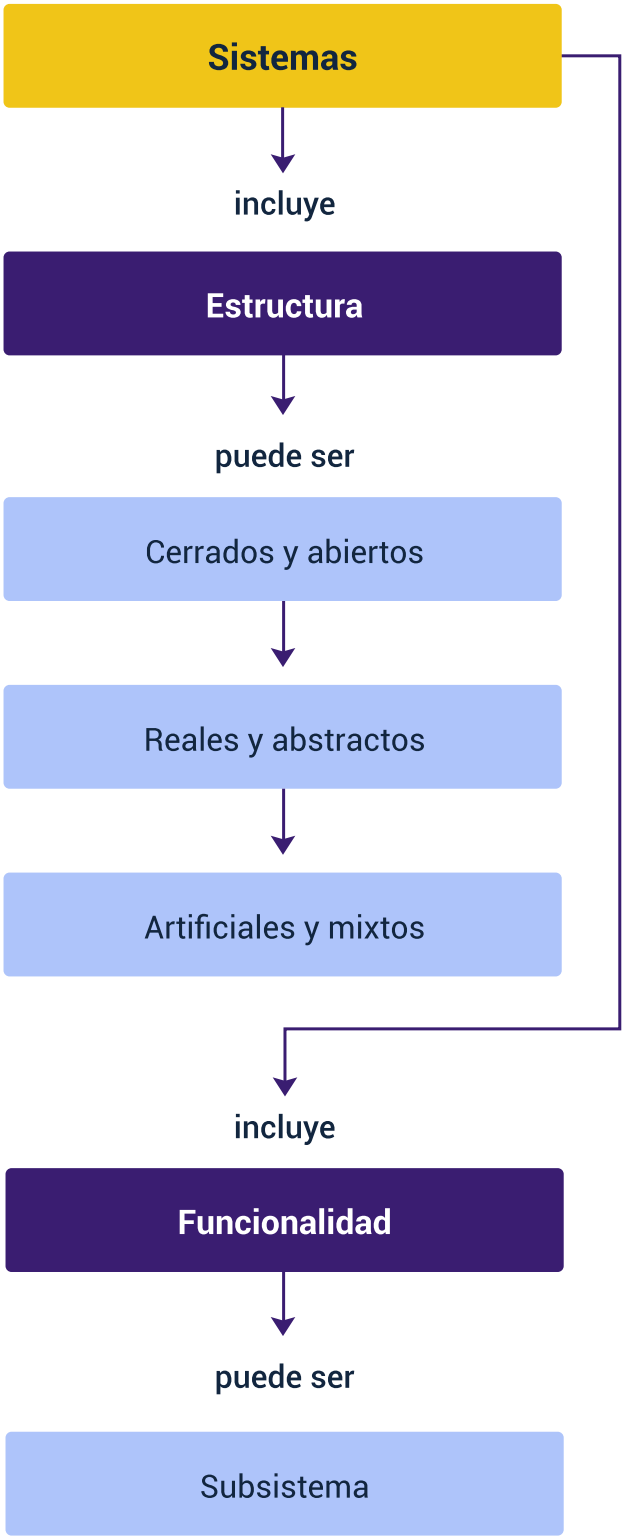
entradas se someten a un conjunto de actividades que alimentan una serie de acciones que las transforman y como consecuencia, se generan unos resultados o salidas; una situación muy interesante se presenta cuando una salida o parte de ella se convierte nuevamente en entrada. Cuando esto sucede, se dice que existe retroalimentación. La retroalimentación ayuda a elevar el grado de perfeccionamiento en las respuestas y en el propio comportamiento del sistema, logrando de manera inmediata altos niveles de control, principio básico de la automatización de estos.

Así mismo, cuando de un subsistema se conocen solo las entradas y las salidas, pero no las actividades internas, se dice que es una caja negra, en ese caso se aconseja utilizar la observación y guiarse por el sentido común para proponer las actividades desde cero, que resuelven el mismo problema.

Clasificación de sistemas

Bertalanffy y otros autores posteriores, han definido distintas formas de clasificar a los sistemas en función de su conformación estructural y sus funcionalidades. La siguiente figura presenta las clasificaciones más importantes:

Figura 1. Definiciones de sistemas



A continuación, se presentan algunos tipos de sistemas:

- **Sistema, supersistema y subsistemas**

  Los sistemas se pueden agrupar teniendo en cuenta su grado de complejidad; se componen a su vez de distintos niveles de jerarquía en que pueden dividirse y que interactúan entre ellos intercambiando algún tipo de información, de modo que estos al final no son autónomos unos de otros.

  Si se entiende por sistema un conjunto de partes, podemos hablar de subsistemas para referirse a tales elementos; un ejemplo de ello es: la conformación de una familia como un sistema y cada miembro en ella es un subsistema. El supersistema es el contexto externo que rodea al sistema y en el que este se encuentra inmerso; en los grupos o comunidades de humanos es identificable con la sociedad.

- **Reales físicos, ideales y abstracciones**

  De acuerdo con su esencia, los sistemas se pueden clasificar en reales físicos, como producto del pensamiento ideal y modelo. Los sistemas reales son aquellos que existen físicamente y que pueden ser objeto de observación y por tanto, objetos de medición, mientras que los sistemas ideales son las representaciones simbólicas extraídas de la abstracción que produce el pensamiento y el lenguaje, que se traducen en diseños y modelos. Las abstracciones y sus correlaciones pretenden representar características reales e ideales, aquí es importante tener en cuenta el conjunto de requerimientos, solicitudes, la definición de límites con las cuales se pretende llegar a los objetivos finales.

- **De la naturaleza, artificiales y mixtos**

  Cuando un sistema está inmerso o corresponde únicamente al funcionamiento de la naturaleza, como lo es un lago, un río, la selva, el cuerpo humano, los planetas o las galaxias, entonces hay que referirse a ellos como sistemas naturales. Por el contrario, los sistemas artificiales son aquellos que emergen de la creación del hombre, es decir, como productos de la acción humana.

  Los sistemas mixtos se dan por la combinación de elementos naturales y artificiales. Cualquier entorno de orden físico que sufre alteraciones dadas por el ser humano, como lo son los pueblos, las ciudades, los ríos en donde se construyen represas, es considerado un sistema mixto; el grado de participación de elementos naturales y artificiales varía en cada caso concreto en proporciones diferentes.

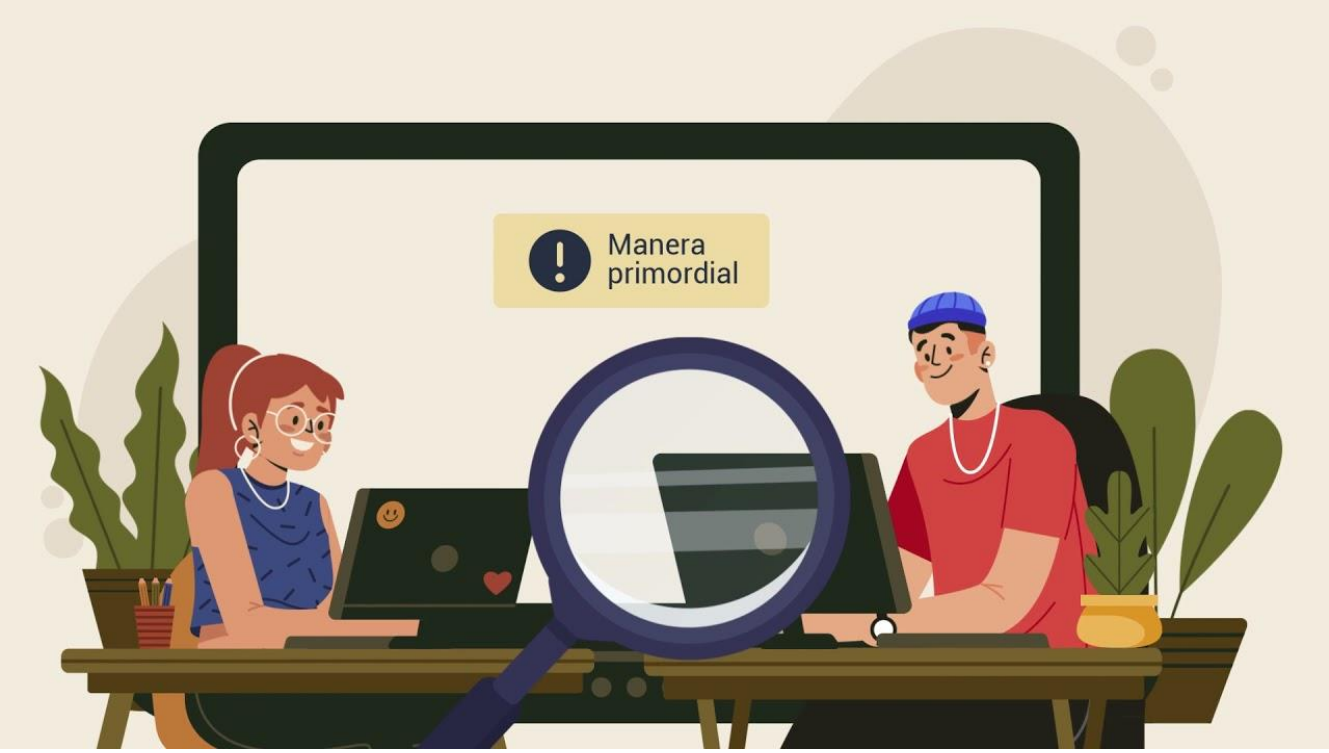
- **Cerrados y abiertos**

  Torres (2021) argumenta: “El criterio principal que define a un sistema es el grado de interacción con el supersistema y otros sistemas. Los sistemas abiertos intercambian materia, energía y/o información con el entorno que los rodea, adaptándose a estos e influyendo en él. En cambio, los sistemas cerrados se encuentran teóricamente alejados de las influencias ambientales; en la práctica se habla de sistemas cerrados cuando están altamente estructurados y la retroalimentación es mínima, puesto que ningún sistema es completamente independiente de su suprasistema”.

Sistemas de información

A continuación, se presenta un breve significado de sistemas de información:

Video 1. Sistemas de información



Enlace de reproducción del video

Síntesis del video: Sistemas de información
El video presenta el concepto de sistemas de información en el contexto de la caracterización de procesos, los cuales son gestionados por procesamiento electrónico de datos que permiten solucionar problemas de la vida real. Convergiendo en las últimas técnicas de procesamiento de datos, big data, inteligencia artificial, minería de datos, domótica entre otros, unido al desarrollo de software, donde es importante analizar su inter-operatibilidad aprovechando funcionalidad y minimizar recursos.

Datos e información

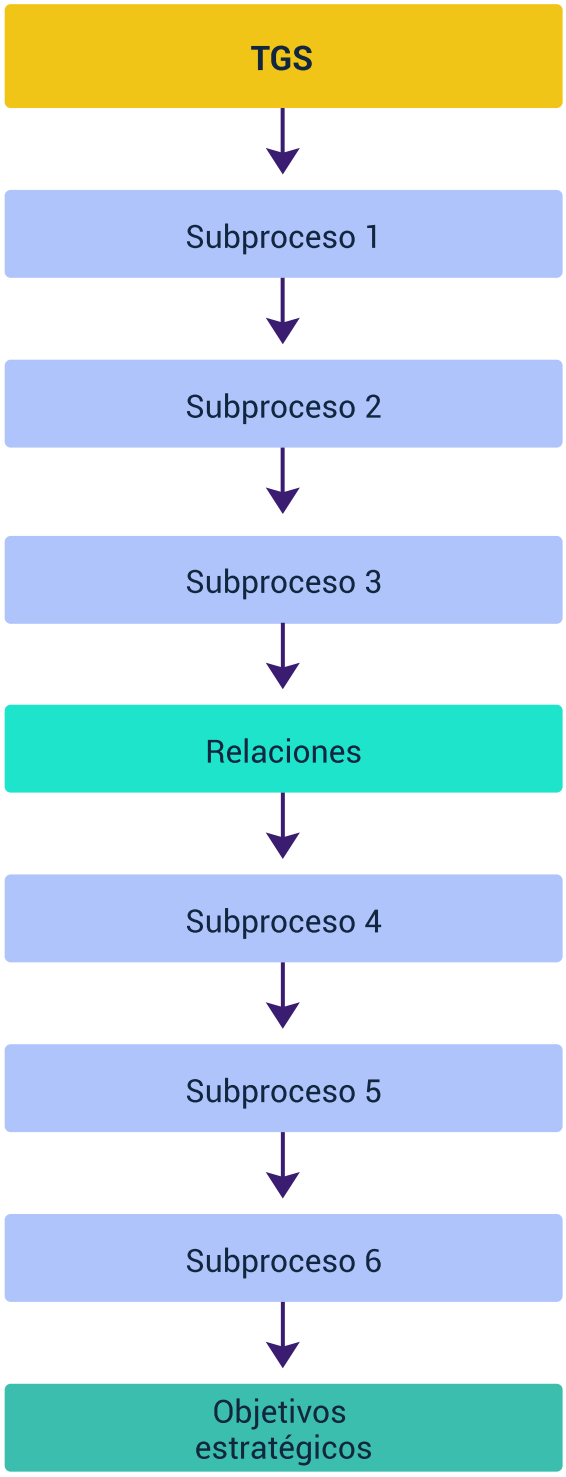
Estos términos tienden a confundirse, pero a la luz de los sistemas que estamos tratando, vale la pena definirlos:

- Un dato no es otra cosa que una representación simbólica de alguna situación o suceso, sin ningún sentido semántico, describiendo un hecho concreto. O lo que es lo mismo, sin transmitir mensaje ninguno, un ejemplo de dato podría ser una letra o un hecho.
- La información es un conjunto de datos, los cuales son adecuadamente procesados, para que puedan proveer un mensaje que contribuya a la toma de decisiones a la hora de resolver un problema o afrontar una situación cualquiera, en la que se requiera de la toma de decisiones de cualquier tipo.

1.4. Análisis de los procesos a nivel de negocio

Con base en los conceptos de la teoría general de sistemas y partiendo de la visión del todo, se debe comprender el funcionamiento real a partir de desestructurar ese todo en procesos, subprocesos, relaciones y actores que intervienen, con el objetivo de identificar, comprender, evaluar y resolver problemas que ayudan a las instituciones a mejorar dichos procesos, reformar las opciones de negocio y brindar a sus consumidores a experimentar una mejor calidad de productos o servicios. Por esto, es necesario que realice de manera constante, frecuente y repetitiva, un análisis de sus procesos de negocio desde la TGS, donde visualicen y comprendan cada uno de los pormenores que transforman el negocio, con el objetivo de mejorar sus operaciones y el servicio al cliente. A continuación, se presenta una gráfica de la explicación.

Figura 2. Análisis del proceso desde la TGS



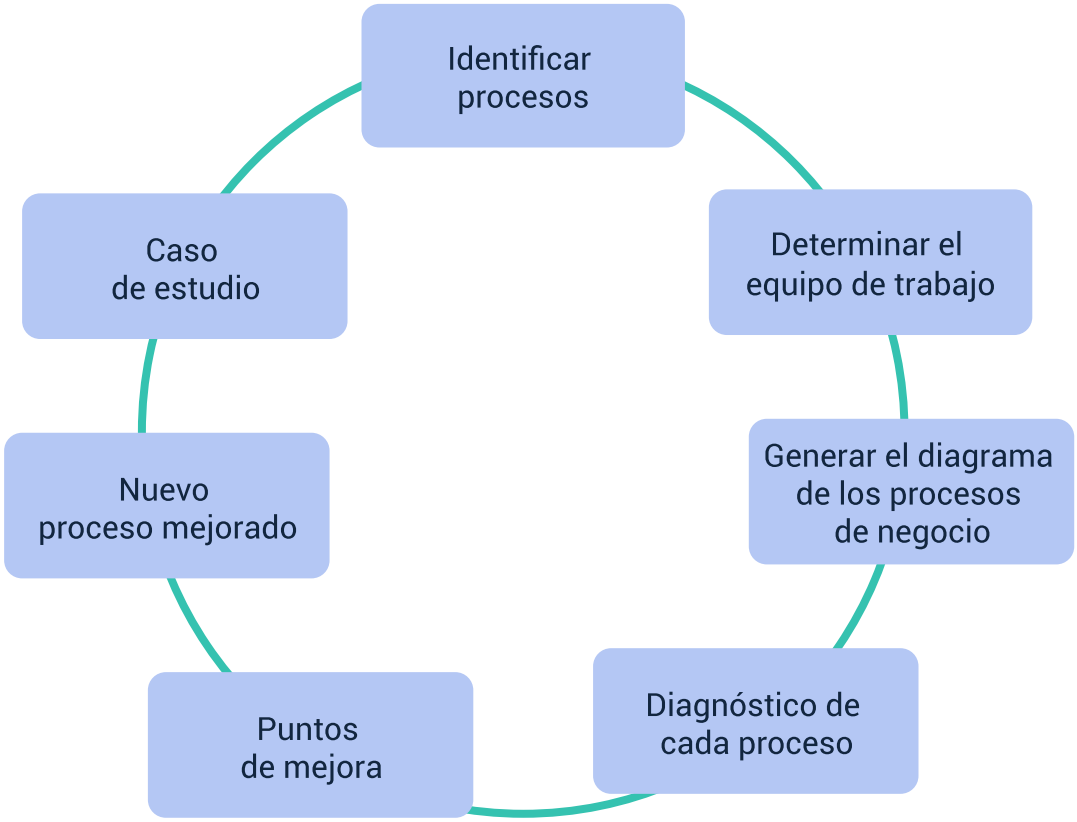
Una de las técnicas que permite conseguir estos objetivos, es el análisis de procesos de negocio, que significa llegar al conocimiento detallado de cada labor y se hace abordando la operación central de valor hasta las que pueden considerarse como labores rama o periféricas, no menos importantes. Ese es el nombre que se le da a la acción que realizan las organizaciones para revisar, documentar y entender sus procesos.

Tiene como objetivo el desglose de cada uno de los componentes en cada proceso:

- Entradas
- Salidas
- Procedimientos
- Controles
- Aplicaciones
- Datos

Es importante comprender las acciones que integran cada uno de los siguientes elementos:

Figura 3. Acciones para el análisis



Ahora, conozcamos de manera detallada, cada proceso:

Identificar los procesos

Para realizar una excelente identificación de procesos, se recomienda realizar las siguientes actividades en dos escenarios:

- **Escenario 1**

  Antes: se debe obtener acceso de información sobre la organización, para antes de llegar a esta, se investigue sobre el contexto de negocio en el que se mueve la empresa. Por ejemplo: determinar si se está hablando de un banco o una empresa de energía o una empresa que presta servicios y en ese caso, qué tipo de servicios presta o, por el contrario, si es una empresa

que fabrica productos, qué productos fabrica, es decir, qué hace. Conocer de ella también sus sedes (instalaciones físicas).

- **Escenario 2**

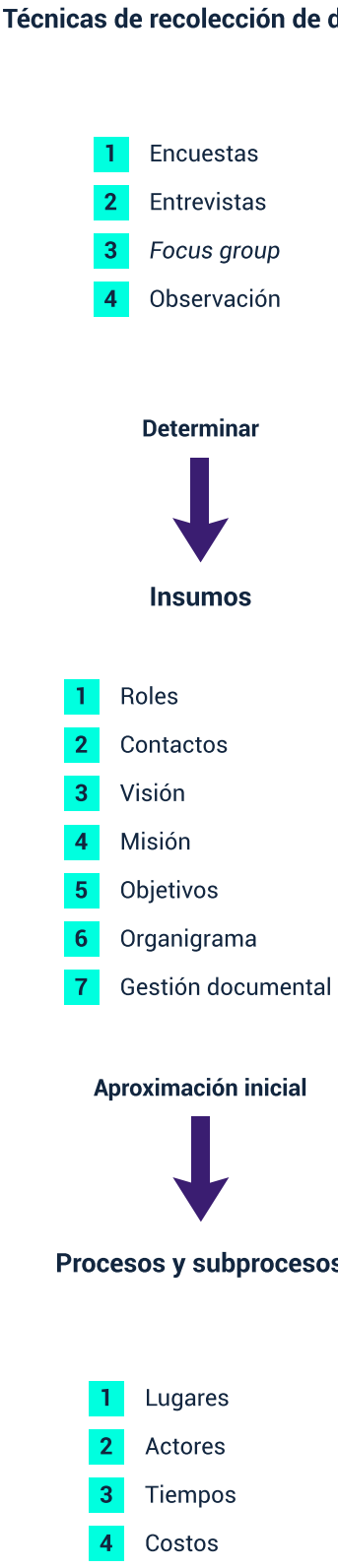
  Cuando exista cercanía, acceso o contacto con la entidad, planear entonces la recolección de información, aplicando las técnicas que existen para ello, como son:

  - Método de la observación.
  - La elaboración de formatos de encuestas.
  - El enfoque de trabajo (focus group).
  - Las entrevistas.

Como resultado de estas técnicas, se deben establecer claramente y como mínimo los siguientes ítems: roles de los intervinientes, información de contactos con responsabilidades, misión, visión, objetivos estratégicos, estructura organizacional un organigrama de la entidad su distribución de dependencias u oficinas, en donde se puede ubicar de manera ágil a los responsables, que seguramente se convertirán en futuros miembros de los equipos de trabajo donde se presentará interacción.

Con lo anterior dispuesto, se puede dar el primer paso fundamental que corresponde a identificar, conocer, incluso ubicar, los sitios donde suceden en el mundo real, aquellos procesos que necesitan ser mejorados, descritos, analizados o propuestos para generar mejoras.

Figura 4. Pasos para tener en cuenta



Determinar el equipo de trabajo

Con base en la información recolectada y a la obtención de los requerimientos que permitieron identificar los procesos que necesitan mejorarse, se procede a formar el equipo de trabajo que los identificará, analizará y documentará. Lo ideal es que los responsables que conocen a detalle los procesos, conformen el equipo.

Por lo tanto, el equipo debería estar integrado por el siguiente grupo humano:

a) El líder del proyecto o el proceso elegido.
b) Los empleados vinculados al proceso.
c) Un alto miembro de la organización que conoce los objetivos estratégicos.
d) Los trabajadores de otras áreas que cuentan con la actual experiencia y que desean conocer las mejoras que se llevarán a cabo.

Generar los diagramas de los procesos de negocio

Se trata de representar en una secuencia lógica, los elementos expresados en la TGS, nombre del proceso, el cual debe ser un verbo (fabricar, validar, informar) que permite modelar en un dibujo, el proceso de manera rápida, dinámica y sencilla acerca de cómo fluyen las entradas en cada elemento y hacia dónde se deben dirigir sus salidas. En el diagrama es fundamental definir las competencias de cada uno de los integrantes del equipo en cada tarea.

Para ampliar la información sobre este tema, lo invitamos a consultar el PDF "**Generar los diagramas de los procesos de negocio**" , el cual se encuentra en la carpeta Anexos.

Realizar un diagnóstico de cada proceso

Esta labor permite a las empresas detectar, por medio la implementación de estrategias, los problemas de funcionamiento y deficiencias. Estas estrategias son:

a) Entender las causas propias que maneja los casos de cada proceso.
b) Ejecutar simulacros de relevamiento de procesos unido a sus responsables.
c) Apoyarse en herramientas de software para registrar resultados, controles, métricas, análisis de tiempos, costos y materias primas.

Puntos de mejora

Luego del diagnóstico de la situación actual, es importante construir un informe que dé a conocer el estado anterior y el estado actual hacia dónde se quiere llegar con las mejoras. En este informe se recomienda, de manera obligatoria, el poder relacionar los alcances de las soluciones unidos en forma transversal con los objetivos estratégicos de la organización.

Nuevo proceso mejorado

Los resultados obtenidos de la aplicación de cada uno de los pasos anteriores, se deben compilar y almacenar para convertirse en información de salida, con el objetivo de llevar trazabilidad. Al realizar encuentros o reuniones de análisis de dicha información, se genera el diseño de un nuevo proceso, el cual estará destinado al logro de los objetivos estratégicos de la compañía y así llegar a un estadio de mayor eficiencia y eficacia.

Tabla 1. Sistemas de compras en línea

Entradas	Subproceso o actividad	Salidas
Datos del solicitante (nombres, apellidos, identificación, dirección de contacto). Producto que comprar (valor, cantidad).	Recepción de solicitud de compra.	Número único de identificación de la compra. Factura. Envío.
Número único de identificación de la compra.	Almacén.	Producto.
Número único de identificación de la compra.	Almacén.	Producto.
Insumos. Mano de obra.	Fábrica.	Producto elaborado.
Facturas.	Contabilidad	Informes contables.

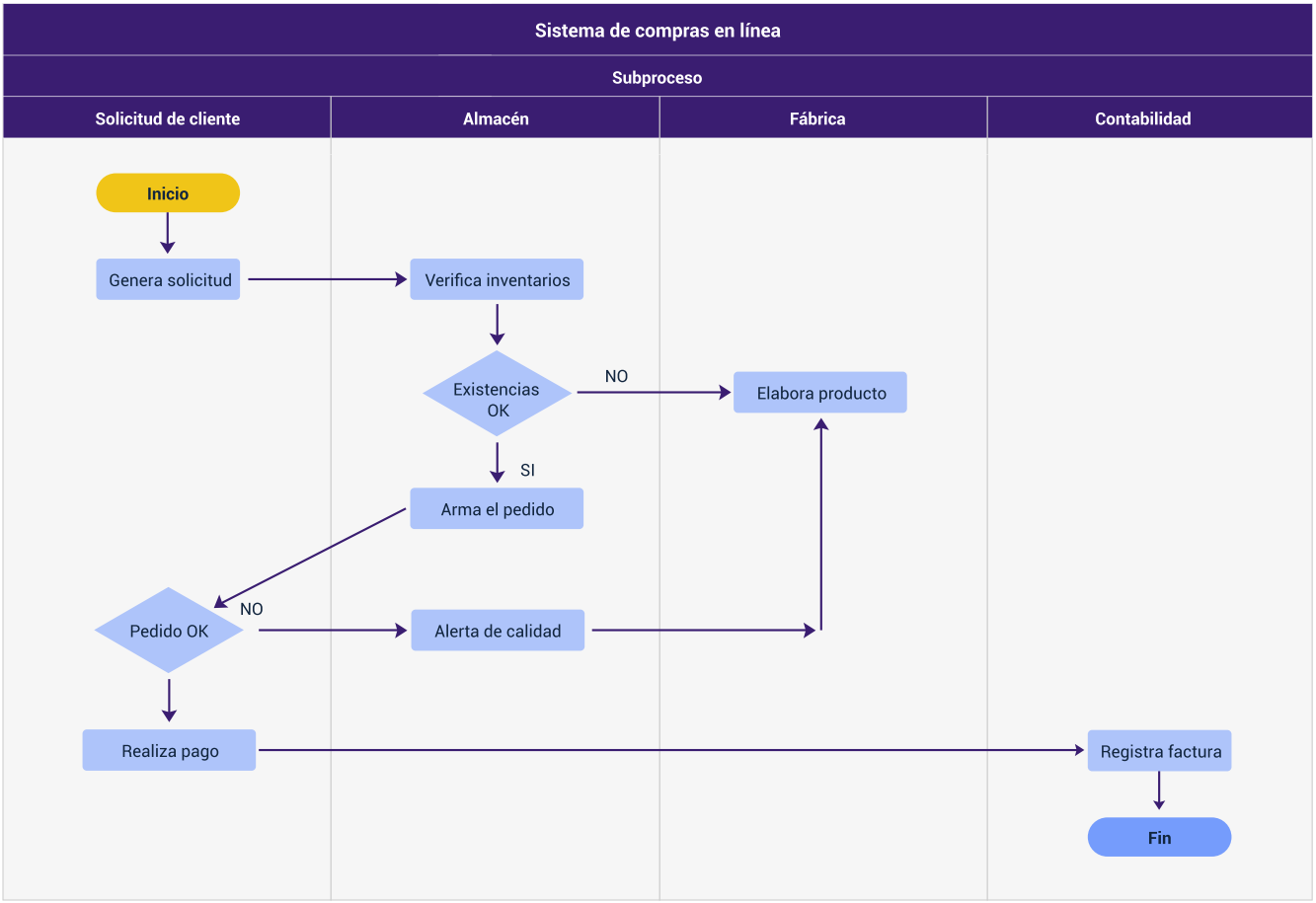
Puesto que el sistema de compras en línea no intercambia elementos de información con el medio ambiente que lo rodea, se considera un sistema cerrado. De lo anterior se concluye:

- **Se pueden inferir las siguientes relaciones:**
  a) Usuario solicitante contra el proceso de solicitud de compra, en donde un usuario puede generar una o muchas solicitudes de compras.
  b) El almacén maneja el inventario de productos.
  c) La fábrica provee de muchos productos al almacén.
  d) El almacén promociona el producto para la solicitud.
  e) El usuario realiza el pago del producto.
  f) Contabilidad registra la factura de compra generada por la solicitud.

- **Actores responsables:**
    a) Usuario que genera la solicitud.
    b) Jefe de ventas.
    c) Administrador del sistema online.
    d) Jefe de contabilidad.
    e) Jefe del control de operaciones de fábrica.
    f) Jefe de inventarios.
    g) Empleados de la fábrica.
    h) Gerencia de área.

La siguiente imagen nos presenta el ejemplo del diagrama de procesos.

Figura 5. Ejemplo del diagrama de procesos



2. 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

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

- **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.

2.1. 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 establecidos.

Fases

Las fases del modelo de ciclo del software son: planificación, análisis, diseño, implementación, pruebas y mantenimiento, las cuales se describen a continuación.

- **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.

- **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.

- **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.

- **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.

- **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.

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.

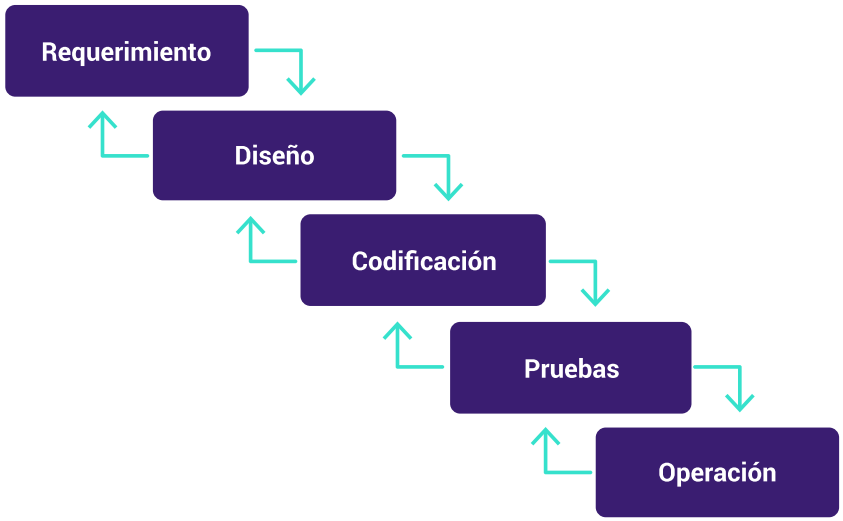
A continuación, se revisan los modelos del paradigma tradicional más utilizados.

- **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 6. Modelo de cascada del ciclo de vida del desarrollo



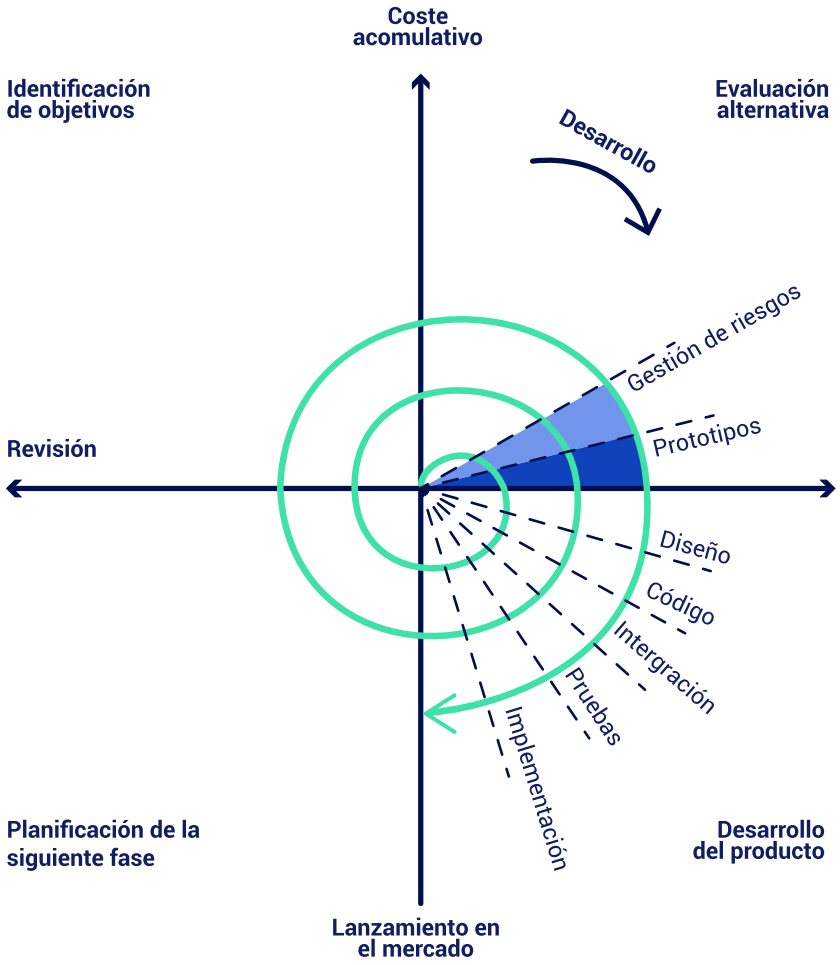
- **Modelo espiral**

  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 7. Modelo espiral del ciclo de vida del desarrollo



Nota. Adaptada de Boehm (2018).

- **Modelo iterativo 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).

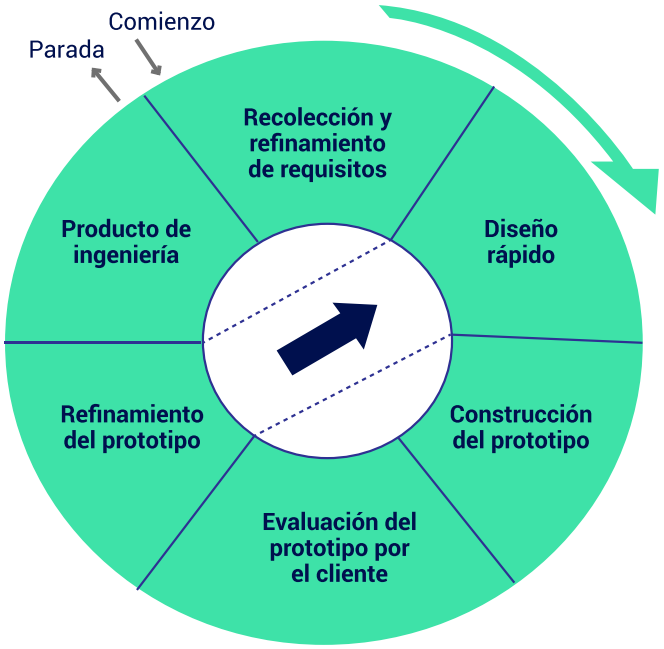
  Objetivos

  - 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.

  Las etapas del modelo son:

  - Colecta y refinamiento de los requerimientos y proyecto rápido.
    - Análisis.
    - Especificación del prototipo.
  - Diseño rápido.
  - Construcción del prototipo.
  - Evaluación del prototipo por el cliente.
  - Refinamiento del prototipo.
    - Diseño técnico.
    - Programación y test.
    - Operación y mantenimiento.
  - Producto de ingeniería.

Figura 8. Modelo 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:

- Product Backlog.
- Sprint Backlog.
- Sprint Planning Meeting.

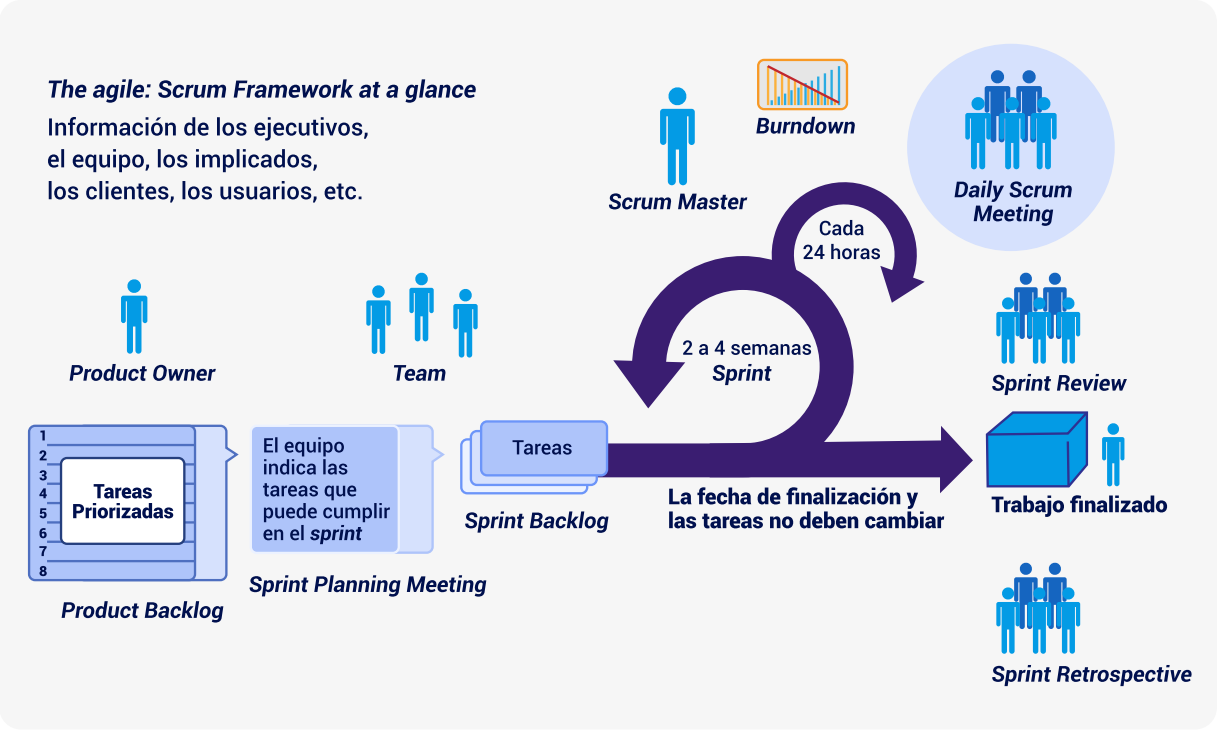
- Daily Scrum.
- Sprint Review.
- Sprint Retrospective.

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 9. Modelo ágil Scrum



Modelo Kanban

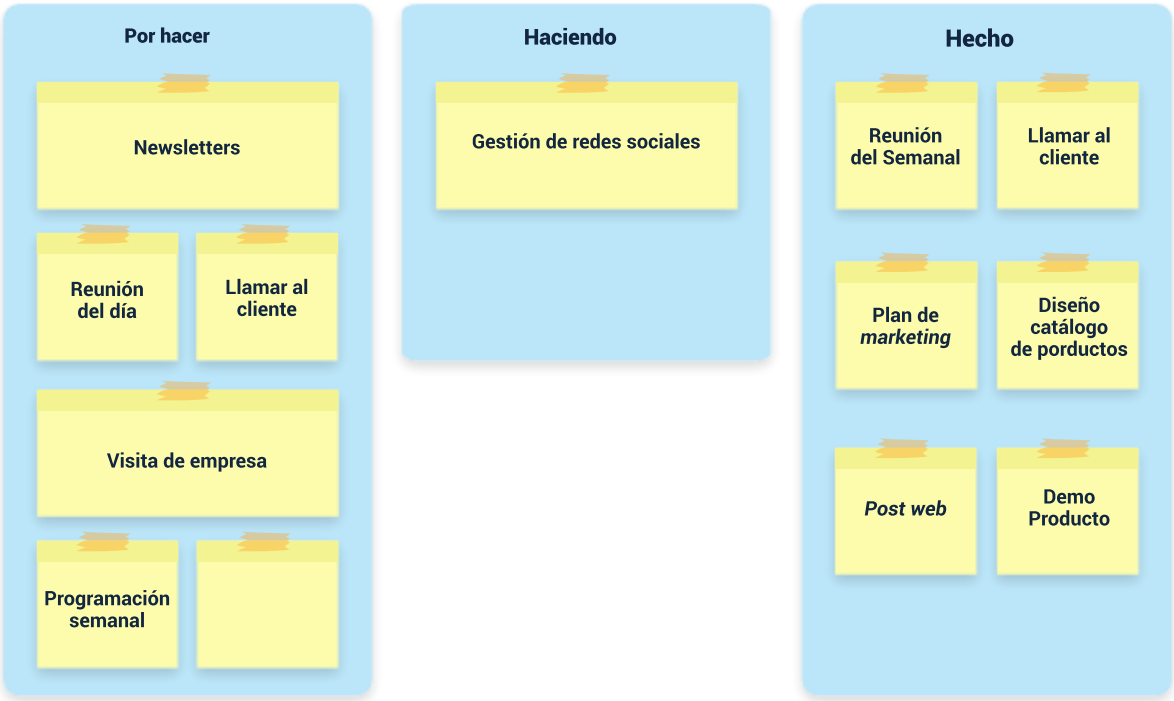
David J. Anderson (reconocido como el líder de pensamiento de la adopción del Lean/Kanban para el trabajo de conocimiento), formuló el método Kanban como una aproximación al proceso evolutivo e incremental y al cambio de sistemas para las organizaciones de trabajo. El método está enfocado en llevar a 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 10. Modelo ágil Kanban



Nota. Adaptada de Anderson.

Modelo XP o programación extrema

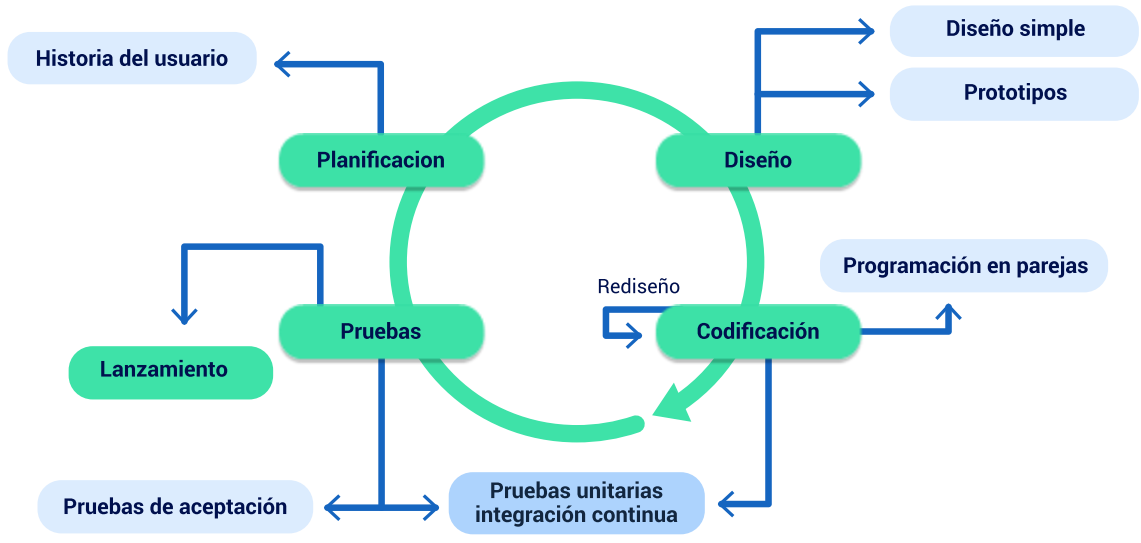
La programación extrema o eXtreme Programming (XP) es un enfoque de la ingeniería de software, formulado por Kent Beck, autor del primer libro sobre este tema: Extreme Programming Explained: Embrace Change (1999). Esta metodología es adaptable según las necesidades y requerimientos a implementar,

además, el cliente se encuentra involucrado en el proceso de desarrollo, lo que hace que el producto pueda ser terminado en un menor tiempo.

Características principales de la programación extrema:

- Tipo de desarrollo iterativo e incremental.
- Pruebas unitarias.
- Trabajo en equipo.
- Trabajo junto al cliente.
- Corrección de errores.
- Reestructuración del código.
- El código es de todos.
- El código simple es la clave.

Figura 11. Modelo XP



Nota. Adaptada de Muradas (2020).

2.2. 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 comúnmente se presenta al inicio de la etapa de desarrollo, es que el cliente no sepa exactamente que necesita, por tanto, se debe averiguar con ayuda de diferentes técnicas.

Por otra parte, la inestabilidad de los requerimientos de un sistema es inevitable, porque se estima que el 25 % de los requerimientos iniciales de un sistema, cambian antes 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 2. Actividades y artefactos de la fase de definición de requisitos

Fase	Actividades	Artefactos
Análisis (definición de requisitos).	Definición del alcance del proyecto.	
Análisis (definición de requisitos).	Identificación del negocio.	Modelo del negocio.
Análisis (definición de requisitos).	Toma de requerimientos.	Análisis y realización de casos de uso.
Análisis (definición de requisitos).	Estudio de procesos de negocio.	Modelo de procesos y actividades de negocio.
Análisis (definición de requisitos).	Calendarización del proyecto.	Cronograma del proyecto.

2.3. 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.

Importancia de los requisitos

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

- 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).

- Indican al equipo del proyecto qué requieren los usuarios (necesidades de negocio).
- 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.

A continuación, 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:** categorizar 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.
- **Rastreable:** 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.

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 3. 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.

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

3. La fase de elicitación de requisitos

El propósito de la elicitación de requerimientos es ganar conocimientos relevantes del problema, que se utilizarán para producir una especificación formal del software necesario para resolverlo.

"Un problema puede ser definido como la diferencia entre las cosas como se perciben y las cosas como se desean". (Gause y Weinberg 1989).

Aquí se ve la importancia que tiene una buena comunicación entre desarrolladores y clientes; de esta comunicación con el cliente depende que sus necesidades queden claras. Además, al final de la fase de análisis de requerimientos, el analista podría llegar a tener un conocimiento extenso en el dominio del problema.

La elicitación de requisitos es la actividad que se considera como el primer paso en un proceso de ingeniería de requisitos; su significado hace referencia a la puesta en marcha de técnicas que sirven para recopilar conocimiento o información y los objetivos de esta fase de elicitación, son:

- Conocer el dominio del problema para poder comunicarse con clientes y usuarios y entender sus necesidades.
- Conocer el sistema actual (manual o informatizado) y sus aspectos positivos y negativos.
- Identificar las necesidades, tanto explícitas como implícitas, de clientes y usuarios y sus expectativas sobre el sistema a desarrollar.

3.1. Planeación

La planeación busca definir las tareas a realizar para elegir y planificar las técnicas a emplear durante la actividad de elicitación de la fase de ingeniería de requisitos del

desarrollo de software. En la siguiente tabla se presenta una relación de estas tareas y sus correspondientes procesos.

Tabla 4. Tareas para elicitación de requisitos

Tareas	Proceso
Identificar las fuentes.	Lista de fuentes de requerimientos.
Identificar interesados del producto.	Categorías de los interesados (stakeholders).
Matriz stakeholders (Describir necesidades y criterios de éxito).	Perfil de stakeholders.
Revisar técnicas.	Identificar combinaciones de técnicas, entrevistas, grupos focales, encuestas, prototipos.
Captura de interesados.	Plan de captura de interesados.

Nota: tomado de Durán y Bernárdez (2001)

A continuación, se describen los procesos relacionados con las tareas para elicitación de requisitos:

Identificar las fuentes de requerimientos

Existe un conjunto de fuentes de requisitos en cada proyecto de desarrollo de software, así, usuarios y expertos abastecen de información detallada acerca del problema y necesidades del usuario. Los procesos y sistemas existentes representan, también, fuentes de requisitos; además, la documentación existente como manuales, formularios y reportes, incluso especificaciones de requisitos anteriores, puede proveer información útil acerca de la organización y su entorno.

En el proceso de esta actividad se identifican:

- Interesados relevantes.
- Documentación que se puede usar como fuente de información de los requerimientos.
- Fuentes de información externas.

Las fuentes de requerimientos incluyen los propietarios del problema, los stakeholders, documentos y otros sistemas (Pearson, 2002). En ese sentido, los requerimientos pueden obtenerse en diversas fuentes que pueden clasificarse en gente (people), productos o documentos, pero cualquiera sea la fuente de esos requerimientos deben ser chequeados con los stakeholders.

Estas fuentes de requerimientos, se pueden clasificar en:

- **Fuentes primarias**

  Aportan material de primera mano (es protagonista o testigo de los hechos), estas fuentes contienen información original, que ha sido publicada por primera vez y que no ha sido filtrada, interpretada o evaluada por nadie más.

- **Fuentes secundarias**

  Toman y reproducen la información que le aportó una fuente primaria. Son las que contienen información primaria, sintetizada y reorganizada y están especialmente diseñadas para facilitar y maximizar el acceso a las fuentes primarias o a sus contenidos. Parten de datos preelaborados, como pueden ser datos obtenidos de anuarios estadísticos, internet, medios de comunicación, bases de datos procesadas con otros fines, artículos y documentos relacionados con un tema, libros, tesis, informes oficiales, etc.

- **Fuentes terciarias**

  Son guías físicas o virtuales que contienen información sobre las fuentes secundarias. Forman parte de la colección de referencia de una biblioteca; facilitan el control y acceso a toda la gama de repertorios de referencia, como las guías de obras de referencia, o a un solo tipo, como las bibliografías.

Por otra parte, las fuentes de información, pueden ser orales, escritas o de otro tipo, dependiendo de cómo se transmitan los datos. A continuación, se pueden revisar algunos ejemplos de fuentes de información.

Las fuentes de información o documentación pueden hallarse en diversos soportes.

- Grabaciones audiovisuales y grabaciones auditivas.
- Libros, artículos, prensa escrita y básicamente cualquier tipo de soporte que permita capturar y preservar la información, para recuperarla luego.
- Los testimonios, los relatos, las reseñas, los ensayos, las páginas web, las reflexiones, los listados bibliográficos y los índices.
- Las grabaciones profesionales, accidentales o clandestinas, las fotografías, las filmaciones e incluso ilustraciones.

Identificar interesados del producto

Uno de los primeros pasos en el proceso es el análisis e identificación de todas las personas relevantes que tienen un grado de interés en el proyecto. Los interesados (stakeholders), son los individuos y organizaciones que están relacionados activamente en un proyecto de software; tienen influencia directa o indirecta sobre

los requisitos, o sus intereses se ven afectados por el proyecto (Baar, 2006, Ventura, 2002).

En resumen, son grupos o individuos que están interesados en el producto de software que se está desarrollando y necesitarán estar informados acerca del progreso, conflictos, cambios y prioridades del proceso de desarrollo del producto.

Los stakeholders se dividen en dos grupos:

- **Primarios**

  Son aquellas personas indispensables para el correcto funcionamiento de la organización, y tienen una relación económica directa con la empresa. Estos pueden ser sus socios, clientes y accionistas.

- **Secundarios**

  Son los entes que no participan directamente de la compañía, pero también son afectados por sus resultados. En este círculo se encuentran los competidores, el mercado y las personas en general.

A continuación, se listan roles más generales de las personas involucradas con sus términos similares, aunque cabe resaltar que existen leves diferencias entre ellos (Sommerville y Sawyer, 2005):

- Líder de proyecto / Administrador de proyecto / Gerente de proyecto.
- Analista / Ingeniero de requisitos.
- Ingeniero de sistemas / Arquitecto.
- Programador / Desarrollador / Ingeniero de software.
- Probador / Asegurador de la calidad.
- Administrador de bases de datos.

En la siguiente tabla se presentan los principales roles involucrados en el proceso de ingeniería de requisitos, así como las actividades en las que tienen mayor participación.

Tabla 5. Roles involucrados en la ingeniería de requisitos

Rol	Descripción
Cliente	Representa a la persona u organización que solicita la creación de un sistema a un área de desarrollo y quien lo paga. Es con quien se negocia el tiempo, costo y alcance del proyecto. Pueden o no ser usuarios del sistema.
Usuario	Son las personas que interactuarán con el sistema. Proporcionan información fundamental para el éxito del proyecto, ya que conocen y conviven con los procesos diarios.
Líder de proyecto	Por parte del equipo de desarrollo, es el representante ante el cliente. Es la persona responsable de completar el proyecto exitosamente con los recursos dados.
Analista	Su labor se enfoca en la ingeniería de requisitos, los identifica, analiza, modela y documenta. Además, establece contacto directo con los usuarios y utiliza diversas técnicas de comunicación y de recopilación de información para lograr su objetivo.
Programador	Con base en los requisitos recibidos de parte de los ingenieros de requisitos, el programador realiza la codificación para producir el sistema deseado.
Asegurador de la calidad	Garantiza el cumplimiento del proceso y de los estándares del producto. Enfocado en los requisitos, los verifica y valida para imprimir la calidad desde las primeras etapas del desarrollo. Paralelamente, prepara planes de prueba para esos requisitos del sistema.
Arquitecto	Es el responsable del diseño de alto nivel y es clave a la hora de precisar los atributos de calidad del producto.

Nota: tomado de Ventura (2002)

Algunas de las técnicas que se pueden emplear para llevar a cabo la labor de análisis de los stakeholders incluyen entrevistas con los expertos, lluvia de ideas en grupo y lista de chequeo. Se espera que este grupo de personas identifiquen y

caractericen a los stakeholders objetivamente, por tal motivo es recomendable involucrar a personas de diferentes contextos (Karisen, 2002 citado en Wessinger, 2012).

Matriz de stakeholders

La utilización de esta herramienta de análisis permite clasificar a los involucrados en el proyecto según sus niveles de interés y poder sobre él, lo que facilita la priorización de los stakeholders más importantes para desarrollar las estrategias de gestión correspondientes.

- **Importancia de la matriz de stakeholders en los proyectos de desarrollo**
  En los proyectos de desarrollo, la gestión de los stakeholders es de suma importancia para alcanzar el éxito de los proyectos, ya que el proceso de identificación de los involucrados y la definición de sus niveles de interés e influencia en el proyecto, marcarán el punto de partida para desarrollar estrategias que posibilitan obtener el apoyo requerido para alcanzar los objetivos por los que el proyecto es emprendido. Es por ello, que la matriz de stakeholders es una herramienta indispensable desde el comienzo del proyecto mismo, ya que proveerá de la información necesaria para gestionar, adecuadamente, las expectativas de los involucrados a lo largo del proyecto, maximizando las influencias positivas y mitigando los impactos negativos potenciales derivados de estos. Además, dado el carácter social de los proyectos de desarrollo, involucrar a la sociedad civil no debe ser solo un ejercicio de comunicación unidireccional, sino una oportunidad para lograr su apoyo al proyecto.

- **Proceso de armado de la matriz de stakeholders**

  Para desarrollar la matriz de stakeholders es necesario identificar las entradas necesarias que proveerán la información con la que el líder y el equipo de proyecto trabajarán para desarrollar la matriz misma. Tales entradas pueden ser el acta de constitución de proyecto, documentos de adquisición, activos de los procesos y factores ambientales de la organización.

  Entradas

  - Acta de constitución del proyecto.
  - Documentos de adquisición.
  - Factores ambientales de la organización.
  - Activos de los procesos de la organización.

  Herramientas y técnicas

  - Análisis de los interesados.
  - Juicio de expertos.

  Salidas

  - Matriz stakeholders (registro y estrategias de gestión).

Para profundizar en detalle, lo invitamos a consultar la Guía PMBOK 6 – 49 procesos, entradas, herramientas y salidas, que se encuentra en el material complementario.

Descripción de los componentes de la matriz de stakeholders

A continuación, se presenta el concepto de cada uno de los componentes que estructuran la matriz de stakeholders.

a) **Stakeholder**

Es el nombre con el que se identifica al stakeholder.

b) **Tipo**

Identifica si el stakeholder desempeña un rol interno o externo al proyecto mismo. Los stakeholder pueden ser internos, como el personal de las unidades ejecutoras, el personal administrativo o ejecutivo de la organización, el personal de las entidades financiadoras con alto nivel de poder e influencia en el proyecto y sus recursos; o externos como los beneficiarios del proyecto, las instituciones del sector o las organizaciones de la sociedad civil, quienes serán de un modo u otro, impactados por los resultados del proyecto.

c) **Objetivo o resultados**

En este campo se enlistan los objetivos o resultados en los que el stakeholder muestra interés o en aquellos en los que puede influir positiva o negativamente con sus acciones. Esta información puede ser suministrada por el acta de constitución de proyectos, la estructura de la organización, la estructura de desglose de trabajo, los diferentes planes que conforman el proyecto, entrevistas a los mismos interesados, etc.

d) **Acciones posibles con impacto positivo / negativo**

Son las acciones que puede emprender el stakeholder y que pueden influir, negativa o positivamente, en los objetivos del proyecto en los que muestra su interés o en aquellos en los que puede influir debido a su jerarquía, estatus, recursos de los que dispone, entre otros.

e) **Estrategias**

Es un listado de acciones que se pueden emprender para obtener el apoyo necesario o evitar obstáculos por parte de los stakeholders durante la ejecución y conclusión del proyecto. Las estrategias se desarrollan considerando el tipo de stakeholder, los objetivos en los que está interesado, el nivel de interés y poder que puede ejercer en el proyecto y las acciones posibles que podría emprender para afectar tanto positiva como negativamente al proyecto.

f) **Conclusiones**

Es la síntesis sobre puntos clave a considerar para gestionar de manera efectiva las expectativas de los stakeholders. Las conclusiones se obtienen de relacionar, analizar y sintetizar toda la información vertida en la matriz de stakeholders.

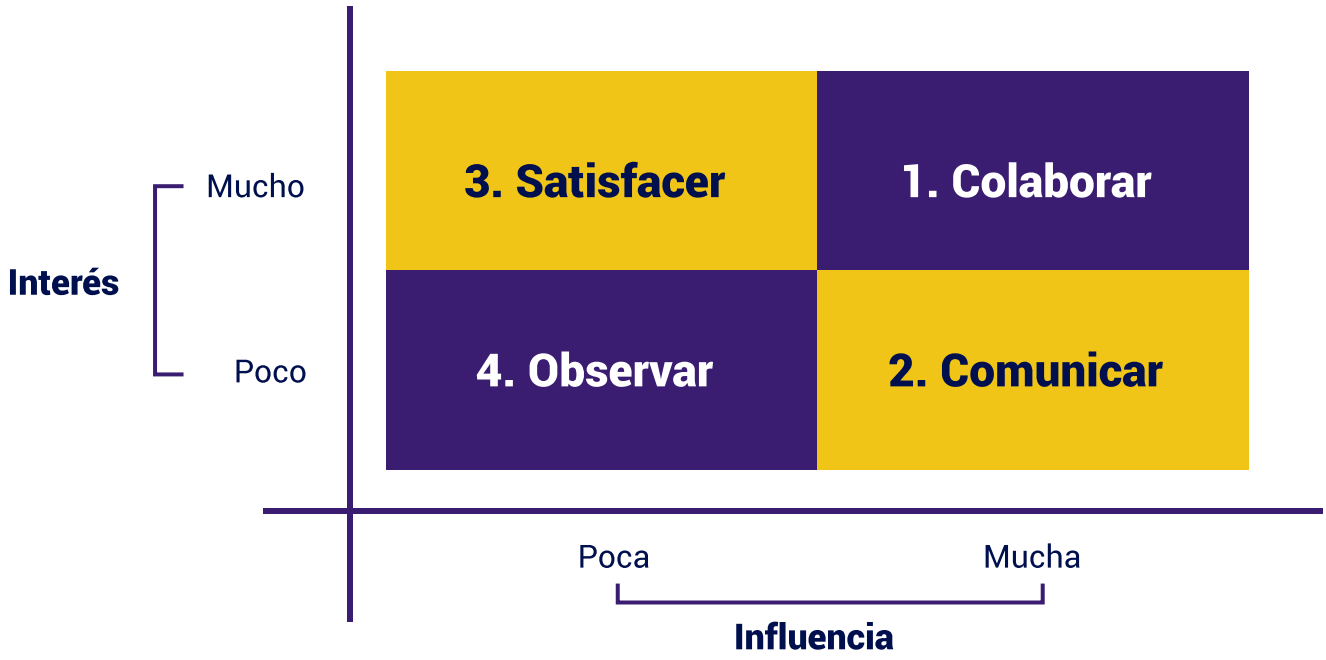
Categorización de stakeholders y estrategias de gestión de las expectativas:

Como ya se había mencionado anteriormente, la matriz de stakeholders es una herramienta muy útil que permite clasificar a los involucrados en el proyecto según sus niveles de interés e influencia, priorizando a los más importantes y desarrollando así las estrategias correspondientes para gestionar sus expectativas. De la misma manera, su clasificación puede cambiar durante la vida del proyecto. Así, aquellos que fueron inicialmente identificados con un alto nivel de influencia en el proyecto, pueden ser reclasificados a un nivel más bajo durante otras etapas de la vida del proyecto.

La categorización de los stakeholders se lleva a cabo una vez que la información sobre éstos esté completa. Para ello se puede utilizar una matriz de 2x2 en la que se pueda graficar el grado de poder e interés que tiene el involucrado en el proyecto,

coadyuvando así a clasificar a cada stakeholder dentro del grupo para el cual se definen diferentes estrategias.

Figura 12. Ejemplo de matriz interés vs. Influencia



3.2. Técnicas e instrumentos para elicitar requisitos

Hay una variedad de técnicas propuestas para ingeniería de requerimientos (Herrera, 2003. p. 12), por lo que es primordial resaltar que estas técnicas pueden ser aplicables a las distintas fases del proceso de la ingeniería de requerimientos (IR), teniendo en cuenta las características propias del proyecto en particular que se esté desarrollándose para aprovechar al máximo su utilidad.

A continuación, se presentará una serie de técnicas destinadas a facilitar la elicitación correcta y efectiva de los requisitos dentro de un proceso de desarrollo.

Entrevista

La entrevista es una forma de recoger información de otra persona a través de una comunicación interpersonal que se lleva a cabo por medio de una conversación estructurada. (Braude, 2003)

En las entrevistas se pueden identificar tres fases: preparación, realización y análisis (Piattini *et al*, 1996), como se observa a continuación.

a) Preparación

El entrevistador debe documentarse e investigar la situación de la organización, analizando los documentos de la empresa disponible.

- Se debe intentar minimizar el número de entrevistados, considerando las entrevistas de cortesía.
- Analizar el perfil de los entrevistados.
- Definir el objetivo y el contenido de la entrevista.
- Planificar el lugar y la hora en la que se va a desarrollar la entrevista (es conveniente realizarla en un lugar confortable).
- Algunos proponen enviar previamente el entrevistado un cuestionario y un pequeño documento de introducción al proyecto de desarrollo.

b) Realización

Dentro de la realización de las entrevistas se distinguen tres etapas, tal como se expone en Piattini et al. (1996):

- **Apertura:** presentarse e informar al entrevistado sobre la razón de la entrevista.

- **Desarrollo:** cumplir las reglas del protocolo, hay que llegar a un acuerdo sobre cómo se va a registrar la información obtenida.

  Durante esta fase se pueden emplear distintas técnicas:

  - **Preguntas abiertas:** también denominadas de libre contexto (Gause y Weinberg, 1989), estas preguntas no pueden responderse con un "sí" o un "no", permiten una mayor comunicación y evitan la sensación de interrogatorio. Por ejemplo, "¿Qué se hace para registrar un pedido?", "Dígame qué se debe hacer cuando un cliente pide una factura" o “¿Cómo se rellena un recibo?".
  - **Utilizar palabras apropiadas:** se deben evitar tecnicismos que no conozca el entrevistado y palabras o frases que puedan perturbar emocionalmente la comunicación (Goleman 1996).
  - **Mostrar interés en todo momento:** es fundamental cuidar la comunicación no verbal (Davis, 1985) durante la entrevista: tono de voz, movimiento, expresión facial.

- **Terminación:** se termina recapitulando la entrevista agradeciendo el esfuerzo y dejando abierta la posibilidad de volver a contactar para aclarar conceptos o bien citándole para otra entrevista.

c) Análisis

Consiste en leer las notas, pasarlas en limpio, reorganizar la información, contrastarlas con otras entrevistas o fuentes de información, evaluar cómo ha ido la entrevista.

En estas entrevistas, el equipo de la ingeniería de requerimientos hace preguntas sobre el sistema que utilizan y sobre el sistema a desarrollar. Los requerimientos provienen de las respuestas a estas preguntas.

Las entrevistas se pueden clasificar fundamentalmente, en:

- **Entrevista estructurada**

  Las preguntas en esta entrevista se deciden, previamente, de acuerdo con el detalle de información requerida.

  - Recoge de forma sistemática y precisa la mayor información sobre los aspectos que quiere explorar.
  - Las preguntas son prefijadas y definidas, las respuestas son esperadas e incluso se le dan al entrevistado en forma de varias opciones.
  - Las etapas son planificadas.
  - La interpretación de las respuestas se realiza de acuerdo con unos criterios establecidos.

    Lo invitamos a consultar el PDF "**Entrevista estructurada**", el cual se encuentra en la carpeta Anexos.

- **Entrevista semiestructurada**

  Esta presenta un grado mayor de flexibilidad que la estructurada, debido a que parten de preguntas planeadas, que pueden ajustarse a los entrevistados. Su ventaja es la posibilidad de adaptarse a los sujetos con enormes posibilidades para motivar al interlocutor, aclarar términos, identificar ambigüedades y reducir formalismos.

- Las preguntas, desarrollo e interpretación se planifican previamente, pero con un cierto grado de libertad de acción para abordar temas que pueden surgir durante la misma.
- Se suele utilizar un protocolo para facilitar al entrevistador seguir un modelo preestablecido.

- **Entrevista no estructurada**

  Las entrevistas no estructuradas suelen describirse como conversaciones mantenidas con un propósito en mente.

  - No se estructura ni planifica previamente.
  - Es la más ágil y la que proporciona más información en general, pero requiere un cierto dominio por parte del entrevistador.

  En el material complementario se pueden revisar ejemplos de entrevistas.

Encuesta

Los cuestionarios son herramientas ampliamente utilizadas para recoger datos de sondeos y pueden ser administradas sin la presencia del investigador. (Cohen, 2011, p. 377).

Pueden variar en cuanto a propósito, diseño y apariencia, y consisten en listas de preguntas escritas. Los individuos participantes en la investigación suelen leer los mismos listados de preguntas, por lo que esto permite consistencia y precisión al analizar las respuestas, además de facilitar el proceso. Una de las ventajas más destacadas de los cuestionarios es que simplifican el proceso de la obtención de datos, preguntando directamente a los individuos participantes para obtener datos de forma rápida y directa y se pueden aplicar a un gran número de sujetos.

Los datos que se obtienen a través de los cuestionarios suelen estar clasificados en dos categorías: hechos y opiniones (Denscombe, 2010, p. 156). La información relacionada con los hechos no requiere el juicio o la actitud personal de los sujetos participantes, pero la información obtenida a través de las opiniones implica creencias, puntos de vista y preferencias de los sujetos participantes.

- **Tipos de preguntas**

  La distinción más general entre los tipos de preguntas de los cuestionarios, además de hechos y opiniones, es la de preguntas abiertas y cerradas; las preguntas abiertas son aquellas en las que no se especifica ninguna respuesta para elegir y se deja abierta a la elección del participante para que escriba en ella. Las preguntas cerradas son las que ofrecen ya unas respuestas predeterminadas para su elección.

- **Tipos de respuestas**

  Las respuestas de escala son las más comunes en los cuestionarios de investigación, ya que implican al participante en una valoración o evaluación de las respuestas objetivo por medio de varias opciones en las que tienen que marcar dentro de una escala la importancia de cada una. Esa escala de valoración indica diferentes grados en una categoría y puede ser de diversa naturaleza; por ejemplo, puede valorar una categoría indicando si es “bueno” o “malo”, “frecuente” o “infrecuente”, “importante” o “poco importante” o también pueden valorar opiniones: “completamente de acuerdo” o “en desacuerdo”. El número de opciones más común es el de cinco, por ser un número impar, ya que existe una tendencia generalizada a seleccionar la opción intermedia (Dornyei, 2010).

Para ampliar la información, lo invitamos a explorar el video Tipos de preguntas en una encuesta, el cual se encuentra en el material complementario.

Observación

Esta permite la obtención de datos para emprender una investigación de tipo cualitativo, no desde el punto de vista de lo que los sujetos dicen, sino que es la evidencia directa de lo que ve y percibe el investigador en un escenario de primera mano (Denscombe, 2010).

Por su parte, Selltiz (citado por Hernández, Fernández y Baptista, 2006, p. 229), al referirse a la observación, recomienda que para que esta se convierta en una técnica como tal, debe cumplir con cuatro condiciones:

- Debe servir a un objeto formulado de investigación.
- Debe de ser planificada sistemáticamente.
- Debe estar controlada y relacionada con proposiciones generales.
- Debe ser sujeta a comprobaciones y controles de validez y fiabilidad.

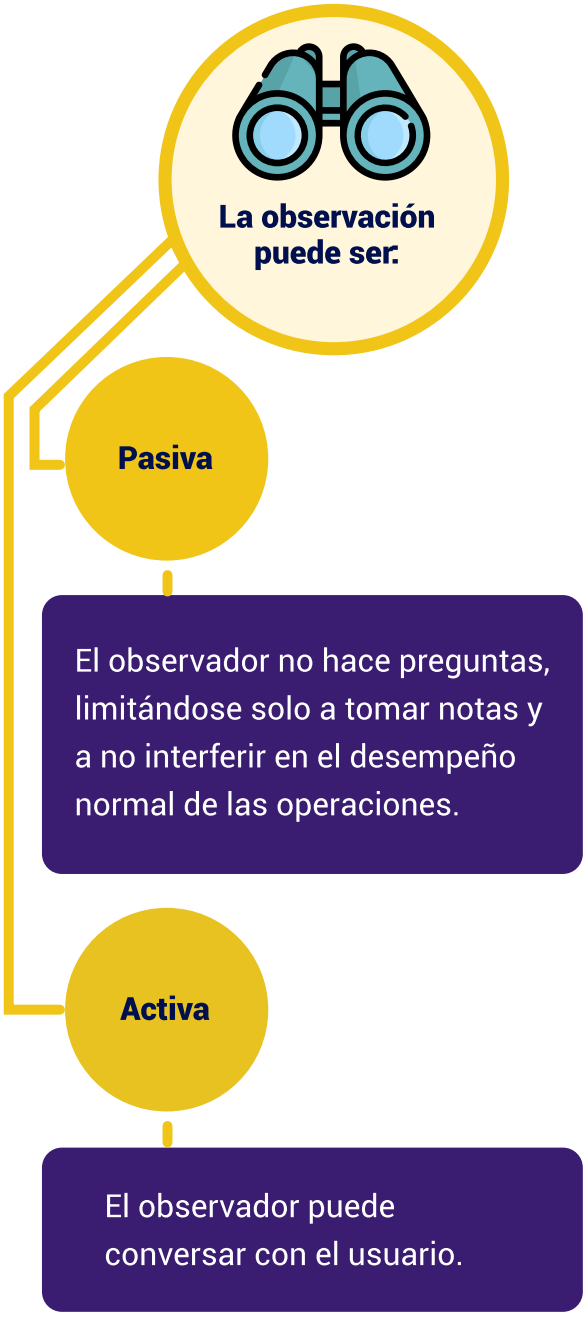
De acuerdo con lo anterior, se puede asumir que la observación:

- Tiene la característica de seguir normas, reglas y procedimientos.

Para el caso de obtención de requerimientos del software la observación nos sirve para estudiar el entorno de trabajo de los usuarios, clientes e interesados de proyecto (stakeholders) y para documentar la situación actual de procesos de negocio.

En la siguiente figura, se pueden revisar los tipos de observación.

Figura 13. Tipos de observación



Ahora bien, para llevar a cabo la observación, el observador puede utilizar como instrumento la guía de observación, la cual le permite situarse de manera sistemática en aquello que realmente es objeto de estudio para la investigación; también es el

medio que conduce la recolección y obtención de datos e información de un hecho o fenómeno.

Tamayo (2004) define a la guía de observación como:

Un formato en el cual se pueden recolectar los datos en sistemática y se pueden registrar en forma uniforme, su utilidad consiste en ofrecer una revisión clara y objetiva de los hechos, agrupa los datos según necesidades específicas, se hace respondiendo a la estructura de las variables o elementos del problema (p. 172).

Para elaborar la guía de observación se ha de diseñar el contenido de la observación; el cual debe incluir por lo menos los siguientes aspectos:

- Datos y características de los sujetos a evaluar.
- Propósitos de la observación o de las observaciones a realizar.
- Temporalidad de la observación.

Lo invitamos a consultar el PDF "**Ficha de observación**" el cual se encuentra en la carpeta Anexos.

Sesiones grupales

Es un proceso por el cual se llevan a cabo reuniones en grupo altamente estructuradas que convocan, en una misma sala, a los usuarios de un sistema, los propietarios del sistema y a los analistas durante un amplio periodo de tiempo. Los objetivos de esta técnica son esencialmente los mismos que los de las entrevistas, con la salvedad de necesitar más analistas para llevarlos a cabo.

Dentro de las sesiones de trabajo en grupo se encuentran técnicas como la lluvia de ideas, las sesiones JAD y el método Delphi.

- **Lluvia de ideas**

  También denominada tormenta de ideas o incluso brainstorming. Faickney (1939) investigó sobre diferentes maneras de generar creatividad. Se percató de que la mejor manera de ser creativo en una empresa es a través de la interacción y el trabajo en equipo; todos juntos podían dar sus opiniones y sugerencias sobre un tema determinado. Creó de esta manera la lluvia de ideas.

- **Sesiones JAD (**Joint Application Design**)**

  Es un proceso usado para reunir requerimientos en el desarrollo de nuevos sistemas de información para una compañía. El proceso JAD consiste en un taller donde los trabajadores del conocimiento y los especialistas en tecnologías de información se reúnen, algunas veces durante varios días, para definir y revisar los requerimientos de negocio para el sistema. Los asistentes incluyen oficiales de administración de alto nivel, quienes se aseguran de que el producto provea los reportes y la información requerida al final, esto actúa como “un proceso de administración” que permite que los departamentos de servicios de información corporativa trabajen más eficientemente con los usuarios en un marco de tiempo más reducido.

  A través de los talleres JAD, los trabajadores del conocimiento y los especialistas en tecnologías de información pueden resolver cualquier dificultad o diferencias entre las posturas referentes al nuevo sistema de información. El taller sigue una detallada agenda para lograr garantizar que todas las incertidumbres entre los grupos sean cubiertas y para ayudar a prevenir cualquier falla en la comunicación, estas fallas de comunicación

pueden provocar repercusiones mucho más serias si no se identifican a tiempo. Al final, este proceso resultará en un nuevo Sistema de Información viable y orientado tanto a diseñadores como a usuarios.

- **Método Delphi**

  Es un método de estructuración de un proceso de comunicación grupal que consiste en la selección de un grupo de expertos a los que se les pregunta su opinión frente a ciertas temáticas.

  - **Fase uno.** Formulación del problema: se define el campo de investigación.
  - **Fase dos.** Elección de expertos: el experto se elige según su preparación y su capacidad de proyección.
  - **Fase tres.** Elaboración de cuestionarios: las preguntas deben hacerse de acuerdo con la temática que se quiere obtener.
  - **Fase cuatro.** Desarrollo y explotación de resultados: el cuestionario se entrega a los expertos para ser contestado por ellos.

3.3. Herramientas para captura de requisitos

Existen varias herramientas para la captura de requisitos potenciales de un nuevo sistema o una actualización de software, a continuación, se explican las más utilizadas:

Diagrama de casos de uso

Al momento de desarrollar un proyecto se debe pensar en cuáles serán las principales funcionalidades que el software debe permitir llevar a cabo y quiénes serán los que podrán ejecutar dichas funcionalidades. La identificación de estos elementos se puede visualizar de manera efectiva a través de la elaboración de

diagramas de casos de uso; estos diagramas, que son elaborados durante las etapas iniciales de un proyecto, se convierten en referente para cada una de las etapas siguientes del desarrollo del proyecto.

Componentes. En los diagramas de casos de uso, se observan los siguientes componentes.

- **Actor**

  Se representa mediante un “hombre de palo”. Este se emplea para indicar el tipo de usuario del sistema que podrá ejecutar alguna función.

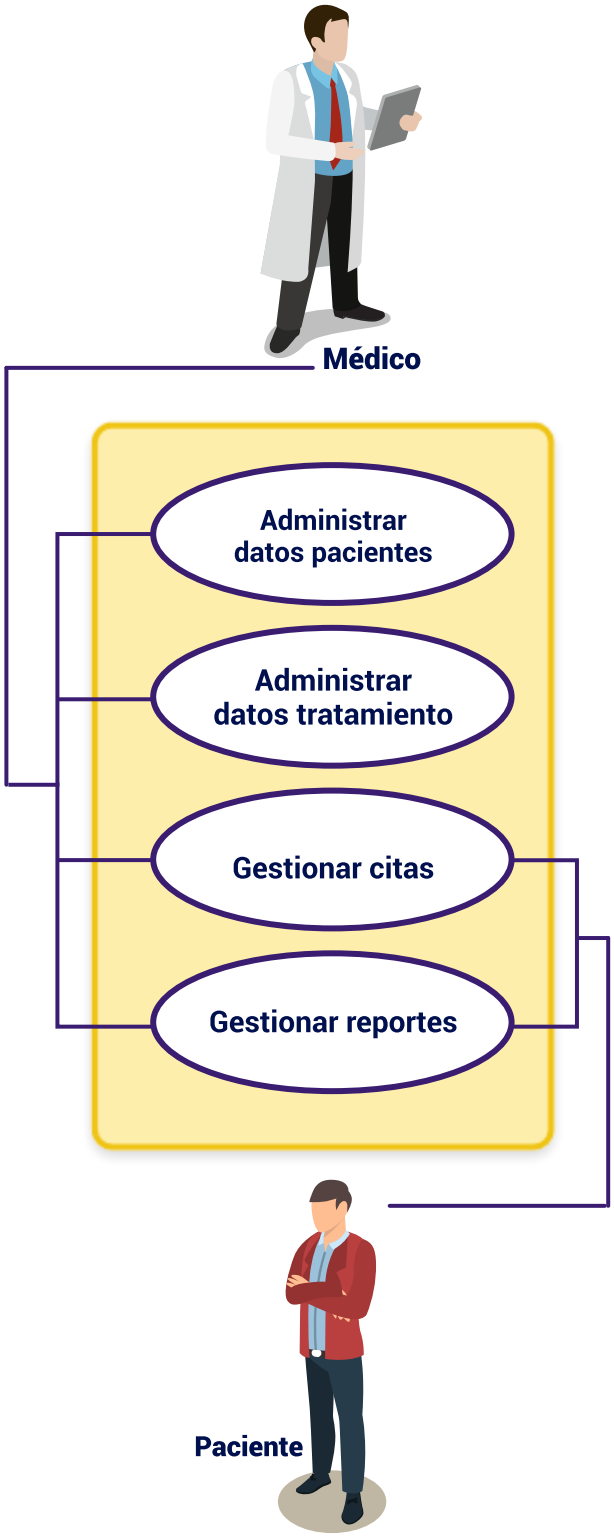
- **Caso de uso**

  Se representa mediante un óvalo e indica una función que el sistema debe proveer.

Para ejemplificar un proceso se puede emplear un verbo conjugado en infinitivo y que represente la función a realizar (administrar, gestionar, registrar, entre otros). A continuación, se presenta un ejemplo, en el cual se presenta un diagrama de casos de uso de la sistematización de un centro médico.

- Administrar datos pacientes.
- Administrar datos tratamientos.
- Gestionar citas.
- Generar reportes.

Figura 14. Caso de uso centro médico



- **Identificación de casos de uso:**

  En el ejemplo anterior se observan los casos de uso identificados en el sistema, es decir, las funcionalidades que el sistema va a proveer (administrar datos pacientes, administrar datos tratamientos, gestionar citas, generar reportes).

- **Identificación de actores:**

  Los actores son los usuarios que podrán ejecutar los casos de uso, en el ejemplo anterior, se identificaron dos actores (médico y empleado).

- **Documentación:**

  La técnica de casos de uso requiere además de construir el diagrama de casos de uso, la descripción de estos. Esta descripción permite detallar el flujo de eventos que se da entre el Sistema y el Actor para llevar a cabo el caso de uso.

Documentación caso de uso

Lo invitamos a consultar el PDF "**Caso de uso**", el cual se encuentra en la carpeta Anexos, y corresponde a un formato diligenciado, de acuerdo con el ejemplo del centro médico.

Historias de usuario

Las historias de usuario son utilizadas en los métodos ágiles para la especificación de requisitos, son una descripción breve de una funcionalidad software tal y como la percibe el usuario (Cohn, 2004).

El formato para las historias de usuario Scrum se basan en una regla de tres palabras:

- **Como** <rol>
- **Quiero** <eventos>
- **Para** <funcionalidades>

Así, el <rol> que se escoja que va a utilizar la aplicación software, requiere de una <Acción> / <evento> que ocurra, porque se desea cubrir una <funcionalidad>. Corto y conciso, directo y claro.

A continuación, se presentan ejemplos de historias de usuario.

- Como un cliente, quiero consultar para poder encontrar el producto que deseo comprar.
- Como un cliente, quiero consultar para poder encontrar el producto que deseo comprar.

  Como un cliente, quiero que los productos seleccionados para la compra queden almacenados en un carrito de compra para poder visualizar todos mis productos y el precio total.

Conversación para explicar mejor la historia de usuario

Como se mencionó anteriormente, las historias de usuario son una frase sencilla y concisa, sin embargo, eso no impide que se pueda abrir un diálogo (conversación) entre todos los miembros del equipo. De hecho, esta conversación se debe llevar a cabo para explicar mejor la propia historia de usuario y conseguir objetivos como:

- Detallar a mayor nivel como se realizará la solución.
- Clarificar aspectos de valor, funcionamiento y técnicos.
- Resolver las dudas que aparezcan.

Estas conversaciones llevarán a alcanzar acuerdos sobre los distintos puntos tratados, que quedarán reflejados en los criterios de aceptación y que permitirán validar cuando una historia de usuario está terminada.

Confirmación de los criterios de aceptación

Los criterios de aceptación, es decir, la confirmación. Se trata de criterios claros y específicos que todo el equipo debe comprender y que permitirán avaluar en el futuro si la implementación que se está desarrollando o las pruebas que se realicen están terminadas.

Storyboard

¿Qué es un Storyboard?

Los storyboards son un tipo de prototipos muy utilizados, consiste básicamente en ir mostrando en una secuencia de imágenes un proceso, acción o ejercicio que se puede realizar en el sistema una vez terminado, las imágenes van mostrando la evolución del sistema conforme el usuario interactúa con el sistema.

Con esta técnica se pretende crear diferentes vistas del sistema en las primeras etapas de su implementación de la manera más rápida y barata posible [SUT02].

Una forma muy común de ejemplificar los storyboards es con las revistas de cómics, ya que van mostrando una secuencia de imágenes en cuadros con un orden establecido que permiten entender la línea de la historia contada. La técnica storyboard permite generar modelos o esquemas visuales como esbozos de interfaces gráficas de usuario (GUI).

A continuación, se detallan las principales características de los storyboards:

- Se preserva el punto de vista del proceso del negocio.
- Se puede validar un escenario.
- Se pueden validar escenarios integradores logrando una visión global.
- Son más fáciles de comprender por el usuario.
- No genera falsas expectativas.
- El usuario sigue trabajando con herramientas conocidas.
- Son fáciles de mantener o adaptar a los cambios.
- Permiten incorporar modificaciones durante la validación.

3.4. Herramientas de modelado

Las herramientas de modelado de sistemas informáticos se emplean para la creación de modelos de sistemas que ya existen o que se desarrollarán; estas herramientas permiten crear un "simulacro" del sistema, a bajo coste y riesgo mínimo. A bajo costo porque, es un conjunto de gráficos y textos que representan el sistema.

El Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; UML es un lenguaje gráfico que permite especificar, modelar, construir y documentar los elementos que forman un sistema software.

De otra parte, las herramientas CASE son un conjunto de programas y procesos “guiados”, que ayudan a los analistas, desarrolladores, ingenieros de software y diseñadores en una o todas las etapas que comprende un ciclo de vida, con el objetivo de facilitar el desarrollo de software. El objetivo general de estas herramientas es acelerar el proceso para el que han sido diseñadas, es decir, para automatizar o apoyar

una o más fases del ciclo de vida del desarrollo de sistemas. CASE proporciona un conjunto de herramientas semiautomatizadas y automatizadas que están creando una nueva cultura de ingeniería en muchas empresas. Las herramientas CASE se diseñaron para aumentar la productividad en el desarrollo de software y reducir su costo.

Existen muchas herramientas para modelado tanto en libres como con derechos comerciales; a renglón seguido se listan las herramientas CASE más utilizadas:

- ER win
- ArgoUML
- Easy Case
- Oracle Designer
- Power Designer
- System Architect
- SNAP
- Gliffy
- MagicDraw
- Lucidchart
- Papyrus Uml
- Modelio
- StarUml
- Dia
- Mono Uml

A continuación, se realizará una descripción del top 5 de las más utilizadas.

- **Gliffy**

La aplicación en línea Gliffy es una herramienta de diagramas UML basada en la nube. Apareció por primera vez en 2006 y se trata de una herramienta de modelado que crea todo tipo de diagramas, tales como diagramas de flujo, diagramas de Venn y, por supuesto, diagramas UML. La herramienta en línea fue escrita en HTML5 y es bastante popular gracias a su rapidez de reacción. Es de anotar que antes de que Gliffy pasara por la fase beta en 2007, la empresa homónima cooperó con el grupo de software australiano Atlassian. Ya en 2006, su software de colaboración Confluence integró un plugin de Gliffy y, más tarde, el equipo de Gliffy desarrolló un plugin para Jira. Google Workspace y Drive de Google también integran esta herramienta UML.

- **ArgoUML**

  Ha sido durante mucho tiempo una de las herramientas UML gratuitas de código abierto más populares para el escritorio y aunque ya no se mantiene, muchos modeladores continúan usando el programa para tareas más pequeñas. Su software es multiplataforma, cuyo el requisito mínimo es Java 5 ArgoUML soporta todos los tipos de diagramas de la versión 1.4 de UML y perfiles UML. El programa también ofrece algunas formas decorativas que no forman parte del estándar UML.

  Además, aunque esta herramienta UML está disponible como descarga gratuita, ArgoUML soporta una amplia gama de lenguajes de programación cuyo código puede generarse a partir de un diagrama. La ingeniería inversa también es posible para Java, C++, PHP, C# y SQL. El programa reconoce otros idiomas como Delphi o Ruby cuando los agrega como extensiones a la carpeta de archivos ArgoULM.

- **MagicDraw**

  Esta aplicación de escritorio destaca por su diseño moderno y claro, así como por su variedad de funciones y la facilidad de su uso. Esta herramienta de diagramas UML ofrece además SysML, representación gráfica de procesos de negocio con BPMN (Business Process Model and Notation) y el marco de arquitectura UPDM (United Profile for DoDAF/MODAF).

  MagicDraw también ofrece lenguaje de especificación OCL (Object Constraint Language), y XMI, que puede usar para exportar diagramas a otros programas sin pérdidas de información.

- **StarUML**

  Es una herramienta para el modelamiento de software basado en los estándares UML (Unified Modeling Language) y MDA (Model Driven Arquitecture).

  Da soporte completo al diseño UML mediante el uso de:

  - Diagrama de casos de uso.
  - Diagrama de clase.
  - Diagrama de secuencia.
  - Diagrama de colaboración.
  - Diagrama de estados.
  - Diagrama de actividad.
  - Diagrama de componentes.
  - Diagrama de despliegue.
  - Diagrama de composición estructural (UML 2.0).

- **Lucidchart**

  Herramienta para hacer diagramas UML online que permite comprender las complejidades en el código de forma más rápida y sencilla, pues automatiza el proceso de generación de un diagrama de clases.
  Simplemente elabora y personaliza los diagramas de secuencia en línea a partir del texto. Al ingresar el marcado en el diálogo emergente, Lucidchart generará automáticamente un diagrama de secuencia que cumple el estándar de PlantUML.

4. Técnicas de análisis y especificación de requisitos

A continuación, abordaremos el análisis de requisitos (priorización, descomposición funcional, matriz de trazabilidad) y estándares, y/o guías existentes para la especificación formal de los mismos dependiendo del tipo de marco de trabajo usado (tradición o ágil).

4.1. Técnicas de análisis de requisitos

El proceso de análisis de requisitos permite, principalmente, el estudio de las necesidades de los usuarios para definir requisitos del sistema por medio de la producción del documento de especificación de requisitos donde se describe lo que el sistema debe hacer, pero no el cómo. Así es como este proceso, además de involucrar un proceso de análisis, también requiere de la síntesis de la información existente.

A continuación, se describen algunas técnicas que pueden ser utilizadas para entender y ordenar los requisitos identificados para su posterior redacción en un artefacto formal dependiendo del marco de desarrollo de software utilizado.

Priorización de requisitos

Esta es una actividad clave para el éxito en la construcción de producto de software y su objetivo es maximizar el valor entregado por el proyecto a sus clientes; es decir, identificar cuáles requisitos se deben priorizar tomando en cuenta factores como la complejidad, las dependencias y el retorno de inversión a los clientes, entre otros indicadores.

La priorización de requisitos, por lo general, es responsabilidad de los gerentes del proyecto o dueños de producto dependiendo del tipo de enfoque utilizado; sin embargo, se requiere de la participación activa de los analistas de requisitos, ya que si

bien es el cliente o su representante quien determina cuáles requisitos son más importantes para su negocio, el analista debe asesorar, facilitar e informarle al cliente para no darle prioridad superior a requisitos que tienen otras dependencias no resueltas; por ejemplo, no se podría priorizar un requerimiento asociado a un carrito de compras sin antes tener resuelto los requerimientos asociados al registro y consulta de productos.

El proceso de priorización es continuo, es decir, la prioridad de un requisito puede variar a lo largo del proyecto, así como los requisitos también pueden variar a lo largo del mismo. La priorización no es un proceso sencillo, por lo que, además de conocer las diferentes técnicas, es importante la toma de decisiones y estas, en el marco de un proyecto, siempre van a generar inconformidades, pero si no se decide respecto a la priorización de requisitos es como si no se fundamentaran las bases sobre las que el equipo de desarrollo va a construir el sistema.

Técnica de clasificación de lista

Esta técnica de clasificación no requiere ningún tipo de entrenamiento o preparación, ya que es una forma de priorización natural que se usa en la vida cotidiana. Este tipo de priorización simple es una de las más utilizadas y consiste en darle un valor numérico a cada requerimiento iniciando por el número 1 y continuando de forma sucesiva con 2, 3 y hasta el número total de requisitos definidos (Porfirio, 2021).

Esta técnica tiene la ventaja de que solo puede existir un número 1, lo cual evita muchos problemas por parte de los interesados o responsables del negocio que quieren que todos los requerimientos tengan prioridad 1; adicionalmente, aporta

claridad y evita confusiones. Cada elemento se prioriza con relación al resto de elementos, lo que simplifica el proceso.

Para usar adecuadamente esta técnica, se requiere de un profundo conocimiento de todos los requerimientos definidos y si bien el proceso parece simple, se requiere de un gran esfuerzo por parte del equipo para situar a cada requerimiento en la posición correcta.

Técnica de puntos de historia y valor del negocio

Esta consiste en realizar el proceso de priorización de acuerdo con factores como el esfuerzo y la opinión de los gerentes de proyecto o dueños de producto, los clientes, el equipo de desarrollo o incluso una combinación de los anteriores. El valor del negocio corresponde a un valor numérico que cuanto más alto sea, más valor representa para el cliente (Porfirio, 2021).

Usar solo el valor del negocio como elemento de priorización puede generar problemas, ya que el valor asociado por el cliente a un requerimiento puede ser muy superfluo y, además, no considera detalles clave como el esfuerzo que se requiere para su desarrollo; por esta razón, también se usa el valor de puntos de historia, que no es más que otro valor numérico asignado por el equipo de desarrolladores a cada requerimiento, en el que se expresa una estimación de esfuerzo. Cuanto más grande sea el número, implica más esfuerzo requerido con miras a realizar el requerimiento.

Para lograr la priorización de los requerimientos, se debe realizar el cálculo del cociente obtenido a partir de los puntos de valor del negocio dividido entre los puntos de historia, quedando entonces una priorización donde estarán en los primeros lugares

los requerimientos más sencillos de resolver por los desarrolladores y que tengan mayor interés por parte del cliente. A continuación, se da un ejemplo.

Tabla 6. Ejemplo de aplicación técnica de puntos de historia y valor del negocio

Requerimientos	Valor del negocio	Puntos de historia	Cociente
R01	6	3	2
R02	7	1	7
R03	9	5	1,8
R04	3	3	1
R05	4	2	2
R06	5	4	1,25
R07	2	8	0,25

Si se trata de un proyecto de siete (7) requerimientos con los valores de negocio, puntos de historia y cocientes como se describe en la tabla anterior, al realizar el proceso de priorización quedaría en el primer lugar el requerimiento R02 ya que tiene el cociente más alto; es decir, representa un requerimiento fácil de construir para el grupo de desarrolladores y, adicionalmente, tiene un alto valor de negocio para el cliente; luego, en el segundo lugar, el requerimiento R01 y R05 y así sucesivamente se registran las prioridades de acuerdo con el valor del cociente de mayor a menor.

Técnica urgente

Aquí se utiliza una tabla de dos dimensiones, donde la horizontal estará determinada por el valor de la urgencia en el requerimiento, el cual corresponde a un valor numérico entre 1 y 5, donde un valor de 5 implica la mayor urgencia y 1 que no

hay tanto apuro en el desarrollo de requerimiento; y la dimensión vertical estará determinada por el valor del negocio, solo que esta vez, a diferencia de la técnica anterior, el valor del negocio también se rige por una escala de 1 a 5, siendo 5 el de mayor valor de negocio posible para un requerimiento (Porfirio, 2021).

Para determinar la prioridad final de un requerimiento, se utiliza una escala de colores que surge a partir de la multiplicación de los valores de las escalas de urgencia y de valor de negocio según la siguiente tabla:

Tabla 7. Referencia para la técnica urgente

Valor del negocio	Urgencia	Urgencia	Urgencia	Urgencia	Urgencia	Urgencia
Valor del negocio	5	5	10	15	20	25
Valor del negocio	4	4	8	12	16	20
Valor del negocio	3	3	6	9	12	15
Valor del negocio	2	2	4	6	8	10
Valor del negocio	1	1	2	3	4	5
Valor del negocio	1	2	3	4	5	

Nota: Tomado de Porfirio (2021).

Luego, se consideran los requerimientos de mayor prioridad que están en el sector de color rojo, después los de color naranja, seguido de los de color amarillo y, por último, los requerimientos del sector de color verde. Para entender mejor este estilo de priorización, observar el siguiente ejemplo:

Tabla 8. Ejemplo de aplicación técnica urgente

Requerimientos	Valor del negocio	Urgencia	Sector
R01	1	4	Verde
R02	2	4	Amarillo
R03	5	5	Rojo
R04	4	3	Amarillo
R05	5	4	Naranja
R06	3	1	Verde
R07	1	3	Verde

Al realizar la multiplicación de los valores de negocio y el valor de la urgencia, se puede establecer en qué sector se encuentra cada requerimiento. Ahora bien, tomando en cuenta los valores del ejemplo de la tabla anterior, se puede concluir que el primer requerimiento a abordar sería el R03 que está en el sector de color rojo, luego el requerimiento R05, que está en el sector de color naranja, y así sucesivamente.

Técnica MoSCoW

Esta técnica se basa en la asignación de etiquetas a cada requerimiento, y las disponibles se relacionan a continuación:

- M: indica una funcionalidad que debe estar (Must).
- S: indica una funcionalidad que debería estar (Should).
- C: indica una funcionalidad que podría estar (Could).
- W: indica una funcionalidad que no estará por ahora, de pronto más adelante (Wont).

Los requerimientos son priorizados utilizando el siguiente orden: primero los que tienen etiqueta M, luego los requerimientos con etiqueta S, después aquellos con C y, finalmente, los etiquetados como W.

Esta técnica requiere de un proceso de consenso sobre el significado de cada una de las etiquetas asignables a cada requerimiento. Los requisitos de tipo M son aquellos obligatorios y que, de no ser abordados, implicaría directamente el fracaso; es importante entonces acordar qué se puede entregar y que sea útil, adicionalmente deben formar parte de un conjunto coherente, ya que si, sencillamente se seleccionan todos los requerimientos de tipo M, automáticamente todos se transforman en requerimientos de tipo M y se pierde la dinámica de la técnica.

Normalmente el proceso de desarrollo de software es iterativo e incremental por lo que dependiendo del momento en que se encuentre un requerimiento, que ahora puede ser W, en la siguiente iteración puede asumir un valor de M.

Juicio de expertos

Es una técnica basada principalmente en función de la complejidad y la exactitud en los resultados, básicamente consiste en realizar el proceso de priorización utilizando como base la opinión del gerente de proyecto o dueño de producto, o de algún stakeholder con conocimientos en la industria asociada al producto a desarrollar (Porfirio, 2021).

Este tipo de técnica funciona muy bien para proyectos pequeños con un dueño de negocio que tiene mucho conocimiento del problema y de la solución que necesita para resolverlo; sin embargo, se debe utilizar esta técnica con cuidado ya que se basa en una visión sesgada del negocio.

Matriz de priorización

Esta técnica consiste en construir una tabla donde cada requerimiento es valorado en una escala de 0 a 10 o de 0 a 100 para cada dimensión alineada con los objetivos del producto. Normalmente cada dimensión tiene un peso porcentual, de modo que cada requerimiento tendrá un valor final a partir de la sumatoria de la multiplicación de cada puntuación por el peso de cada dimensión (Porfirio, 2021).

Por ejemplo, se debe considerar la siguiente tabla:

Tabla 9. Ejemplo de aplicación técnica de matriz de priorización

Requerimiento	Criterio: Conversión 30%	Criterio: Satisfacción de usuario 40%	Criterio: Retención 30%	Resultado: 100%	Ranking
R01	4	3	8	4,8	2
R02	6	4	3	4,3	7
R03	8	7	6	7	1,8
R04	4	2	4	3,2	1

Suponiendo las dimensiones de conversión con un peso porcentual del 30%, la satisfacción del cliente con un peso porcentual del 40% y la retención de clientes con peso porcentual del 30%, cada uno de los requerimientos (R01-R04) son evaluados en una escala de 0 a 10, a los cuales se les aplica el cálculo de acuerdo con el valor porcentual de cada dimensión evaluada y el resultado se obtendría de la sumatoria de los valores parciales de cada dimensión, con lo cual los mayores valores totales obtenidos serán priorizados sobre los de menor valor.

Esta técnica sigue siendo muy subjetiva al igual que la técnica de juicio de expertos, aunque quedan claros cuáles son los criterios de evaluación utilizados para determinar la prioridad de los requisitos.

Matriz de trazabilidad

La matriz de trazabilidad es una herramienta que permite alinear los requisitos del proyecto con los logros de los objetivos, es una tabla que relaciona cada uno de los requerimientos con el entregable solicitado. Es decir, permite identificar qué resultado se alcanza con cada requisito y, a la vez, permite visualizar qué requisitos son necesario cumplir para determinado entregable (Pantaleo, 2018).

Este es un instrumento clave para el responsable del proyecto, en especial en el proceso de seguimiento y control de cambios, ya que permite analizar qué requerimientos, eventualmente, convendría modificar, eliminar o añadir; así, se puede ajustar la planificación de tareas pendientes del proyecto. También permite identificar inconsistencias entre los requerimientos y los beneficios que se esperan alcanzar, tener una visualización rápida de los requisitos que han sido abordados y cuáles están pendientes por realizar y, por lo tanto, qué entregables están próximos a producirse y cuáles todavía requieren de mayor tiempo.

La estructura de la matriz de trazabilidad se puede construir en una hoja de cálculo, en ella se relacionan todos los requisitos y las metas a alcanzar junto con una serie de valores complementarios que aportan información y coherencia a los vínculos generados.

Cada organización es responsable de adaptar la matriz de correlación a sus necesidades particulares; en el siguiente recurso se propone el contenido base de una matriz de correlación dividida por secciones.

- **Identificación**

  Identificador: código único para cada requisito.

  Código jerárquico: código que permite categorizar cada grupo de requisitos.

  Descripción: texto explicativo del requerimiento.

  Tipo: categoría del requisito (requerimiento de negocio, requerimiento de los interesados, funcionales y no funcionales, requerimientos del proyecto, requerimiento de calidad).

- **Estado**

  Versión: versión del requerimiento (importante para el control de cambios).

  Estado: activo, cancelado, diferido, agregado, aprobado, asignado o completado.

  Fecha de estado: momento en el que se realizó la última modificación del estado.

  Responsable: persona encargada del requisito.

  Prioridad: escalas de prioridad usada en el proyecto (alta, media, baja; 1, 2, 3, 4, 5).

  Otras características: cualquier otro campo de valor para la organización.

- **Objetivo**

  Objetivo: objetivo que pretende alcanzar el requerimiento.

Necesidad de negocio: necesidad de negocio que se pretende cubrir con el objetivo del proyecto y por ende con el requisito.

Entregable: producto.

Ejemplo de la estructura de una matriz de trazabilidad

Al construir una matriz de trazabilidad se deben usar los campos que se consideren útiles para el proyecto, pues no todos los proyectos son iguales y la estructura definida para uno puede no resultar conveniente para otro proyecto. Cuando se usa esta matriz debe permanecer actualizada a lo largo del ciclo de vida de construcción del proyecto.

Lo invitamos a consultar el documento "**Ejemplo estructura matriz trazabilidad**", el cual se encuentra en la carpeta Anexos, como tabla base para una estructuración de trazabilidad.

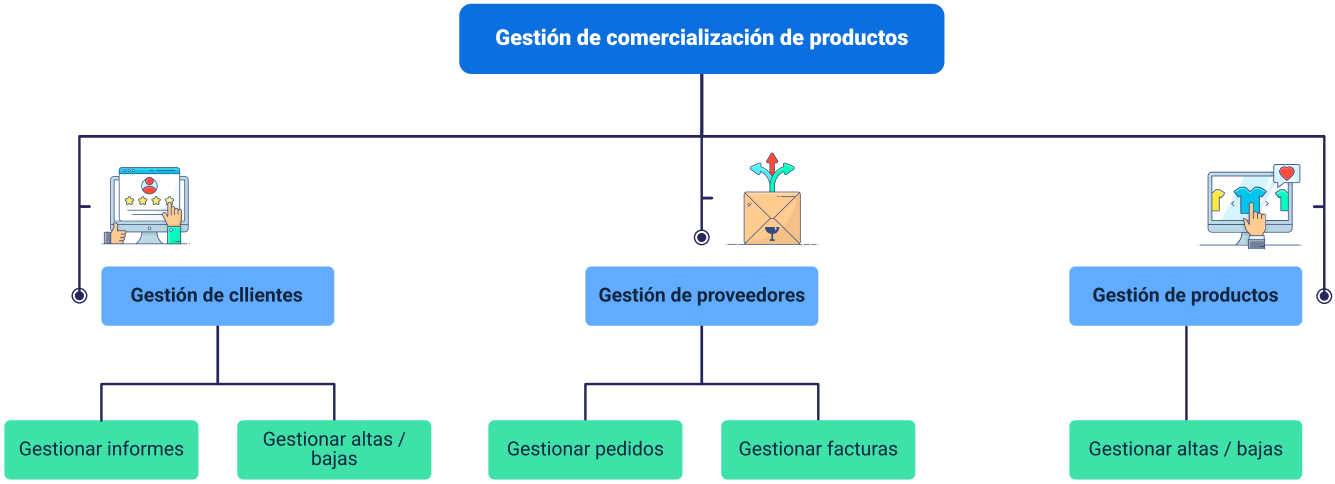
Descomposición funcional

Esta estrategia consiste en definir los requerimientos como una relación de entradas y salidas de un sistema. Normalmente, estos requerimientos se definen de manera muy general y poco detallada y luego empieza a descomponerse en funcionalidades y subfuncionalidades un poco más detallados con el ánimo de analizarlas individualmente hasta lograr un nivel de detalle adecuado para el proyecto. A esta estructura se le conoce como top-down; el resultado es una estructura jerárquica (Pantaleo, 2018).

La descomposición funcional se realiza, por lo general, para identificar y entender los componentes o partes que constituyen un todo; en este proceso es vital identificar

las interacciones entre componentes. A continuación, se expone un ejemplo gráfico sobre esta descomposición funcional.

Figura 15. Ejemplo de descomposición funcional de un sistema de gestión de comercialización de productos (fragmento)



4.2. Especificación de requisitos

En esta sección se describen algunos estándares y/o técnicas que pueden ser usadas por las organizaciones para describir formalmente cada uno de los requisitos del sistema.

Estándar IEEE 830

Este estándar presenta un conjunto de prácticas recomendadas para la redacción de un documento de especificación de requerimientos, mejor conocido como SRS. Este documento está dividido en secciones y cada una de ellas aborda aspectos particulares. A continuación, se describirán de forma general algunos de los elementos que conforman este documento (IEEE 830-1998).

Ejemplo de la estructura de una matriz de trazabilidad

Lo invitamos a consultar el PDF " **Infografía EstandarIEEE830**", el cual se encuentra en la carpeta Anexos.

Este documento está dividido en secciones y cada una de ellas aborda aspectos particulares. A continuación, se describirá de forma general algunos de los elementos que conforman este documento (IEEE 830-1998).

Sumado a esto, se presenta la estructura base de un documento SRS, indicando cuáles son los apartados principales.

Estructura base de un documento SRS

- **Introducción**
  - Propósito
  - Ámbito del sistema
  - Definiciones, acrónimos y abreviaturas
  - Referencias
- **Descripción general**
  - Perspectiva del producto
  - Funciones del producto
  - Características de los usuarios
  - Restricciones
  - Suposiciones y dependencias
- **Requerimientos específicos**
  - Interfaz
  - Requisitos funcionales

- Requerimientos no funcionales
- Otros requisitos

- **Apéndices**

Se sugiere revisar algunos ejemplos de proyectos que se presentan sobre el diligenciamiento del formato SRS, los cuales se encuentran en el material complementario.

Estándar IEEE 29148:2018

Este estándar reemplaza los estándares IEEE 830, IEEE 1233, IEEE 1362, y contiene disposiciones para los procesos y productos relacionados con la ingeniería de requisitos para sistemas, productos y servicios de software a lo largo del ciclo de vida (Penzenstadler, 2021).

Además, define la construcción de un buen requisito, proporciona atributos y características de los requisitos, y analiza la aplicación iterativa y recursiva del proceso de requisitos a lo largo del ciclo de vida. También proporciona orientación adicional en la aplicación de procesos de ingeniería y gestión de requerimientos relacionados con la ingeniería de requisitos, al tiempo que define los elementos de información aplicables a la ingeniería de requisitos y su contenido.

El estándar IEEE 29148:2018 está estructurado de la siguiente forma:

- Introducción, resumen y tabla de contenido.
- Propósito, alcance del estándar, generalidades.
- Explicación de los otros estándares que lo conforman.
- Referencias a normas que lo conforman.

- Clarificación de la terminología, lo que es muy valioso para cuando se quiere establecer nuevos procesos de ingeniería de requerimientos en una empresa.
- Clarificación de los conceptos y procesos.
- Explicación y contenido de los ítems de información que vienen a través de la especificación de requerimientos o que debemos considerar incluir en la especificación de requerimientos.
- Anexos adicionales para mayor detalle.

Esta norma propone un listado de requerimientos mínimos, los cuales son la base de la especificación de requerimientos; en ese sentido, se proponen los siguientes tipos de requerimientos del sistema:

- Requerimientos funcionales: representan las necesidades de los interesados del software.
- Requerimientos de usabilidad: requerimientos que son utilizados directamente por los involucrados en la solución (requerimientos de uso).
- Requerimientos de desempeño: disponibilidad de servicios y procesos transaccionales.
- Interfaces del sistema: interacción entre personas con el software.
- Operaciones del sistema.
- Modos y estados del sistema.
- Características físicas (hardware).
- Condiciones del ambiente (operativas y operacionales).
- Seguridad del sistema.
- Manejo de la información.

- Políticas y regulación: normas y estándares que fundamentan el software.
- Ciclo de vida del sistema: establece las etapas y duración del desarrollo y uso en producción.

La especificación de requisitos a través de marcos de trabajo ágiles

Los marcos de trabajo ágiles promueven la comunicación oral sobre la documentación exhaustiva en la mayoría de los procesos del ciclo de vida, particularmente en los procesos de identificación de necesidades y diseño. Sin embargo, uno de los artefactos presentes para el modelado de requerimientos son las historias de usuario.

Las historias de usuario son una explicación general e informal de una función del software escrita desde la perspectiva del usuario final o cliente. Permiten describir de una manera muy breve un requerimiento, estimar prioridades, alcance y tiempo de realización (Rivadeneira, 2014). En la siguiente tabla, se puede observar la estructura base de un documento de historia de usuario.

La estructura base de un documento de historia de usuario es:

- Número
- Nombre de la historia de usuario
- Usuario
- Prioridad
- Puntos estimados
- Descripción
- Observaciones
- Criterios de aceptación

Las historias de usuario tienen varios beneficios respecto a otros instrumentos de redacción de requerimientos, entre los cuales se pueden listar:

- Las historias de usuario se centran en solucionar problemas a usuarios reales.
- Las historias de usuario permiten la colaboración, ya que como su descripción es corta, se necesita que el equipo colabore para decidir cómo dar solución a la historia para cumplir con la necesidad expresada por el usuario.
- Las historias impulsan la creatividad, ya que fomentan que el equipo piense de forma crítica y creativa sobre cómo solucionar de la mejor manera el objetivo.
- Las historias de usuario motivan a pensar en la mejor solución para una problemática particular, representan retos y pequeñas victorias para el equipo.

Scrum y la especificación de requisitos

El marco de trabajo Scrum está soportado en un proceso de construcción iterativo e incremental evolutivo, en el que se identifican tres roles principales: el equipo de trabajo (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 de un director de proyecto, pero más enfocados en garantizar que el equipo de trabajo tenga todas las herramientas y recursos necesarios para el desarrollo de su trabajo; y, finalmente, el dueño del producto (product owner) que se convierte en un representante del

cliente y quien es el único encargado de la gestión de requisitos del proyecto (ScrumStudy, 2021). En el siguiente video se explica un poco más sobre el Scrum y la especificación de los 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 iterativo 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 realiza funciones parecidas a las de un director de

proyecto, pero más enfocados que el equipo de trabajo tenga todas las herramientas y recursos necesarios para el desarrollo de su trabajo. Y finalmente el dueño de producto o Product owner que se convierte en un representante del cliente, quien es el único encargado de la gestión de requisitos del proyecto.

Scrum establece el concepto de sprint para referirse a una iteración que contempla tiempos fijos entre 2 y 4 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 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.

El artefacto mediante el cual se condensan todos los requerimientos del sistema se denomina pila de producto (product backlog), la cual es una lista ordenada por prioridad de todos los requerimientos del sistema, generalmente descritos en la forma de historias de usuario.

Todas las historias de usuario se priorizan utilizando números enteros consecutivos de 1 hasta N, donde 1 representa la máxima prioridad. Esta priorización la realiza únicamente el dueño del producto y, para eso, utiliza la información que tiene sobre el negocio y recomendaciones de expertos buscando el mayor retorno de inversión a sus clientes. La priorización de las historias de usuario en la pila de producto

puede variar en el transcurso del tiempo, pero solo la podrá realizar el dueño del producto.

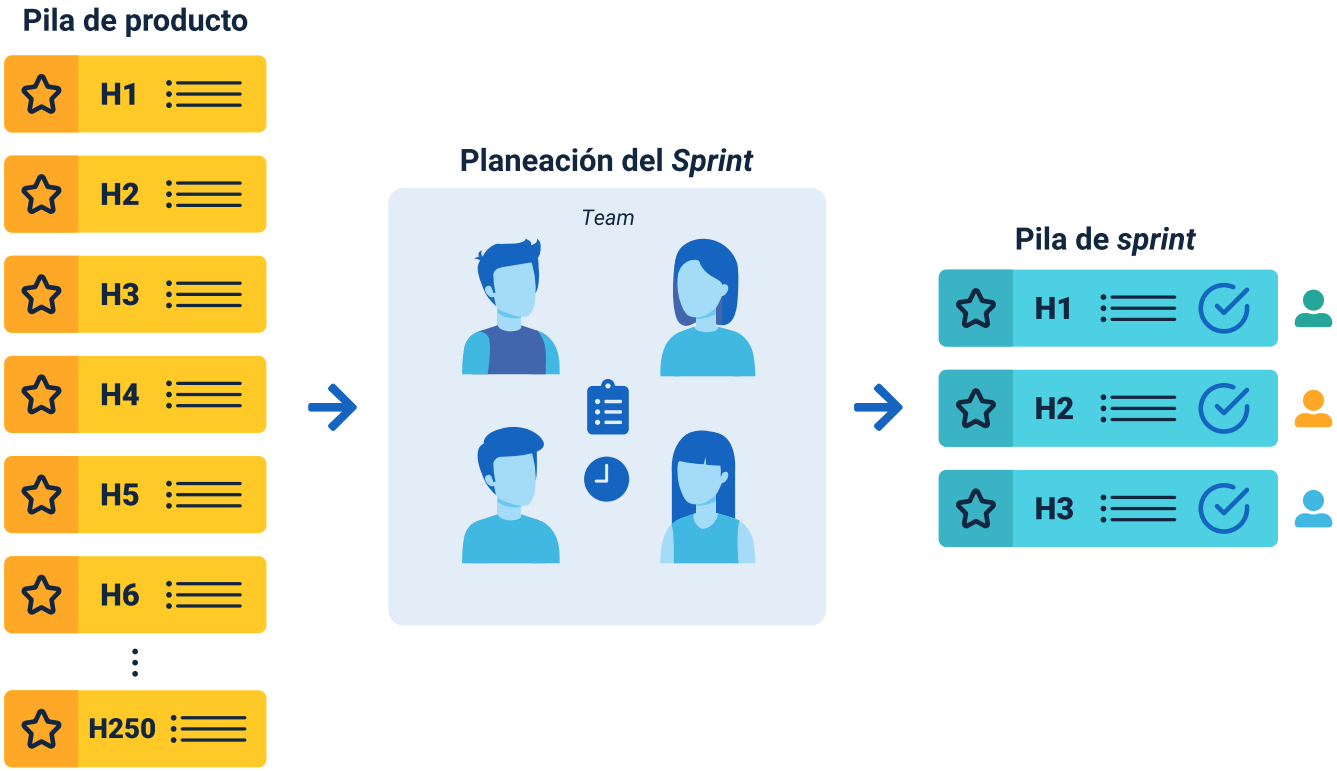
Scrum, siendo un marco de trabajo ágil, no requiere para su funcionamiento que todas las historias de usuario de la pila de producto estén detalladas, pero sí, por lo menos, las de mayor prioridad para poder iniciar el trabajo con el equipo en sus respectivos sprints (ScrumStudy, 2021).

Al inicio de cada sprint se realiza el proceso de planeación que involucra principalmente tres tareas:

- Estimar el valor de esfuerzo requerido para un conjunto de historias de usuario de la pila de producto, trabajo que es realizado únicamente por el equipo de desarrollo.
- Selección de las historias de usuario a desarrollar durante el sprint, tomando como referencia la prioridad y el valor del esfuerzo asociado a cada historia. Esto genera un artefacto llamado pila del sprint (sprint backlog) que lista los requerimientos descritos como historias de usuario a ser realizadas y evaluadas en el sprint.
- Descomposición de cada historia de usuario en tareas y, de ser necesario, asignar responsables a cada tarea.

A continuación, se expone una figura en la que se representan los artefactos generados dentro del marco de trabajo Scrum y que permiten la gestión de los requisitos y el evento desde el cual se construye inicialmente.

Figura 16. Pila de producto vs. pila del Sprint



Kanban y la especificación de requisitos

Kanban es una metodología para gestionar el trabajo que surge del sistema de producción Toyota Production System (TPS) a finales de la década de 1940, el cual representaba un sistema de producción basado en la demanda de los clientes y no en la producción masiva, lo anterior sentó los fundamentos para los sistemas de producción ajustada que consisten en minimizar los desperdicios sin afectar la producción y en crear más valor a los clientes sin generar más gastos.

A principios del siglo XXI, la industria del software adoptó el Kanban para cambiar la forma en la que se producían y entregaban productos y servicios; además, tiene en cuenta los principios de las metodologías ágiles, en especial de Scrum, pero

busca darle más protagonismo al proceso de experimentación y mejora continua (Rivadeneira, 2014).

Kanban en la industria del software se basa en cuatro principios fundamentales:

- Calidad garantizada, todo lo que se produce debe salir bien sin márgenes de errores, prima la calidad sobre la rapidez.
- Reducción del desperdicio: hacer solamente lo justo y necesario, pero hacerlo bien.
- Mejora continua.
- Flexibilidad: se pueden priorizar tareas entrantes según las necesidades del momento.

Además de los principios, Kanban propone seis (6) prácticas:

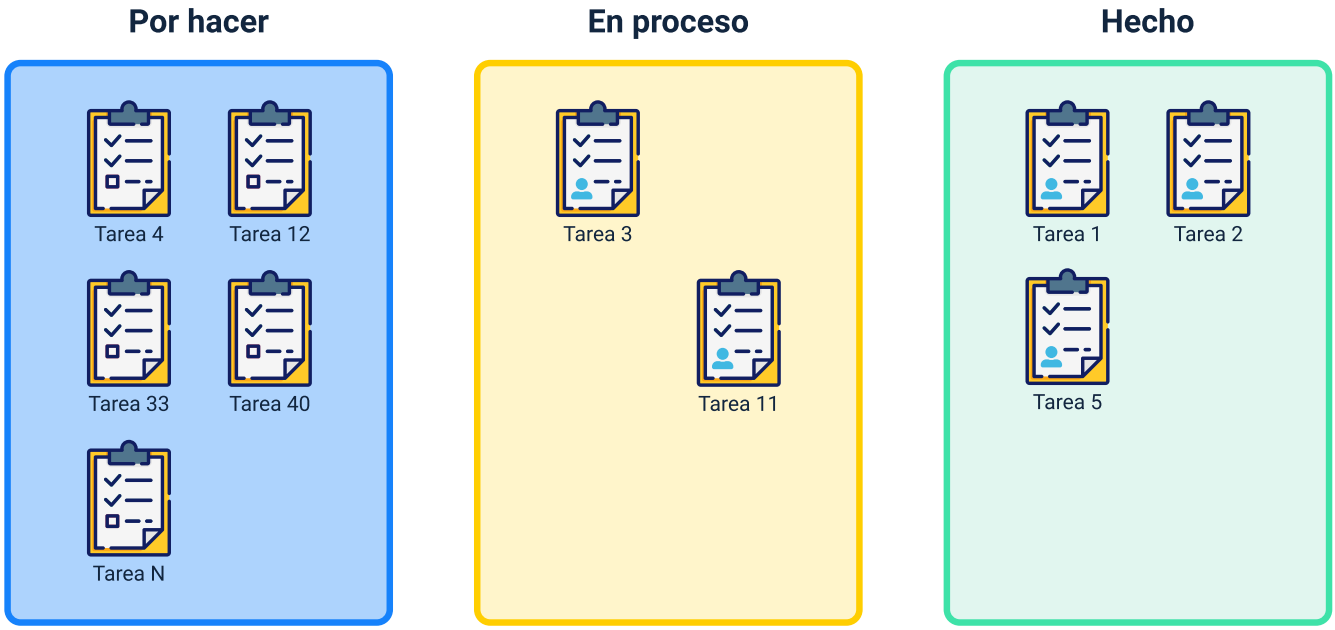
- Visualizar el flujo de trabajo.
- Eliminar interrupciones.
- Gestionar el flujo.
- Hacer políticas explícitas que fomenten la visibilidad.
- Circuitos de realimentación.
- Mejorar colaborando.

Tal vez la herramienta más conocida que permite implementar los principios de Kanban es el tablero Kanban, el cual permite mapear y visualizar el flujo de trabajo. Este se divide en columnas a través de las cuales se pueden visualizar cada una de las fases del proceso; las filas del tablero representan los diferentes tipos de actividades específicas que se desarrollan en el marco del proyecto.

Normalmente, el tablero tiene tres secciones que representan el estado de cada una de las tareas: por hacer, en proceso, hecho.

Cada equipo de trabajo puede realizar un mapeo más detallado de su proceso y agregar tanta sección como considere pertinente, como se muestra en la siguiente figura.

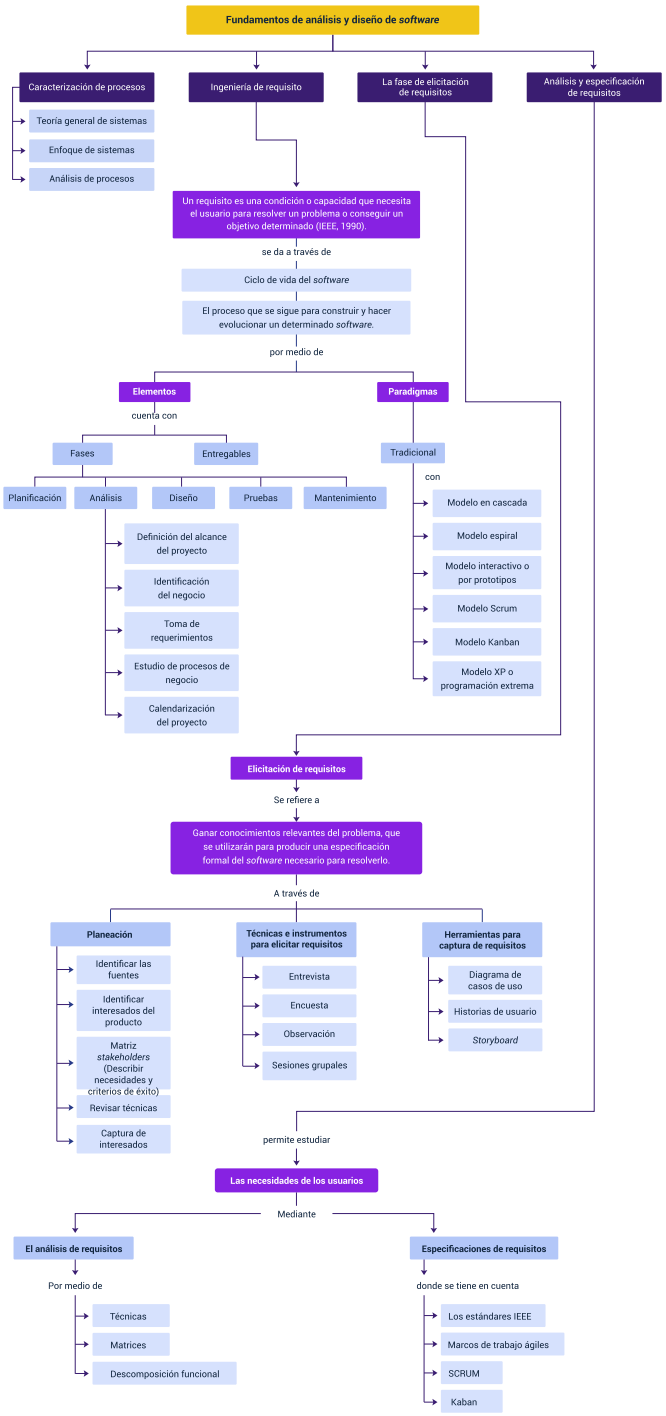
Figura 17. Tablero Kanban



Dependiendo del marco de trabajo, varía la forma en la que se describen cada una de las tareas del tablero Kanban, por ejemplo, dentro de un marco de trabajo como Scrum cada una de las tareas se podría describir en el formato de historias de usuario.

Síntesis

A continuación, se presenta una síntesis de la temática estudiada en el componente formativo:



Material complementario

Tema	Referencia	Tipo de material	Enlace del recurso
Análisis de los procesos a nivel de negocio	Saber programas. (2021). Cómo crear un DIAGRAMA de FLUJO en WORD paso a paso 2021. [Video]. YouTube.	Video	https://www.youtube.com/watch?v=nJq8A85zNZU
Análisis de los procesos a nivel de negocio	Lucidchart Software. (2021). Lucichart.com.	Página web	https://www.lucidchart.com/pages/es
Análisis de los procesos a nivel de negocio	Digignos. (2013). Elaboración de procesos. Sugerencia de una metodología. [Video]. YouTube.	Video	https://www.youtube.com/watch?v=-U-RQKJ9KKg&abchannel=Digignos
Ingeniería del software - Ciclo de vida	Universidad Católica de Murcia. (2015). Ingeniería del software - Ciclo de vida - Raquel Martínez. YouTube.	Video	https://www.youtube.com/watch?v=4tWmULUzVdE&t=199s
Tipos de requerimientos	Itunes U – UAEH. (2019). Tipos de requerimientos. YouTube.	Video	https://www.youtube.com/watch?v=Lv7XbZtnQ6A
Planeación	todopmp.com. (s.f.). Guía PMBOK 6 – 49 procesos, entradas, herramientas y salidas.	Página web	https://todopmp.com/cards/
Planeación	EDAP – Project Business School. (2016). MOOC PMP 302 Identificar Interesados	Video	https://www.youtube.com/watch?v=aUkTxgaajBo
Planeación	Virtual Training Lteam. (2016). Partes Interesadas Stakeholders	Video	https://www.youtube.com/watch?v=9AtaIAZEu0c

Tema	Referencia	Tipo de material	Enlace del recurso
Planeación	Calle, M. (2020). Análisis de Interesados Matriz Poder Interés - PMI	Video	https://www.youtube.com/watch?v=hDZ0uu0H1wc
Técnicas e instrumentos para elicitar requisitos	Jibaro X. (2019). Tipos de Preguntas en una encuesta	Video	https://www.youtube.com/watch?v=mwnQuUi9014
Especificación de requisitos	UMNG. (2019). Elementos de la norma IEEE 830. YouTube.	Video	http://www.foss2serve.org/index.php/RequirementsEngineering,CSULongBeach,Penzenstadler
Especificación de requisitos	UMNG. (2019). Elementos de la norma IEEE 830. YouTube.	Video	https://ieeexplore.ieee.org/document/720574
Estándar IEEE 830	Scrum Certification, Agile Certification | Scrum, Agile Training. (n.d.).ScrumStudy.	Video	https://www.scrumstudy.com/

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, estándares, políticas y objetivos para crear y modificar un conjunto de productos de trabajo.

ANSI: American National Standards Institute.

Ciclo de vida de software: aplicación de metodologías para el desarrollo del sistema software [AECC, 1986].

Estándar: referencia, patrón o modelo que es utilizado a nivel general en un determinado ámbito.

Marcos de trabajo ágiles: conjunto de estándares, metodologías, técnicas, frameworks o guías que rigen un proceso de desarrollo de software basadas en principios y/o valores ágiles como, por ejemplo: Scrum, Lean Software, XP, TDD, entre otros.

Marcos de trabajo tradicionales: conjunto de estándares, metodologías, técnicas, frameworks o guías que rigen un proceso de desarrollo de software basadas en el ciclo de vida tradicional del software como, por ejemplo: RUP, CMMI, ISO 9001, Microsoft Solution Framework, entre otros.

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

Metodología: síntesis de un conjunto de técnicas, métodos y procedimientos que se deben seguir durante el desarrollo de un proyecto.

Pert: Program Evaluation Research Task.

Pruebas de integración: prueba que se ejecuta una vez se aprueban las pruebas unitarias y lo que busca es verificar que el conjunto de fragmentos de código funciona junto de forma correcta. Es una prueba de conjunto.

Pruebas unitarias: forma de comprobar el correcto funcionamiento de una unidad de código.

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

Requisito: es la condición que debe cumplir algo, en general el requisito cumple con lo que se requiere con el requerimiento.

Retroalimentación: cuando parte de un mensaje de salida se convierte nuevamente en entrada.

Simulación: corresponde a la posibilidad de conducir experimentos en una computadora.

Sinergia: cuando varios elementos de una organización actúan de manera concentrada.

Stakeholders: individuo u organización que comparte, reclama o le interesa un sistema o le compete una característica que satisface sus necesidades y expectativas (ISO 29148).

Técnica: manera en la que un conjunto de procedimientos es aplicado en una tarea específica, con base en un conocimiento para obtener un resultado específico.

Teoría de colas: teoría que estudia los tiempos de espera dentro de un sistema.

Teoría de comportamientos: orientada al estudio del comportamiento humano en la administración.

Teoría de juegos: corresponde a la aplicación de un modelo matemático juego para entender la toma de decisiones.

TGS: Teoría General de sistemas.

Referencias bibliográficas

830-1998 - IEEE Recommended Practice for Software Requirements Specifications. (1998). IEEE Standard | IEEE Xplore. https://ieeexplore.ieee.org/document/720574

Baar, B. (2006). Using Stakeholder Analysis in Software Project Management. Universidad de Amsterdam.

Bertalanffy, L. (1968). Theory General Systems.

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

Braude, J. (2003). Ingeniería de software, una perspectiva orientada a objetos. Alfaomega.

Centro Tic de Andalucía. (2019). El enfoque sistémico.

Cohen, L. (2011). Métodos de investigación educativa. La Muralla.

Cohn, M. (2004). User Stories Applied for Agile Software Development. Pearson Education, Inc.

Cox, K., Niazi, M., y Verner, J. (2009). Empirical study of Sommerville and Sawyer's requirements engineering practices. IET Software, 3(5), 339. https://doi.org/10.1049/iet-sen.2008.0076

Curso de interacción persona-ordenador. (2021). Storyboarding. https://mpiua.invid.udl.cat/storyboarding

Denscombe, M. (2010). The Good Research Guide. McGraw-Hill Education.

Dornyei, Z. (2010). Questionnaires in Second Language Research: Construction, Administration, and Processing. Routledge.

Durán, A., Bernárdez, B., Ruiz, A. y Toro, M. (1999). A Requirements Elicitation Approach Based in Templates and Patterns.

ESAN Graduate School of Business. (2019). ¿Qué es el análisis de procesos de negocio y cómo aplicarlo en mi empresa?). https://www.esan.edu.pe/apuntes-empresariales/2019/11/que-es-el-analisis-de-procesos-de-negocio-y-como-aplicarlo-en-mi-empresa/

Espinal, I., Gimeno, A. y González, F. (1998). El enfoque sistémico en los estudios sobre la familia. https://www.uv.es/jugar2/Enfoque%20Sistemico.pdf

Gause, C., & Weinberg, G. M. (1989). Exploring Requirements: Quality Before Design. Dorset House.

Granollers, T., Lorés, J., y Perdrix F. (2002). Prototipado. Capítulo 5: modelo de proceso de la ingeniería de la usabilidad y de la accesibilidad.

Hernández, S., Fernández C., y Baptista L. (2006). Metodología de la investigación. McGraw Hill.

Herrera J., Lizka J. (2003). Ingeniería de requerimientos, ingeniería de software. http://www.monografias.com/trabajos6/resof/resof.shtml

McCracken, D., y Jackson, M. A. (1981). “A Minority Dissenting Opinion”. En W. W. Cotterman, J. D. Couger, N. L. Enger, F. Harold (Eds.). Systems Analysis and Design: A Foundation for the 1980s (pp. 551-553). Elsevier.

Pantaleo, G., y Rinaudo, L. (2018). Ingeniería de software. Alfaomega. https://www.iso.org/obp/ui/#iso:std:iso-iec:12207:ed-2:v1:en

Pantaleo, L., y Rinaudo. (2018). Ingeniería de software. Alfaomega.

Penzenstadler, B. (s. f.). Requirements Engineering. CSU Long Beach. http://www.foss2serve.org/index.php/RequirementsEngineering,CSULongBeach,Penzenstadler

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

Piattini M., Calvo-Manzano J., Cervera J., y Fernández, L. (2004). Análisis y diseño de aplicaciones informáticas de gestión. Una perspectiva de ingeniería de software. Alfaomega-Rama.

Porfirio, C. (2021). Técnicas de priorización: el desafío de conseguir un orden para las funcionalidades. https://www.knowmadmood.com/es/blog/tcnicas-de-priorizacin-el-desafo-de-conseguir-un-orden-para-las-funcionalidades

Rivadeneira, M., S. (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.

Tamayo. A (1999). Teoría general de sistemas.

Torres. A (2021). La Teoría General de Sistemas, de Ludwig von Bertalanffy. https://psicologiaymente.com/psicologia/teoria-general-de-sistemas-ludwig-von-bertalanffy

Ventura, M. T. (2002). La ingeniería de requerimientos como factor clave para el éxito de los proyectos de desarrollo de software.

Wessinger, K., (2012) Project Stakeholder Management. Engineering Management Journal, 14(84), 19-24.

Créditos

Nombre	Cargo	Centro de Formación y Regional
Milady Tatiana Villamil Castellanos	Responsable del Ecosistema	Dirección General
Olga Constanza Bermúdez Jaimes	Responsable de Línea de Producción	Centro de Servicios de Salud - Regional Antioquia
Carlos Hernán Muñoz Carvajal	Experto Temático	Centro de Teleinformática y Producción Industrial - Regional Cauca
Zulema Yidney León Escobar	Experta Temática	Centro de Teleinformática y Producción Industrial - Regional Cauca
Jonathan Guerrero Astaiza	Experto Temático	Centro de Teleinformática y Producción Industrial - Regional Cauca
Henry Eduardo Bastidas Paruma	Experto Temático	Centro de Teleinformática y Producción Industrial - Regional Cauca
Peter Emerson Pinchao Solis	Experto Temático	Centro de Teleinformática y Producción Industrial - Regional Cauca
Ana Catalina Córdoba Sus	Evaluadora Instruccional	Centro de Servicios de Salud - Regional Antioquia
Yerson Fabián Zárate Saavedra	Diseñador de Contenidos Digitales	Centro de Servicios de Salud - Regional Antioquia
Edward Leonardo Pico Cabra	Desarrollador Fullstack	Centro de Servicios de Salud - Regional Antioquia
Edgar Mauricio Cortés García	Actividad Didáctica	Centro de Servicios de Salud - Regional Antioquia
Jaime Hernán Tejada Llano	Validador de Recursos Educativos Digitales	Centro de Servicios de Salud - Regional Antioquia

Nombre	Cargo	Centro de Formación y Regional
Margarita Marcela Medrano Gómez	Evaluador para Contenidos Inclusivos y Accesibles	Centro de Servicios de Salud - Regional Antioquia
Daniel Ricardo Mutis Gómez	Evaluador para Contenidos Inclusivos y Accesibles	Centro de Servicios de Salud - Regional Antioquia