PROCESO DE GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL
FORMATO GUÍA DE APRENDIZAJE

1. IDENTIFICACIÓN DE LA GUÍA DE APRENDIZAJE

- **Denominación del programa de formación:** Análisis y desarrollo de *software*
- **Código del programa de formación:** 228118
- **Nombre del proyecto:** Construcción de *software* integrador de tecnologías orientadas a servicios
- **Fase del proyecto:** análisis.
- **Actividad de proyecto:** Determinar las especificaciones funcionales del *software* y metodología a utilizar.

- **Competencias:**
  Técnicas
  220501092- Establecer requisitos de la solución de *software* de acuerdo con estándares y procedimiento técnico.
  220501093- Evaluar requisitos de la solución de *software* de acuerdo con metodologías de análisis y estándares**.**
  Claves
  220501046- Utilizar herramientas informáticas de acuerdo con necesidades de manejo de información.
  240202501- Interactuar en lengua inglesa de forma oral y escrita dentro de contextos sociales y laborales según los criterios establecidos por el Marco Común Europeo de Referencia para las Lenguas**.**

- **Resultados de aprendizaje a alcanzar:**
  Técnicos:
  220501092-01- Caracterizar los procesos de la organización de acuerdo con el *software* a construir.
  220501092-02- Recolectar información del *software* a construir de acuerdo con las necesidades del cliente.
  220501092-03- Establecer los requisitos del *software* de acuerdo con la información recolectada.
  220501092-04- Validar el informe de requisitos de acuerdo con las necesidades del cliente.
  220501093-01- Planear actividades de análisis de acuerdo con la metodología seleccionada.
  Claves:
  220501046-01- Alistar herramientas de tecnologías de la información y la comunicación (TIC), de acuerdo con las necesidades de procesamiento de información y comunicación.
  220501046-02- Aplicar funcionalidades de herramientas y servicios TIC, de acuerdo con manuales de uso, procedimientos establecidos y buenas prácticas.
  220501046-03- Evaluar los resultados, de acuerdo con los requerimientos.
  220501046-04- Optimizar los resultados, de acuerdo con la verificación.
  240202501-01- Comprender información sobre situaciones cotidianas y laborales actuales y futuras a través de interacciones sociales de forma oral y escrita.
  Duración de la guía: 336 horas.
- Técnico: 240 horas.
- Claves: 96 horas.

2. PRESENTACIÓN

Estimado aprendiz, el SENA extiende una cordial bienvenida a la primera guía de aprendizaje que comprende las competencias técnicas de establecer requisitos de la solución de *software* de acuerdo con estándares y procedimiento técnico, y la competencia de evaluar requisitos de la solución de *software* de acuerdo con metodologías de análisis y estándares.

Por otra parte, ya no es un secreto que la sociedad ha avanzado a pasos agigantados en los procesos de las diferentes áreas de ocupación. Una de las áreas que más ha tenido repunte es la informática, que se encarga del estudio del *hardware*, las redes de datos y el *software* necesario para tratar la información de manera automática y se convierte en factor primordial a la hora de gestionar la información para la administración de métodos, técnicas y procesos en todas las áreas de ocupación.

Finalmente, encontrarán la competencia clave de inglés como una de las habilidades contemporáneas más importantes y de mayor impacto en el mundo laboral y social, teniendo en cuenta el aprendizaje articulado de las cuatro habilidades de la lengua (leer, escribir, hablar y escuchar) alineado al MCERL (Marco Común Europeo de Referencia para Lenguas) como estándar del nivel de conocimiento de lengua y categorización de los conocimientos en niveles de competencia en un idioma.

Para el desarrollo de las actividades planteadas en esta guía contará con el acompañamiento de los instructores asignados al programa, los cuales de forma continua y permanente lo orientarán con las pautas necesarias para el logro de las actividades de aprendizaje, brindando herramientas básicas de tipo conceptual y metodológico. Los instructores programarán encuentros de asesoría virtual, para brindar orientaciones específicas relacionadas con las temáticas a desarrollar en las actividades. Es importante que organice su tiempo, dada la exigencia que demanda la realización de esta guía de aprendizaje. No olvide revisar y explorar los materiales de estudio del programa.

Por consiguiente, se presentan cada una de las acciones de aprendizaje que le permitirán desarrollar lo anteriormente mencionado.

3. FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE

En este apartado se describirán las actividades de aprendizaje para cada una de las competencias que plantea la fase de análisis del proyecto formativo: determinar las especificaciones funcionales del *software* y metodología a utilizar.

Es importante que organice su tiempo, dada la exigencia que demanda la realización de las actividades mencionadas en esta guía de aprendizaje. No olvide revisar y explorar los materiales de formación del programa. Para ello, debe reconocer el espacio de trabajo, junto con las posibilidades que se tiene para interactuar, comunicar, visualizar y utilizar las herramientas necesarias en el LMS desde su rol de aprendiz. En este sentido, le invitamos a realizar las siguientes acciones:

- **Actualización de los datos personales**.
- Consulte la **Información del programa**, disponible en la plataforma, para conocer la descripción del proceso de formación y la metodología a seguir.
- Lea el presente documento **Guía de aprendizaje**, el cual le orientará en el desarrollo de las actividades.

- Tenga en cuenta que es fundamental entender los resultados de aprendizaje para tener mayor comprensión del proceso.
- Revisar el **cronograma** que le permitirá conocer la planeación diseñada para lograr de manera secuencial los resultados de aprendizaje.
- Participar de los espacios de comunicación, tales como foros y otros.

3.1. Actividades de aprendizaje de la competencia 220501092: establecer requisitos de la solución de *software* de acuerdo con estándares y procedimiento técnico

La recolección de requerimientos cumple un papel primordial en el proceso de desarrollo de *software*: la definición de lo que se desea producir. Su principal tarea consiste en la generación de especificaciones correctas que describan con claridad, sin ambigüedades, en forma consistente y compacta, el comportamiento del sistema; de esta manera, se pretenden minimizar los problemas relacionados con el desarrollo de sistemas.

3.1.1 Actividad de aprendizaje GA1-220501092-AA1: elaborar diagramas de procesos a partir de la caracterización de los mismos usando una técnica de notación.

Esta actividad se centra en el estudio de las diferentes características asociadas a la teoría general de sistemas conceptos, tipos de sistemas, sistemas de información.

Duración: 48 horas.

Materiales de formación: para el desarrollo de esta actividad es importante la lectura y análisis del material de formación: "**Caracterización de procesos".**

Evidencias:

A continuación, se describen las acciones y las correspondientes evidencias que conforman la actividad de aprendizaje:

- **Evidencias de conocimiento: GA1-220501092-AA1-EV01: infografía sobre la teoría General de Sistemas**

Teniendo en cuenta los conceptos vistos en el componente caracterización de procesos realizar una infografía sobre la Teoría General de Sistemas.

Elementos a tener en cuenta en el documento de requisitos:

- Se deben seguir las normas básicas de creación de infografías como tipo, elementos que incluyen, imágenes, ilustraciones, viñetas, mapas conceptuales etc.
- Debe contener los principales conceptos de la teoría general de sistemas.
- Puede utilizar cualquier herramienta tic o informática para realizar la infografía.

Lineamientos para la entrega del producto:

- **Producto para entregar:** documento con infografía.
- **Formato:** PDF o Word.
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **infografía sobre la teoría General de Sistemas GA1-220501092-AA1-EV01**

- **Evidencia de conocimiento: GA1-220501092-AA1-EV02 identificación de procesos organizacionales.**

Teniendo en cuenta las características del *software* a construir y las diferentes fuentes de información existentes identificar el conjunto de elementos que intervienen (relaciones, actores responsables) en este aplicando la TGS (Teoría General de Sistemas).

Elementos para tener en cuenta en el documento de requisitos: se deben seguir las normas básicas de presentación de un documento escrito, es decir el documento debe tener como mínimo una portada, introducción y nombramiento de los elementos que intervienen en el *software* a construir.

Lineamientos para la entrega del producto:

- **Producto para entregar:** documento de identificación de elementos.
- **Formato:** PDF o Word.
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **identificación de procesos organizacionales GA1-220501092-AA1-EV02**

- **Evidencia de producto: GA1-220501092-AA1-EV03 mapa de procesos del *software* a construir**

Teniendo en cuenta las características del *software* a construir y tomando como base la identificación de elementos realizada en la evidencia **identificación de procesos organizacionales GA1-220501092-AA1-EV02**, realizar el mapa de procesos del *software*.

Elementos para tener en cuenta en el documento de requisitos:

- Se deben seguir las normas básicas de presentación de un documento escrito, es decir el documento debe tener como mínimo una portada, introducción y nombramiento de los elementos que intervienen en el *software* a construir.
- Se puede utilizar cualquier herramienta TIC para la realización del mapa de procesos.

Lineamientos para la entrega del producto:

- **Producto para entregar:** documento con el mapa de procesos y su descripción.
- **Formato:** PDF o Word.
- **Extensión:** libre.

- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **mapa de procesos del *software* a construir GA1-220501092-AA1-EV03**

3.1.2 Actividad de aprendizaje GA1-220501092-AA2: identificar conceptos básicos de teoría general de sistemas y enfoque sistémico

Esta actividad se centra en la identificación de fuentes de requisitos.

Duración: 12 horas.

Materiales de formación: para el desarrollo de esta actividad es importante la lectura y análisis del material de formación: "**Ingeniería de requisitos".**

Evidencias:

- **Evidencia de Conocimiento: GA1-220501092-AA2-EV01 mapa mental sobre ingeniería de requisitos**

A continuación, se describen las acciones y las correspondientes evidencias que conforman la actividad de aprendizaje:

- Realizar un mapa mental con los conceptos sobre ingeniería de requisitos vistos en el componente formativo "**Ingeniería de requisitos".**
- Utilizar una herramienta TIC para la realización del mapa mental.
- El mapa debe tener las principales características de los mapas mentales.

Lineamientos para la entrega del producto:

- **Producto para entregar:** mapa mental con los conceptos sobre ingeniería de requisitos.
- **Formato:** PDF o Word.
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **mapa mental sobre ingeniería de requisitos-GA1-220501092-AA2-EV01.**

- **Evidencia de desempeño GA1-220501092-AA2-EV02: foro temático. Fuentes de requisitos**

Conteste la pregunta planteada con referencia al tema de identificación de fuentes de requisitos y de la opinión de las respuestas de, al menos, tres (3) de sus compañeros.

- Responda la pregunta: ¿cuáles son los tipos de fuentes y de un ejemplo de cada una? ¿En qué caso uso cada tipo de fuente?
- Se debe responder de forma concisa donde el aprendiz de la opinión con referencia al tema.
- El aprendiz debe dar su opinión a la respuesta de, al menos, tres (3) compañeros.
- Se deben cumplir con normas ortográficas en las respuestas realizadas.

Lineamientos para la entrega del producto:

- **Producto para entregar:** respuesta foro temático.
- **Formato:** PDF o Word.
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **foro temático. Fuentes de requisitos GA1-220501092-AA2-EV02**

3.1.3 Actividad de aprendizaje: GA1-220501092-AA3 elaborar instrumentos de recolección de datos, de acuerdo a técnicas que permitan el procesamiento de la información.

Esta actividad se centra en el estudio de los componentes de la fase de elicitación de requisitos, técnicas y herramientas de captura de requisitos.

Duración: 24 horas.

Materiales de formación: Para el desarrollo de esta actividad es importante la lectura y análisis del material de formación: "**La fase de elicitación de requisitos".**

Evidencias:

A continuación, se describen las acciones y las correspondientes evidencias que conforman la actividad de aprendizaje:

- **Evidencia de desempeño: GA1-220501092-AA3-EV01 diseño del instrumento de recolección de información**

Seleccione una de las técnicas utilizadas para la licitación de requisitos y realice el diseño de un instrumento de recolección de información, teniendo en cuenta las características del *software* a construir.

Elementos a tener en cuenta en el documento de requisitos:

- Se deben seguir las normas básicas de presentación de un documento escrito, es decir el documento debe tener como mínimo una portada, introducción, objetivo, preguntas claras y alcance

Lineamientos para la entrega del producto:

- **Producto para entregar:** documento de diseño
- **Formato:** PDF o Word.
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **diseño del instrumento de recolección de información GA1-220501092-AA3EV01**

- **Evidencia de desempeño: GA1-220501092-AA3-EV02 formulación del proyecto de *software*.**

Teniendo en cuenta las características del *software* a construir seleccione una de las herramientas para captura de requisitos y defina sus funcionalidades en un documento.

Elementos a tener en cuenta en el documento de requisitos:

- Se deben seguir las normas básicas de presentación de un documento escrito, es decir el documento debe tener como mínimo una portada, introducción, objetivo, preguntas claras y alcance
- Seleccione una de las herramientas para captura de requisitos mencionada en el componente (diagramas de casos de uso, historias de usuario, *storyboard*) para realizar la formulación de los requisitos.
- Utilice una de las herramientas o plantillas (starUml, plantilla historias de usuario) que se describieron en el componente formativo.

Lineamientos para la entrega del producto:

- **Producto para entregar:** documento de requisitos.
- **Formato:** PDF o Word.
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **formulación del proyecto de *software*. GA1-220501092-AA3-EV02**

- **Evidencia de producto: GA1- 220501092-AA3-EV03 formulario de recolección de información**

Teniendo en cuenta las características del *software* a construir elaborar un documento donde se pueda identificar las técnicas de licitación para el levantamiento de requisitos utilizada.

Elementos a tener en cuenta en el documento de requisitos:

- Se deben seguir las normas básicas de presentación de un documento escrito, es decir el documento debe tener como mínimo una portada, introducción, las técnicas utilizadas para el levantamiento de requisitos, y el análisis arrojado después de aplicadas las técnicas.
- Respecto a la técnica deberá adjuntar el formato o la estructura de la técnica realizada; si es entrevista, el guion utilizado; si es encuesta, las preguntas realizadas, etc.
- Respecto al análisis se deberá realizar un informe de los resultados de la técnica aplicada.

Lineamientos para la entrega del producto:

- **Producto para entregar:** documento con formulario**.**
- **Formato:** PDF o Word.
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **formulario de recolección de información GA1- 220501092-AA3-EV03.**

3.1.4 Actividad de aprendizaje: GA1-220501092-AA4: determinar los requisitos funcionales y no funcionales del software de acuerdo con los requerimientos del cliente

Esta actividad se centra en dos grandes bloques:

(i) análisis de las diferentes técnicas de análisis de requisitos

(ii) estándares o modelos para el proceso de documentación de requisitos del *software*

Duración: 36 horas.

Materiales de formación: para el desarrollo de esta actividad es importante la lectura y análisis del material de formación: "**Análisis y especificación de requisitos".**

Evidencias:

A continuación, se describen las acciones y las correspondientes evidencias que conforman la actividad de aprendizaje:

- **Evidencia de desempeño: GA1-220501092-AA4-EV01 especificación de los requerimientos funcionales y no funcionales del *software***

Construir el documento de requisitos que especifique los requisitos funcionales y no funcionales teniendo en cuenta las características del *software* a realizar.

Elementos para tener en cuenta en el documento técnico de validación: se deben seguir las normas básicas de presentación de un documento escrito, es decir el documento debe tener como mínimo una portada, introducción, y la lista de:

  - Requisitos funcionales.
  - Requisitos no funcionales.

Lineamientos para la entrega del producto:

- **Productos para entregar:** documentos requerimientos funcionales y no funcionales.
- **Formato:** PDF o Word.
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **especificación de los requerimientos funcionales y no funcionales del *software* GA1-220501092-AA4-EV01**

- **Evidencia de producto: GA1-220501092-AA4-EV02 documento con especificación de requerimientos**

Respecto a lista de requerimientos el aprendiz deberá agregar una sección donde se describa cada requisito usando los siguientes elementos del estándar IEEE830.

  - Perspectiva del producto.
  - Funciones del producto.
  - Características de los usuarios.
  - Restricciones.
  - Requisitos funcionales (formato de casos de uso).
  - Requisitos no funcionales.

Respecto a la lista de requerimientos el aprendiz deberá agregar una sección donde se describa cada requisito usando la estructura de historias de usuario con los siguientes elementos por historia:

- Número de historia (priorizada).
- Nombre de la historia.
- Usuario.
- Puntos estimados de esfuerzo.
- Descripción de la historia de usuario.
- Observaciones.
- Criterios de aceptación.

Elementos a tener en cuenta en el documento técnico de validación: se deben seguir las normas básicas de presentación de un documento escrito, es decir el documento debe tener como mínimo una portada, introducción, alcance, lista de requerimientos y versión del documento. Los requerimientos serán redactados usando el modelo IEEE830 y también el modelo de descripción de requisitos por medio de historias de usuario.

Lineamientos para la entrega del producto:

- **Producto para entregar:** documento de requisitos.
- **Formato:** PDF o Word.
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **documento con especificación de requerimientos GA1-220501092-AA4-EV02**

3.1.5 Actividad de aprendizaje: GA1-220501092-AA5 aplicar técnicas de validación de requisitos del software.

Esta actividad se centra en el estudio de los componentes de la fase de elicitación de requisitos, técnicas y herramientas de captura de requisitos.

Duración: 24 horas.

Materiales de formación: para el desarrollo de esta actividad es importante la lectura y análisis del material de formación: "**Validación de requisitos".**

Evidencias:

A continuación, se describen las acciones y las correspondientes evidencias que conforman la actividad de aprendizaje:

- **Evidencia de conocimiento: GA1-220501092-AA5-EV01 taller para la determinación de las especificaciones funcionales del *software* y metodología a utilizar**

A partir de la revisión de las diferentes herramientas tecnológicas disponibles para la gestión de requisitos, seleccionar una para configurar los requerimientos del proyecto seleccionado.

Redactar un documento donde describa la herramienta a utilizar y colocar los pantallazos de la configuración.

Lineamientos para la entrega del producto:

- **Producto para entregar:** documento.
- **Formato:** PDF o Word.
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **taller para la determinación de las especificaciones funcionales del *software* y metodología a utilizar GA1-220501092-AA5-EV01.**

- **Evidencia de producto: GA1-220501092-AA5-EV02 informe de evaluación de los requerimientos**

Teniendo en cuenta las características de las técnicas de validación de requisitos abordadas en la actividad de aprendizaje, construir los prototipos del sistema y los casos de prueba asociados a cada interfaz desarrollada.

Elementos a tener en cuenta en el documento técnico de validación

- Se deben seguir las normas básicas de presentación de un documento escrito, es decir el documento debe tener como mínimo una portada, introducción, alcance, lista de requerimientos y versión del documento.
- Deberá seleccionar cuatro (4) historias de usuario o requisitos del proyecto seleccionado para representarlos por medio de prototipos y realizar sus respectivos casos de prueba.
- Para la construcción de prototipos deberá usar una de las herramientas propuestas en el componente formativo y exportar dichos elementos desarrollados como imágenes para ser incluidos dentro del documento.
- Por cada interfaz desarrollada deberá construir un caso de prueba el cual debe tener los siguientes elementos.
    - Objetivo del caso de prueba.
    - Identificador.
    - Nombre del requerimiento o historia de usuario asociada.
    - Precondiciones.
    - Lista de pasos con los resultados esperados.

Lineamientos para la entrega del producto:

- **Producto para entregar:** documento informe técnico de validación.
- **Formato:** PDF o Word.
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **informe de evaluación de los requerimientos GA1-220501092-AA5-EV02.**

3.2. Actividades de aprendizaje de la competencia 220501093 evaluar requisitos de la solución de *software* de acuerdo con metodologías de análisis y estándares

Terminada la fase de recolección de requerimientos, se procede a evaluar si los requisitos cumplen con todos los requisitos y es adecuado para dar solución a la problemática bajo las metodologías del proceso de desarrollo de *software*.

3.2.1 Actividad de aprendizaje: GA1-220501093-AA1 estructurar el plan de actividades de análisis a partir de las características del proyecto y el modelo de desarrollo seleccionado.

En esta actividad de aprendizaje se abordan las técnicas de validación de requisitos: revisiones, los prototipos y los casos de prueba.

Duración: 96 horas.

Materiales de formación: Para el desarrollo de esta actividad es importante la lectura y análisis del material de formación: "**Metodologías de desarrollo de *software*".**

Evidencias:
A continuación, se describen las acciones y las correspondientes evidencias que conforman la actividad de aprendizaje:

- **Evidencia conocimiento: GA1-220501093-AA1-EV01 taller sobre metodologías de desarrollo de *software***

Las metodologías de desarrollo son indispensables en los grupos de trabajo y organizaciones relacionadas con la industria de *software*, partiendo de la información abordada en este componente desarrollar el taller sobre metodologías de desarrollo de *software* propuesto.

Elementos para tener en cuenta en el taller:
- Seleccionar diferentes fuentes de información relacionadas con las metodologías de desarrollo de *software*.
- Detallar las características que identifican a los marcos de trabajo tradicionales y los marcos de trabajo ágiles.
- Utilizar imágenes de construcción propia o que tengan los derechos respectivos de uso.

Lineamientos para la entrega del producto:
- **Producto para entregar:** documento con el desarrollo del taller propuesto.
- **Formato:** PDF o Word.
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **taller sobre metodologías de desarrollo de *software*. GA1-220501093AA1-EV01.**

- **Evidencia conocimiento: GA1-220501093-AA1-EV02 infografía sobre metodologías de desarrollo de *software***

Realizar una investigación corta profundizando en las metodologías de desarrollo de *software* que existen en la industrial y a partir de esta investigación construir una infografía en la que se resuman las principales características, ventajas y desventajas de 3 marcos de trabajo ágiles y tres marcos de trabajo tradicionales.

Elementos para tener en cuenta en la infografía:

- Debe incluir tres marcos de trabajo ágiles y tres marcos de trabajo tradicionales.
- Debe incluir un marco de trabajo tradicional y un marco de trabajo ágil que no se haya desarrollado durante el componente formativo 6.
- Utilizar imágenes de construcción propia o que tengan los derechos respectivos de uso.

Lineamientos para la entrega del producto:

- **Producto para entregar:** infografía y metodologías de desarrollo.
- **Formato:** PDF o Word.
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **infografía sobre metodologías de desarrollo de *software*. GA1-220501093AA1-EV02.**

- **Evidencia de desempeño: GA1-220501093-AA1-EV03 Foro. Especificación de la metodología a aplicar**

Argumentar y debatir sobre la metodología de desarrollo a aplicar en el proyecto por medio del foro dispuesto para este fin.

Elementos para tener en cuenta en la actividad:

- Recuerde que toda participación en el foro debe presentar una argumentación clara, con buena ortografía.
- Apoyar cada argumentación teniendo en cuenta los siguientes elementos de las metodologías:
    - Origen.
    - Características / Principios.
    - Elementos / Roles / Herramientas de apoyo.
    - Gráfica resumen.
- Debe comentar las argumentaciones realizadas por otros.

Lineamientos para la entrega del producto:

- **Producto para entregar:** argumentación en el foro.
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **Foro. Especificación de la metodología a aplicar. GA1-220501093-AA1-EV03.**

- **Evidencia de producto: GA1-220501093-AA1-EV04 documento identificando la metodología para el proyecto de desarrollo de *software***

Teniendo en cuenta la información recopilada y la idea de proyecto selecciona realizar un informe donde se describa y justifique la metodología de desarrollo de *software* a utilizar.

Elementos para tener en cuenta en el informe:

- Se deben seguir las normas básicas de presentación de un documento escrito, es decir el documento debe tener como mínimo una portada, introducción, desarrollo y bibliografía.

- El informe debe evidenciar una justificación clara de la selección de la metodología respecto al proyecto a desarrollar.
- Debe incluir una descripción del contexto y características del proyecto.
- El informe debe evidenciar el uso de filtros tales como tamaño del proyecto, periodicidad de realimentación con el cliente, estado de la tecnología, etc.

Lineamientos para la entrega del producto:

- **Producto para entregar:** informe.
- **Formato:** PDF o Word
- **Extensión:** libre.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **documento identificando la metodología para el proyecto de desarrollo de *software*. GA1-220501093-AA1-EV04.**

3.3. Actividades de aprendizaje de la competencia 220501046 utilizar herramientas informáticas de acuerdo con necesidades de manejo de información

La informática se encarga del estudio del *hardware*, las redes de datos y el *software* necesario para tratar la información de manera automática y se convierte en factor primordial a la hora de gestionar la información para la administración de métodos, técnicas y procesos en todas las áreas de ocupación.

3.3.1 Actividad de aprendizaje: GA1-220501046-AA1 reconocer software de sistemas, software de programación y software de aplicaciones.

El objetivo de la actividad es la identificación y apropiación de los conceptos sobre el *software*, su clasificación y las herramientas que ofrece internet, aplicando la estrategia de la lluvia de ideas.

Duración: 12 horas.

Materiales de formación: Para el desarrollo de esta actividad es importante la lectura y análisis del material de formación: "**Tecnologías de la Información y la Comunicación (TIC)".**

Evidencias:

A continuación, se describen las acciones y las correspondientes evidencias que conforman la actividad de aprendizaje:

- **Evidencia de producto: GA1-220501046-AA1-EV01 mapa Conceptual - *Software* y servicios de internet. Relacionar correctamente los tipos de *software* y servicios de internet**

Realizar un mapa conceptual sobre los tipos de *software* y servicios de internet, utilizando figuras, imágenes, frases cortas y colores. Para la elaboración del mapa conceptual debe ser creativo y se recomienda:

- Exploración de la plataforma y documentos guía del programa de formación.
- Prestar atención a las características y componentes propias de los tipos de *software* y servicios de internet presentados en los recursos educativos digitales del componente.

- Simultáneamente tomar nota de los elementos que considere relevantes sobre tipos de *software* y servicios de internet para incluirlos en el mapa conceptual.
- Puede revisar el material tantas veces como lo considere necesario, lo importante es que sea una construcción propia a partir del análisis de la información presentada, su experiencia y/o material complementario.
- Incluya aspectos relevantes sobre los tipos de *software* y recursos de internet para el área ocupacional de su desempeño.
- Si tiene dudas sobre cómo realizar un mapa conceptual y las herramientas a utilizar para realizar la entrega de la evidencia de aprendizaje, explore el anexo "Mapa-220501046-AA1-EV01" para el aprendiz que le orientará en el proceso.

Lineamientos para la entrega del producto:

- **Producto para entregar:** mapa conceptual sobre los tipos de *software* y servicios de internet.
- **Formato:** PDF con el desarrollo del mapa y en la parte inferior derecha colocar sus datos personales.
- **Extensión:** 1 hoja.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **mapa Conceptual - *Software* y servicios de internet. Relacionar correctamente los tipos de *software* y servicios de internet -GA1-220501046-AA1-EV01.**

3.2.2 Actividad de aprendizaje: GA1-220501046-AA2 desarrollar un taller práctico sobre los términos y funcionalidades de la ofimática

El objetivo de esta actividad es el dominio de herramientas ofimáticas y colaborativas, aplicando la estrategia de aprender haciendo mediante la resolución de un taller propuesto.

Duración: 24 horas.

Materiales de formación: Para el desarrollo de esta actividad es importante la lectura y análisis del material de formación: "**Tecnologías de la Información y la Comunicación (TIC)".**

Evidencias:

A continuación, se describen las acciones y las correspondientes evidencias que conforman la actividad de aprendizaje:

- **Evidencia Conocimiento: GA1-220501046-AA2-EV01 taller- Utilización de las herramientas de Ofimática. Realizar un taller práctico con las herramientas ofimáticas**

Realizar un taller práctico con las herramientas ofimáticas, tomando como base los temas presentados en el componente formativo **"Tecnologías de la Información y la Comunicación (TIC)"**. Para la elaboración del taller se debe tener en cuenta:

- Parte 1. Realizar el taller sobre la aplicación de herramientas informáticas que se presenta en el archivo comprimido anexo "**Taller_Ofimatica-220501046-AA2-EV01**". Este taller se debe realizar de manera Individual.
- Parte 2. Tomando como base el siguiente enlace sobre el uso de Trello: https://blog.trello.com/es/como-usartrello, se debe elaborar un tablero de Trello y realizar las diferentes acciones descritas en el enlace (incluir múltiples tarjetas, elementos *checklist*, fondos personalizados y responsabilidades), se deben organizar

equipos de máximo cinco (5) integrantes para compartir el tablero creado el cual debe compartirse además con el instructor.

Lineamientos para la entrega del producto:

- **Producto para entregar:** se debe entregar un archivo comprimido que contenga:
- El archivo en formato Word con la parte 1 de la actividad, es decir, el taller sobre herramientas ofimáticas.
- Un documento que describa las acciones realizadas en la parte 2 de la actividad, es decir, la construcción del tablero en Trello, además de la dirección de acceso al mismo. El informe debe incluir las dificultades encontradas en el uso de esta herramienta.
- **Formato:** archivo comprimido en formato RAR o ZIP.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **taller- Utilización de las herramientas de Ofimática. Realizar un taller práctico con las herramientas ofimáticas- GA1-220501046-AA2-EV01.**
-

3.2.2 Actividad de aprendizaje: GA1-220501046-AA3 elaborar un informe con las mejoras de producto orientado desde las TIC.

El objetivo de esta actividad es identificar y reconocer la importancia de la incorporación de las Tecnologías de la Información y la Comunicación (TIC) en los procesos relacionados con las diferentes áreas y su injerencia en la mejora de la productividad.

Duración: 12 horas.

Materiales de formación: para el desarrollo de esta actividad es importante la lectura y análisis del material de formación: "**Tecnologías de la Información y la Comunicación (TIC)".**

Evidencias:

A continuación, se describen las acciones y las correspondientes evidencias que conforman la actividad de aprendizaje:

- **Evidencia de desempeño: GA1-220501046-AA3-EV01 informe mejora de productos y procesos con la incorporación de TIC**

Elaborar un informe sobre las mejoras que pueden realizarse a los diferentes procesos y productos del área de ocupación con la incorporación de las TIC. Para esto debe tener en cuenta:

- Plantear las mejoras que ofrece la incorporación de las Tecnologías de la Información y la Comunicación (TIC), en lo concerniente con las herramientas ofimáticas y colaborativas.
- Identificar los aspectos procesos del área de ocupación que se pueden fortalecer con la incorporación de las Tecnologías de la Información y la Comunicación (TIC).
- Presentar el informe de forma creativa, por ejemplo, a través de cuentos, caricaturas, etc. Haciendo uso de una herramienta TIC en línea.

Lineamientos para la entrega del producto:

- **Producto para entregar:** informe sobre las mejoras que pueden realizarse a los diferentes productos y procesos del área de ocupación con la incorporación de TIC**.**

- **Formato:** PDF.
- **Extensión:** máximo 10 hojas.
- Para hacer el envío del producto remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **informe mejora de productos y procesos con la incorporación de TIC GA1220501046-AA3-EV01.**

3.4. Actividades de aprendizaje de la competencia 240202501 interactuar en lengua inglesa de forma oral y escrita dentro de contextos sociales y laborales según los criterios establecidos por el Marco Común Europeo de Referencia para las Lenguas

Para comenzar a interactuar tanto de forma oral como escrita acerca de las actividades de la vida diaria, se requiere comprender información personal y familiar haciendo uso de la estructura, el vocabulario y contextos requeridos.

3.4.1 Actividad de aprendizaje: GA1-240202501-AA1 identificar situaciones cotidianas y futuras a través de una interacción social oral y escrita

El aprendizaje de una segunda lengua se facilita cuando se comienza a estudiar desde temas conocidos como son las situaciones de la cotidianidad ya sea abordándolas desde el momento actual.

Es importante seguir las indicaciones del instructor en cuanto a la mejor manera de apropiar el material del curso y la correlación que tiene este con el programa de formación.

Duración: 48 horas.

Material de formación: los contenidos de soporte para el desarrollo de la actividad son **"LEVEL 1 - MCER A1.1".**

Evidencias:

- **Evidencia de conocimiento: GA1-240202501-AA1-EV01 Cuestionario**

Presenta un cuestionario (evaluación en línea) para evaluar comprensión lectora y gramatical del nivel, el cual consta de quince preguntas (15) y un tiempo aproximado de 45 minutos.

Lineamientos para la entrega de la evidencia:

- Para responder el cuestionario (evaluación en línea), remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia **Cuestionario GA1-240202501AA1-EV01.**

- **Evidencia de desempeño: GA1-240202501-AA1-EV02 Video**

De acuerdo con la temática estudiada en el primer nivel escoja un miembro de su familia o un amigo para presentarlo por medio de un corto video activando su cámara web en inglés con el fin de identificar de forma oral las características personales sobre edad, nacionalidad, lugar de residencia, hobbies y alguna información relevante sobre las actividades cotidianas que realiza.

Para la elaboración del video con la cámara web, tenga en cuenta el desarrollo de un guion o estructura a través de una herramienta como PowerPoint, Emaze, Prezi en la que se dispongan algunas diapositivas con imágenes y textos para dar cuenta de la presentación del personaje.

Diapositivas: durante el desarrollo de la presentación oral es ideal que presente entre 3 y 5 diapositivas, donde se incluyan los siguientes elementos:

- Diapositiva de portada (datos básicos del aprendiz, nombre del curso, instructor y nombre de la actividad).
- Diapositivas de datos del personaje seleccionado (nombre completo, edad, fecha de cumpleaños, ocupación, actividades de tiempo libre), acompañado de imágenes, frases, textos cortos y elementos gráficos que le permitan hablar durante la presentación.

Estructura del video con cámara web: cuando grabe el video, tenga en cuenta que las fotos, imágenes y textos que use en las diapositivas deben ser un apoyo visual para denotar buena pronunciación y su aprendizaje inicial sobre las temáticas del primer nivel.

Para la realización de la emisión del video deberá encender su cámara web, mostrar la pantalla con las diapositivas creadas. La recomendación es utilizar alguna herramienta digital que permite grabar el video y pantalla como Screencast-o-Matic, Loom, Camtasia, recordscreen.io, scrnrcrd.com e incluso existen aplicaciones como X Recorder para que pueda realizarlo desde su teléfono móvil. Lo importante es mostrar las diapositivas, su cámara web e ir realizando su presentación de forma oral.

Una vez finalizado el video debe cargarlo a YouTube o Vimeo, con su cuenta de correo personal o institucional; compruebe que no tenga restricciones de visualización para que pueda compartir el enlace de visualización.

Lineamientos para la entrega de la evidencia:

- **Producto a entregar**: documento con los datos del aprendiz y enlace del Video.
- **Formato**: Word o PDF con la URL del video.
- **Extensión:** de 2 a 5 minutos.
- Para hacer el envío del documento remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia **Video GA1-240202501-AA1-EV02.**

- **Evidencia de producto: GA1-240202501-AA1-EV03. Folleto**

A partir de los datos, conceptos, e investigación obtenida de su programa de formación deberá realizar un folleto en inglés con la información y características de su programa de formación.

Para realizar el folleto puede utilizar alguna herramienta digital como Canva, Crello, Word, PowerPoint o cualquier otra que conozca y le permita trabajar de forma creativa.

Estructura: datos personales del aprendiz, mensaje inicial, información básica del programa, expectativas e intereses que se vinculen a su interés particular por estudiar el programa de formación.

Describa de manera sencilla y clara utilizando frases, vocabulario e imágenes relacionadas con temas de interés y características de su programa de formación.

Lineamientos para la entrega de la evidencia:

- **Producto a entregar:** folleto. Haga uso de las normas gramaticales y redacción utilizando el idioma inglés, además, aplique las normas APA para las referencias y citación de información obtenida de diferentes recursos digitales.
- **Extensión:** folleto tríptico entre 70 y 150 palabras.
- **Formato:** el folleto deberá exportarlo a PDF.
- Para hacer el envío del documento remítase al área de la actividad correspondiente y acceda al espacio para el envío de la evidencia: **Folleto GA1-240202501-AA1-EV03.**

4. ACTIVIDADES DE EVALUACIÓN

Evidencias de aprendizaje	Criterios de evaluación	Técnicas e instrumentos de evaluación
Evidencias de conocimiento: Infografía sobre la teoría general de sistemas **GA1-220501092-AA1-EV01** Evidencia de conocimiento: Identificación de procesos organizacionales. **GA1-220501092-AA1-EV02** Evidencia de producto: Mapa de procesos del *software* a construir. **GA1-220501092-AA1-EV03**	Identifica procesos de la organización de acuerdo con la estructura organizacional de la empresa y los requerimientos del cliente. Aplica técnicas de análisis de procesos, siguiendo la metodología establecida. Elabora diagrama de procesos identificando áreas de incidencia directa con el sistema de información a construir. Reconoce las fronteras y el contexto del sistema de acuerdo con el alcance.	**IE-GA1-220501092-AA1-EV01** Lista de chequeo **IE-GA1-220501092-AA1-EV02** Lista de chequeo **IE-GA1-220501092-AA1-EV03** Lista de chequeo
Evidencia de conocimiento: Mapa mental sobre ingeniería de requisitos. **GA1-220501092-AA2-EV01** Evidencias de desempeño: Foro temático. Fuentes de requisitos. **GA1-220501092-AA2-EV02**	Identifica las fronteras del sistema y del contexto de acuerdo de las fuentes de requisitos. Reconoce las fuentes de requisitos de acuerdo con el proyecto. Diferencia los tipos de requisitos según sus características particulares.	**IE-GA1-220501092-AA2-EV01** Lista de chequeo **IE-GA1-220501092-AA2-EV02** Lista de chequeo

Evidencia de desempeño: Diseño del instrumento de recolección de información. **GA1-220501092-AA3-EV01** Evidencia de desempeño: Formulación del proyecto de *software*. **GA1-220501092-AA3-EV02** Evidencia de producto: Formulario de recolección de información. **GA1- 220501092-AA3-EV03**	Diseña instrumentos para recolección de información siguiendo normas y procedimientos técnicos. Utiliza las técnicas de captura de requisitos de acuerdo con las fuentes Identificadas. Organiza la información recolectada para analizarla.	**IE-GA1-220501092-AA3-EV01** Lista de chequeo **IE-GA1-220501092-AA3-EV02** Lista de chequeo **IE-GA1-220501092-AA3-EV03** Lista de chequeo
Evidencias de Desempeño: Especificación de los requerimientos funcionales y no funcionales del *software.* **GA1-220501092-AA4-EV01** Evidencia de producto: Documento con especificación de requerimientos. **GA1-220501092-AA4-EV02**	Genera la documentación de la especificación de requisitos de acuerdo con normatividad y estándares relacionados. Presenta el informe de requisitos de acuerdo con estándares establecidos.	**IE-GA1-220501092-AA4-EV01** Lista de chequeo **IE-GA1-220501092-AA4-EV02** Lista de chequeo
Evidencias de conocimiento: Taller para la determinación de las especificaciones funcionales del software y metodología a utilizar. **GA1-220501092-AA5-EV01** Evidencia de producto: Informe de evaluación de los requerimientos. **GA1-220501092-AA5-EV02**	Evalúa el informe de requisitos con el cliente según las necesidades establecidas. Realiza cambios a la documentación de especificación de requisitos a partir de los hallazgos encontrados.	**IE-GA1-220501092-AA5-EV01** Lista de chequeo **IE- GA1-220501092-AA5-EV02** Lista de chequeo

Evidencias de conocimiento: Taller sobre metodologías de desarrollo de *software*. **GA1-220501093-AA1-EV01** Evidencias de conocimiento: Infografía sobre metodologías de desarrollo de *software*. **GA1-220501093-AA1-EV02** Evidencia de desempeño: Foro. Especificación de la metodología a aplicar. **GA1-220501093-AA1-EV03** Evidencia de producto: Documento identificando la metodología para el proyecto de desarrollo de *software*. **GA1-220501093-AA1-EV04**	Identifica metodologías de desarrollo de *software* de acuerdo con las características del *software* a desarrollar. Establece las actividades de análisis de acuerdo con la metodología seleccionada.	**IE-GA1-220501093-AA1-EV01** Lista de chequeo **IE-GA1-220501093-AA1-EV02** Lista de chequeo **IE-GA1-220501093-AA1-EV03** Lista de chequeo **IE-GA1-220501093-AA1-EV04** Lista de chequeo
Evidencia de producto: Mapa conceptual - *Software* y servicios de internet. Relacionar correctamente los tipos de *software* y servicios de internet. **GA1-220501046-AA1-EV01**	Identifica equipos TIC, tipos de *software* y servicios de internet, de acuerdo con las necesidades de uso. Compara equipos TIC, tipos de *software* y servicios de internet, de acuerdo con las características. Escoge equipos TIC, tipos de *software* y servicios de internet, de acuerdo con las necesidades de procesamiento de información y características.	**IE-GA1-220501046-AA1-EV01** Lista de verificación

Evidencia de conocimiento: Taller - Utilización de las herramientas de ofimática. Realizar un taller práctico con las herramientas ofimáticas. **GA1-220501046-AA2-EV01**	Maneja computadores, tabletas, celulares y otros equipos TIC, de acuerdo con las funcionalidades de los mismos. Aplica funcionalidades de sistemas operativos, de acuerdo con las necesidades de administración de los recursos del equipo. Maneja procesador de texto, hoja de cálculo, *software* para presentaciones, diagramación, bases de datos y *software* específico, de acuerdo con las funcionalidades de cada programa y las necesidades de manejo de la información. Utiliza motores de búsqueda, navegación, correo electrónico, transferencia de archivos, chat, listas de correos, blogs, wikis, foros web, spaces, grupos de noticias, telefonía IP, televisión IP, comprar en internet, E-learning, marketing digital, trabajo colaborativo, computación en la nube, redes sociales y videoconferencia por internet, de acuerdo con las necesidades de información y comunicación. Participa en redes sociales, de acuerdo con las necesidades de comunicación. Maneja herramientas colaborativas en internet, de acuerdo con las necesidades del equipo de trabajo. Comprueba el funcionamiento de los equipos, productos o servicios obtenidos con el uso de herramientas TIC, de acuerdo con los resultados esperados.	**IE-GA1-220501046-AA2-EV01** Lista de verificación
Evidencia de desempeño: Informe mejora de productos y procesos con la incorporación de TIC. **GA1-220501046-AA3-EV01**	Aplica procesos de mejora a los productos, de acuerdo con las comprobaciones realizadas.	**IE-GA1-220501046-AA3-EV01** Lista de verificación

Evidencia de conocimiento: Cuestionario. **GA1-240202501-AA1-EV01**	Reconoce la idea general y detalles específicos en interacciones orales de la vida cotidiana articuladas con claridad y una velocidad promedio.	**IE-GA1-240202501-AA1-EV01** Cuestionario
Evidencia de desempeño: Video presentación. **GA1-240202501-AA1-EV02**	Reconoce la idea general y detalles específicos en interacciones orales de la vida cotidiana articuladas con claridad y una velocidad promedio	**IE-GA1-240202501-AA1-EV02** Lista de verificación
Evidencia de producto: Folleto. **GA1-240202501-AA1-EV03**	Reconoce la idea general y detalles específicos en textos escritos de la vida cotidiana articulados con claridad.	**IE-GA1-240202501-AA1-EV03** Lista de verificación

5. GLOSARIO DE TÉRMINOS

Cláusulas adverbiales: una cláusula adverbial es un grupo de palabras que desempeña el papel de un adverbio. (Como todas las cláusulas, una cláusula adverbial contiene un sujeto y un verbo).

Diagrama de flujo: es un esquema de los pasos separados de un proceso en orden secuencial, que se puede adaptar para una amplia variedad de propósitos y se puede utilizar para describir varios procesos, como un proceso de fabricación, un proceso administrativo o de servicio o un plan de proyecto. Es una herramienta común de análisis de procesos.

Gerundio: es una forma del verbo finalizada en “ing” que funciona como un sustantivo y tiene relación con actividades finalizadas o concretas.

***Hardware*:** corresponde a todas las partes físicas y tangibles de una computadora: sus componentes eléctricos, electrónicos, electromecánicos y mecánicos, sus cables, gabinetes o cajas, periféricos de todo tipo y cualquier otro elemento físico involucrado

Infinitivo: Es una forma del verbo antecedida por “to” que funciona como un sustantivo y tiene relación con futuras o abstractas.

Metodología: ciencia que consta de métodos y técnicas, que se aplican sistemáticamente durante un proceso de investigación o para solucionar una problemática.

Requerimientos: es una descripción completa del comportamiento del sistema que se va a desarrollar. Incluye un conjunto de casos de uso que describe todas las interacciones que tendrán los usuarios con el *software*.

Requisitos de usuarios: necesidades que los usuarios expresan verbalmente

Requisitos del sistema: son los componentes que el sistema debe tener para realizar determinadas tareas

Requisitos funcionales: servicios que el sistema debe proporcionar al finalizar el sistema

Stakeholders: interesados o participantes en un proyecto.

***Software*:** soporte lógico, programas, parte no mecánica de un sistema. Serie de instrucciones necesarias para ejecutar diversas aplicaciones y tareas.

Teoría general de sistemas: es un esfuerzo de estudio interdisciplinario que trata de encontrar las propiedades comunes a entidades, los sistemas, que se presentan en todos los niveles de la realidad, pero que son objetivo tradicionalmente de disciplinas académicas diferentes**.**

Verbo transitivo: es un verbo que requiere del uso de un objeto directo sobre el que recae la acción.

Verbo intransitivo: es un verbo que no requiere del uso de un objeto directo.

Verbos compuestos: son verbos que contienen una o más partículas que hacen que el significado del verbo cambie ligera o totalmente.

6. REFERENTES BIBLIOGRÁFICOS

Andrade, A. M., Del Río, C. A., y Alvear, D. L. (2019). A study on time and motion to increase the efficiency of a shoe manufacturing company | Estudio de Tiempos y Movimientos para Incrementar la Eficiencia en una Empresa de Producción de Calzado. *Información Tecnológica, 30*(3), 83–94. https://doi.org/10.4067/S0718-07642019000300083

De Guevara, D. M. Á. L. (2014). *Sistema operativo, búsqueda de la información: Internet/Intranet y correo electrónico. UF0319*. Tutor Formación. Figarola, I. (s.f.). *Fonética inglesa (th).* https://www.abaenglish.com/es/fonetica-inglesa/th/

Fresno, C., C. (2018). *¿Cómo funciona internet?* El Cid Editor. https://elibronet.bdigital.sena.edu.co/es/ereader/senavirtual/36728?page=34

Gallardo, Y. (2020). *Word para principiantes – 2020.* [Video]. YouTube https://www.youtube.com/watch?v=4ooZlyprmc

Gaskin, S., Vargas, A., & Martin, C. L. (2011). *Go! with Microsoft Word 2013, Introductory*. Langara College.

Gómez de Silva, G., A., y Briseño, A. (2008). *Software* (pp. 23–44). Cengage Learning. https://link.gale.com/apps/doc/CX3004400004/GVRL?u=sena&sid=GVRL&xid=d8990326

Ibarra, S., J. I. (2013). *Manual sistema operativo, búsqueda de la información: Internet/intranet y correo electrónico*. Editorial CEP, S.L. https://elibro-net.bdigital.sena.edu.co/es/ereader/senavirtual/50724?page=19

Ibiza, D. (2019). *Tutorial Trello: Guía de uso con ejemplos reales prácticos.* [Video]. YouTube. https://www.youtube.com/watch?v=_UB44coH3SM&feature=youtu.be

Ladrón de Guevara, M. (2018). *Sistema operativo, búsqueda de la información: internet/intranet y correo electrónico UF0319*. Editorial Tutor Formación. https://elibro-net.bdigital.sena.edu.co/es/lc/senavirtual/titulos/44263

Moure, O.(1999). *El acento en las palabras de dos sílabas.* http://www.ompersonal.com.ar/ompronounce/unit11/page1.htm

Naranjo, G., M. R. (2010). *Manual Ofimática básica en formación continua. Formación para el empleo.* Editorial CEP, S.L https://elibro-net.bdigital.sena.edu.co/es/ereader/senavirtual/50987?page=1

Pressman, R. (2010). *Ingeniería del software, un enfoque práctico.* McGraw-Hill.

Real Academia Española. (1970). *Diccionario de la lengua española* (Vol. 19). Espasa-Calpe.

Systems, V. (2013). *Inglés: grado superior.* McGraw-Hill https://elibro-net.bdigital.sena.edu.co/es/ereader/senavirtual/50221?page=1

Valentín, L., G.M. (2015). *Ofimática.* Editorial CEP, S. L. https://elibronet.bdigital.sena.edu.co/es/ereader/senavirtual/51049?page=16

7. CONTROL DEL DOCUMENTO

	Nombre	Cargo	Dependencia	Fecha
Autor (es)	Ramiro Barrientos Gómez	Experto temático	Centro de Formación en Diseño, Confección y Moda	Noviembre 2020
	Sergio Arturo Medina Castillo	Diseñador Instruccional	Centro para la Industria de la Comunicación Gráfica	Noviembre 2020
	Liliana Victoria Morales	Evaluadora Instruccional	Centro para la Industria de la Comunicación Gráfica	Noviembre 2020
	Elkin Rodolfo Moreno Merchan	Experto temático	Regional Distrito Capital - Centro de Formación de Talento Humano en Salud	Febrero 2021
	Jonathan Guerrero Astaiza	Experto temático	Centro de Teleinformática y Producción Industrial - Regional Cauca	Noviembre 2021
	Zulema Leon Escobar	Experta temática	Centro de Teleinformática y Producción Industrial - Regional Cauca	Noviembre 2021

	Oscar Absalon Guevara	Diseñador Instruccional	Centro de Gestión Industrial - Regional Bogotá	Febrero 2021
	Deivis Eduard Ramirez Martinez	Diseñador Instruccional	Centro para la Industria de la Comunicación Gráfica - Regional Distrito Capital -	Noviembre 2021
	Silvia Milena Sequeda Cárdenas	Evaluadora instruccional	Centro de gestión industrial - Distrito capital.	Noviembre 2021
	Rafael Neftalí Lizcano Reyes	Asesor Pedagógico	Centro Industrial del Diseño y la Manufactura – Regional Santander	Noviembre 2021
	José Gabriel Ortiz Abella	Corrector de estilo	Regional Distrito Capital – Centro para la Industria de la Comunicación Gráfica.	Noviembre del 2021.

8. CONTROL DE CAMBIOS

	Nombre	**Cargo**	**Dependencia**	**Fecha**	**Razón del Cambio**
Autor (es)					

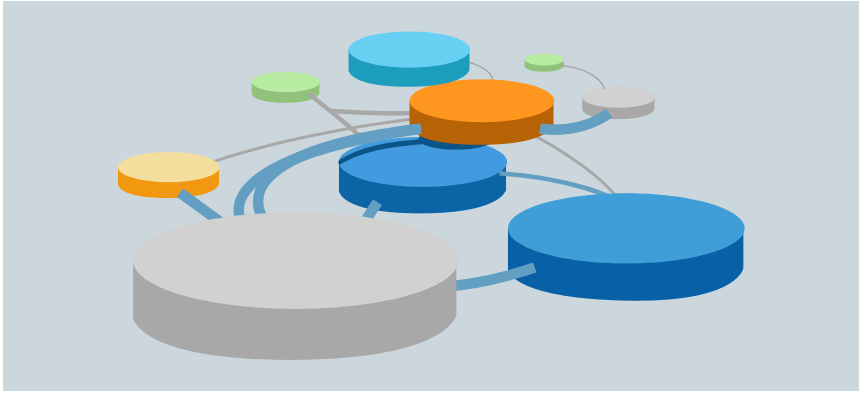
Teoría general de sistemas



ɔ de la ciencia que pretende examinar las propiedades que definen a los sistemas; es
ɔrías formadas por partes interrelacionadas que llevan al cumplimiento de una acción

ɲto de vista histórico las primeras contribuciones a este campo y su definición vienen
ɡeneral de sistemas de Ludwig von Bertalanffy. Este modelo ha perdurado como
para la aplicación del método científico y sigue siendo utilizado como apoyo
en el análisis de sistemas, organizaciones incluso para grupos humanos entre otros

Desde la biología Karl Ludwig von Bertalanffy (1968) "propone la teoría general de sistemas como un instrumento amplio que se aplica y es común su uso para la resolución de problemas hoy en día y en todas las áreas del saber" an 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; el pensamiento de Bertalanffy presenta objeciones a tales creencias.





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 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:

"Un marco común para los procesos del ciclo de vida de los programas informáticos, con una terminología bien definida, a la que pueda remitirse la industria del software. **Contiene procesos, actividades y tareas aplicables durante la adquisición, el suministro, el desarrollo, el funcionamiento, el mantenimiento o la eliminación de sistemas, productos y servicios informáticos.** Estos procesos del ciclo de vida se llevan a cabo mediante la participación de los interesados, con el objetivo final de lograr la satisfacción del cliente (s.p.)".

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



Fases:

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



Entregables:

Son los productos intermedios que generan las fases. Pueden ser materiales o inmateriales (documentos, software). Los entregables permiten evaluar la marcha del proyecto mediante

comprobaciones de su adecuación o no a los requisitos funcionales y de condiciones de realización previamente establecido.

1.1 Fases

Las fases del modelo de ciclo del software son: planificación, análisis, diseño, implementación, pruebas y mantenimiento, las cuales se describen en la siguiente figura.





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 (definició

Esta fase busca definir los requisitos que son los que dirig software.

Objetivos: conocer los requisitos, asegurar que los requi acuerdo con el cliente.

1/5

1.2. Paradigmas de los modelos de ciclo de vida del software

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

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

Según la norma 1074 IEEE se define al ciclo de vida del software como:

Una aproximación lógica a la adquisición, el suministro, el desarrollo, la explotación y el mantenimiento del software

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

Un marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de software, abarcando la vida del sistema desde la definición de los requisitos hasta la finalización de su uso (2008).

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



Paradigma tradicional

Los paradigmas tradicionales se identifican, fundamentalmente, por ser lineales, es decir se trata de completar cada proceso de principio a fin hasta que quede listo para avanzar a la segunda fase del ciclo del software.

Si bien es verdad que las metodologías actuales están basadas en lo que fueron los paradigmas tradicionales, hoy en día se ha evolucionado, sin embargo, los paradigmas tradicionales se mantienen.



Desventaja:

Pérdida de tiempo si se encuentran errores en una fase avanzada porque al devolverse se debe pasar nuevamente por todas las fases y reestructurar de acuerdo con las modificaciones.



Paradigma orientado a objetos:

Las etapas de desarrollo de software en el paradigma orientado a objetos, se conforma principalmente por la creación de clases, análisis de requisitos y el diseño. Con este paradigma se pretende que el código fuente sea reutilizable para otros proyectos.



Paradigma de desarrollo ágil

El objetivo de este paradigma es el desarrollo de proyectos en poco tiempo, se simplifican procesos tediosos, se agilizan las fases del desarrollo y las interacciones se hacen en corto tiempo.

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

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

Modelo en cascada

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

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

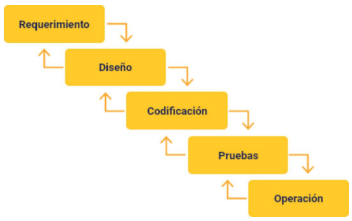


Figura 1. Modelo de cascada del ciclo de vida del desarrollo.

Modelo espiral

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

Modelo Kanban

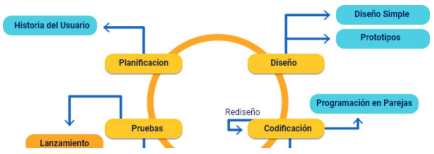
Modelo XP o programación extrema

La programación extrema o eXtreme Programming (XP) es un enfoque de la ingeniería de software formulado por Kent Beck, autor del primer libro sobre este tema: Extreme Programming Explained: Embrace Change (1999). Esta metodología es adaptable según las necesidades y requerimientos a implementar, además, el cliente se encuentra involucrado en el proceso de desarrollo lo que hace que el producto pueda ser terminado en un menor tiempo.

Características principales de la programación extrema:

1. Tipo de desarrollo iterativo e incremental.
2. Pruebas unitarias.
3. Trabajo en Equipo.
4. Trabajo junto al cliente.
5. Corrección de errores.
6. Reestructuración del código.
7. El código es de todos.
8. El código simple es la clave.

Programación Extrema (XP).



1 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.

1.1. 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.

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

Tabla 1/ Tareas para elicitación de requisitos.

Nota: tomado de Durán y Bernárdez (2001)

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

A. 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.

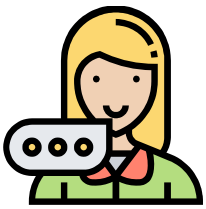


En el proceso de esta actividad se identifican:

- 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.



B. 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

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.

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.

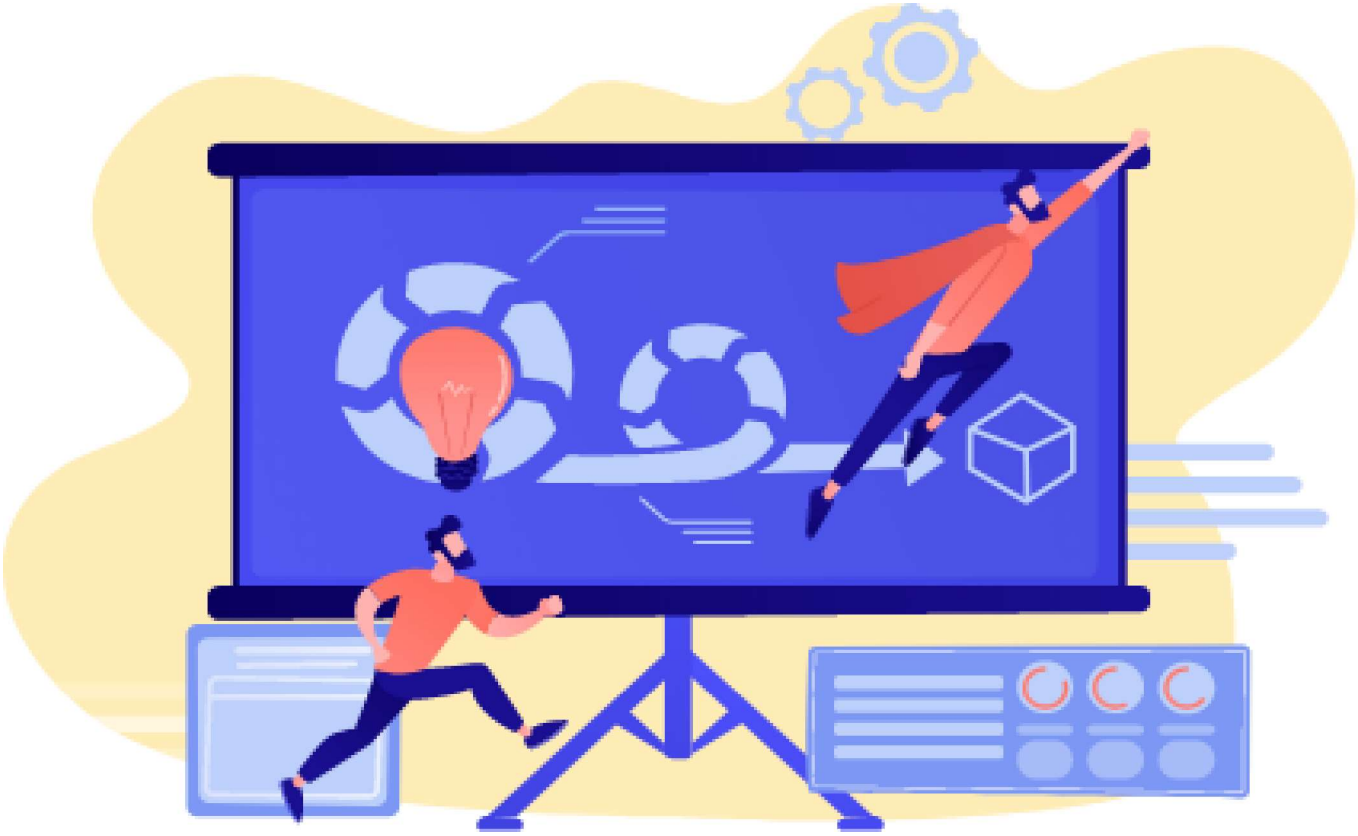
Tabla 2/ Roles involucrados en la ingeniería de requisitos.

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).

C. 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.



Figura 2 / Entradas, herramientas y técnicas, salidas.

Anexo. Para profundizar en detalle, se invita a consultar la Guía PMBOK 6 - 49 procesos, entradas, herramientas y salidas, como 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.

- **Stakeholder**
- **Tipo**
- **Objetivo o resultados**
- **Acciones posibles con impacto positivo / negativo**
- **Estrategias**
- **Conclusiones**

Categorización de stakeholders y estrategias de gestión de las expectativas.

Como ya se había mencionado anteriormente, la matriz de stakholders 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 2).



Figura 2 / Ejemplo de matriz interés vs. influencia.

Nota: tomado de Gardnet et al. (1986)

1.2 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.

1.2.1. 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 puede observar en el siguiente gráfico.



1. 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.



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.

Anexo. Entrevista estructurada.

Entrevista semiestructurada

Entrevista no estructurada

1.2.2. 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).

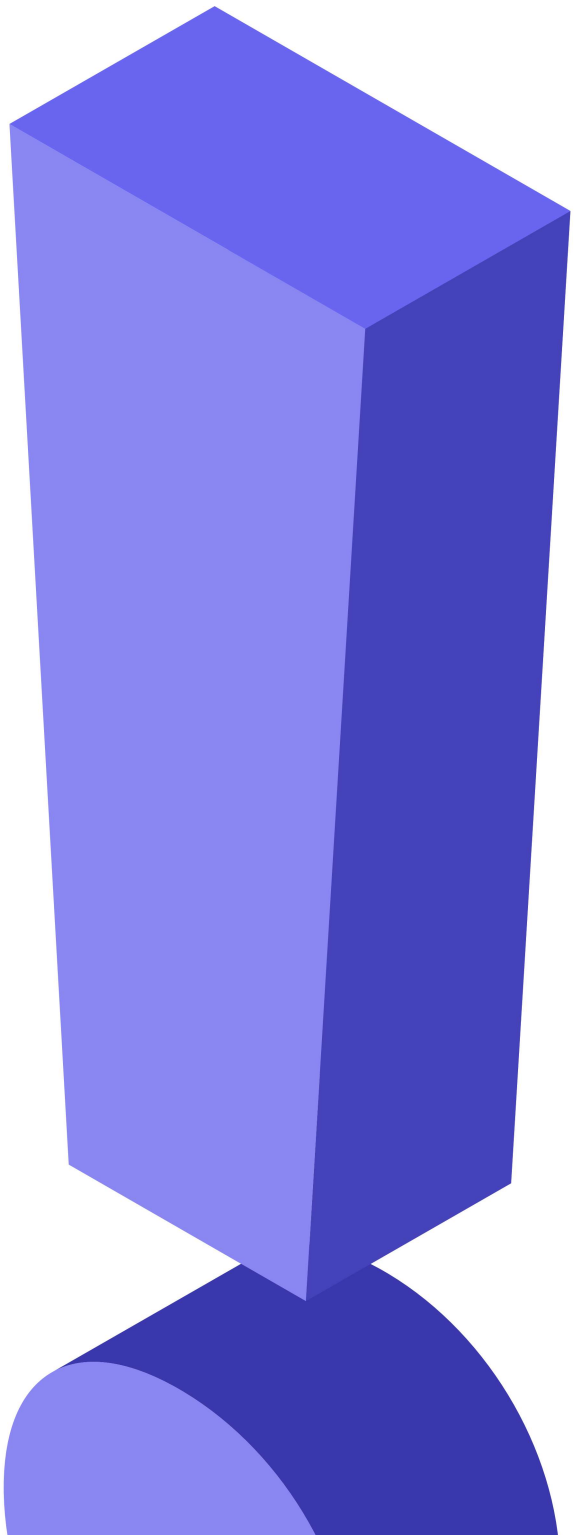
Pueden variar en cuanto a propósito, diseño y apariencia, y consisten en listas de preguntas escritas. Los individuos participantes en la investigación suelen leer los mismos listados de preguntas, por lo que esto permite consistencia y precisión al analizar las respuestas, además de facilitar el proceso. 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).

Anexo. Tipos de preguntas.

1.2.3. Observación.

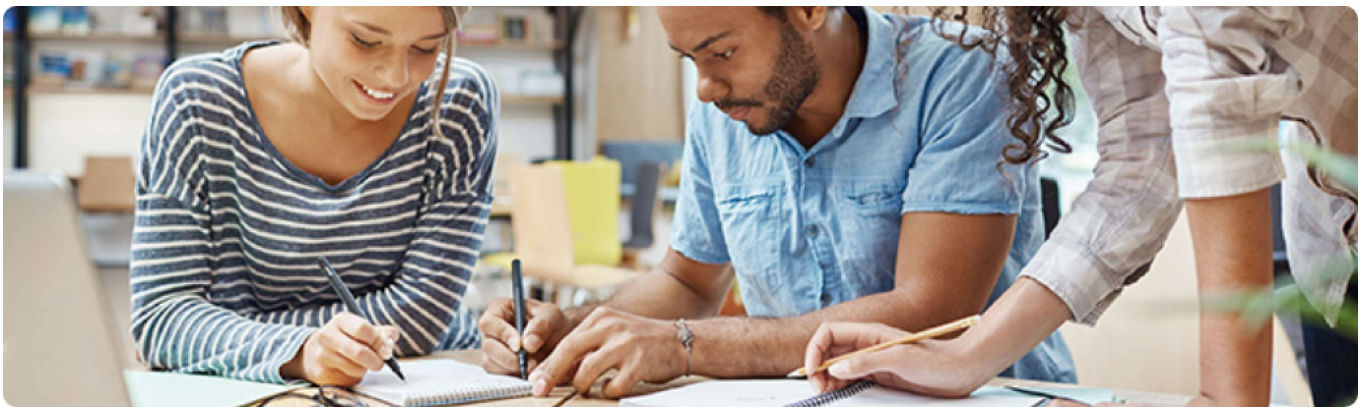
Esta permite la obtención de datos para emprender una investigación de tipo cualitativo, no desde el punto de vista de lo que los sujetos dicen, sino que es la evidencia directa de lo que ve y percibe el investigador en un escenario de primera mano (Denscombe, 2010).

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:

1. Debe servir a un objeto formulado de investigación.
2. Debe de ser planificada sistemáticamente.
3. Debe estar controlada y relacionada con proposiciones generales.
4. 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.
- Permite a los sujetos y objetos establecer relaciones de manera directa.



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 3 / 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:

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

Anexo. Ficha de observación.

1.2.4. 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



Sesiones JAD (Joint Application Design)

Método Delphi

1.3. 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:

1.3.1. 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.

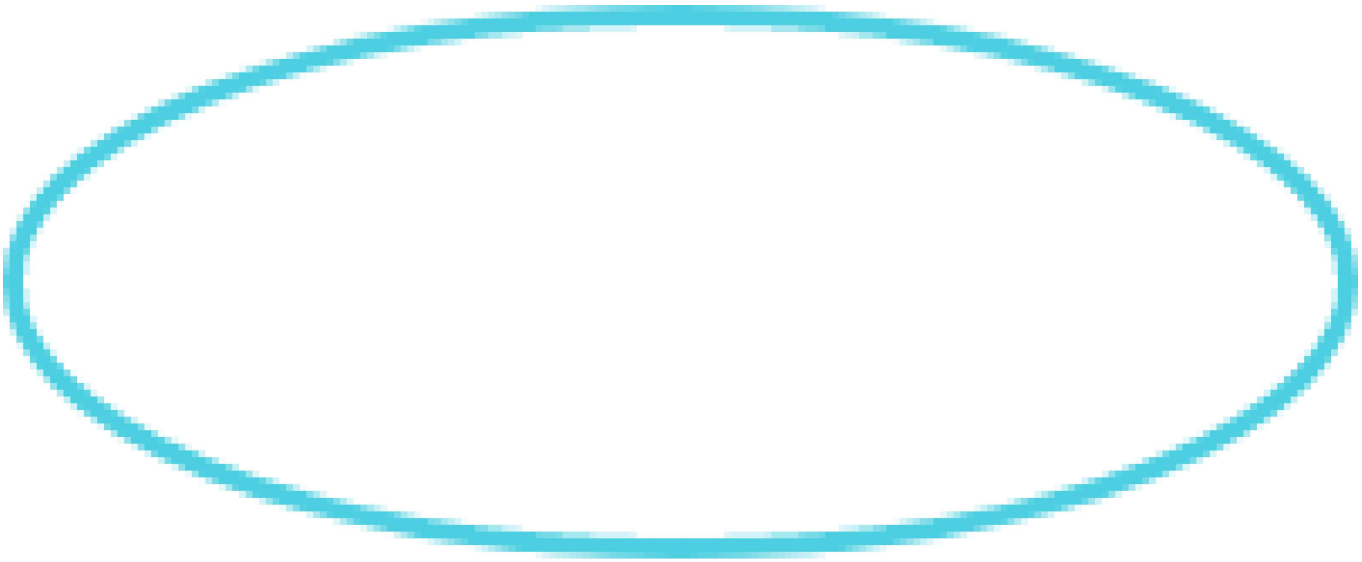
Actor

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.



Caso de uso:

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.

Representación gráfica

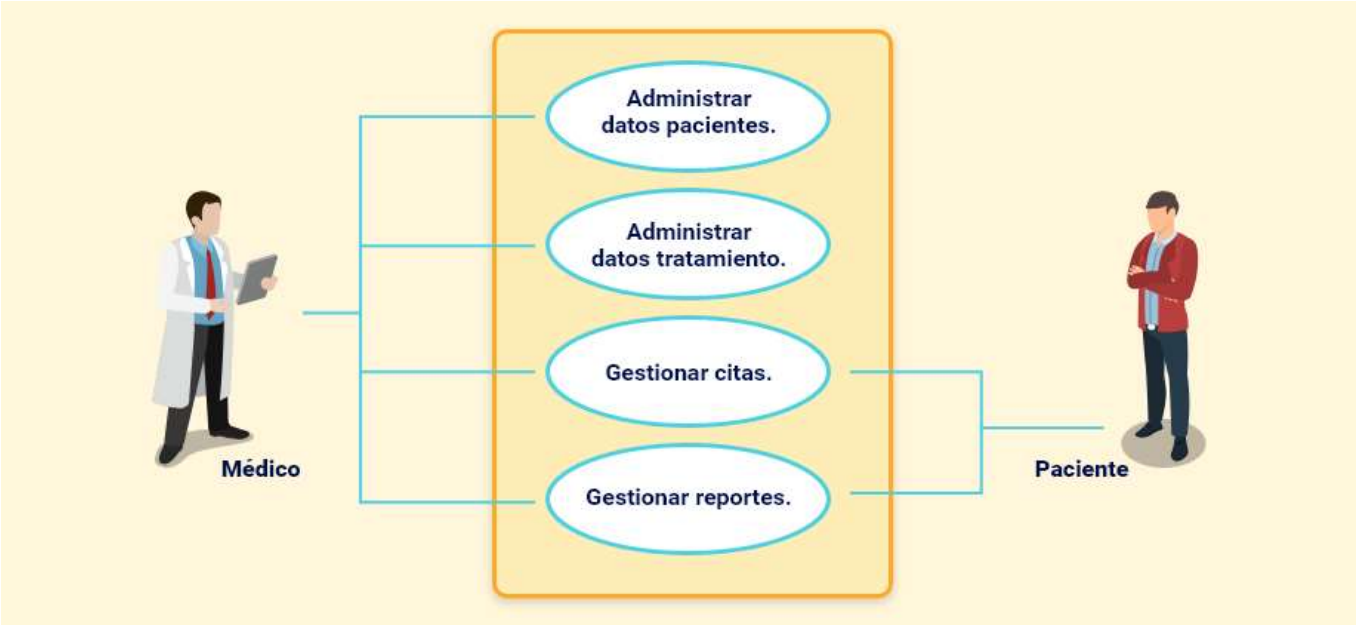


Figura 7 / Caso de uso centro médico.

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. A continuación, se presenta el formato diligenciado de acuerdo con el ejemplo del centro médico.



Documentación caso de uso

A continuación, se presenta el formato diligenciado de acuerdo con el ejemplo del centro médico.

Descargar

Nota: tomado de Gutierrez (s.f.)

1.3.2. Historias de usuario.

Las historias de usuario son utilizadas en los métodos agiles 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:

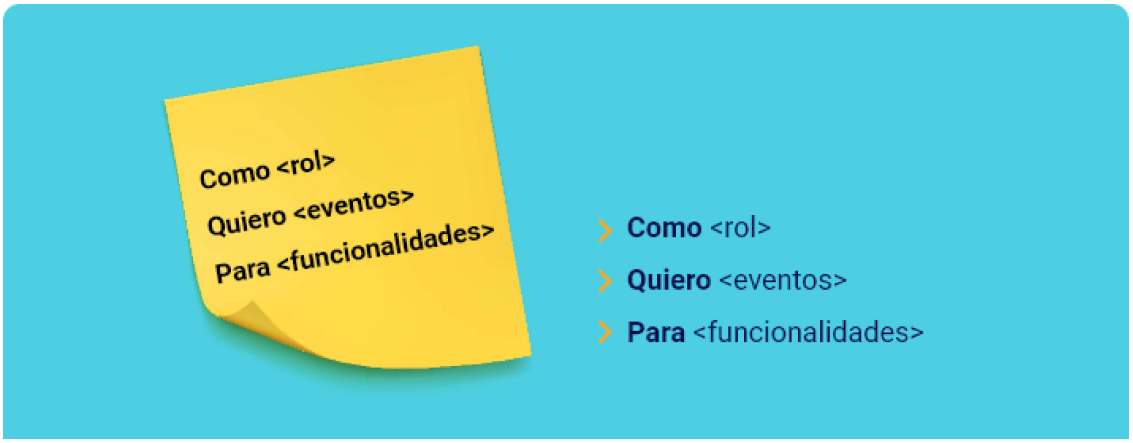


Figura 8 /Regla de tres palabras.

Nota: tomado de Martin (s.f.)

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.

En las siguientes figuras se presentan ejemplos de historias de usuario.



Figura 9 /Ejemplos historias de usuario.

Nota: tomado de fatto www.fattocs.com http://i.ytimg.com/vi/Zi9E1aUO_1U/hqdefault.jpg

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.

A continuación, un ejemplo de una historia de usuario usando plantilla.

Enunciado de la historia					Proceso		
Identificador (ID) de la historia	Rol	características / Funcionalidad	Razón / Resultado	Número (#) de escenario	Criterio de aceptación (Titulo)	Contexto	Evento

Figura 10 / Ejemplos de historias de usuario.

Anexo. Revisar el anexo Plantilla de Historias de Usuario, para analizar su estructura.

Nota: Tomado de la oficina de proyectos de informática. (2012)

1.3.3. 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.



Las dos figuras siguientes muestran el ejemplo de un storyboard que representa un escenario de situación de vendedores de una empresa para explicar el cambio que sufrirá el trabajo, el primero representa la situación actual y el segundo como quedará con la implantación del sistema.

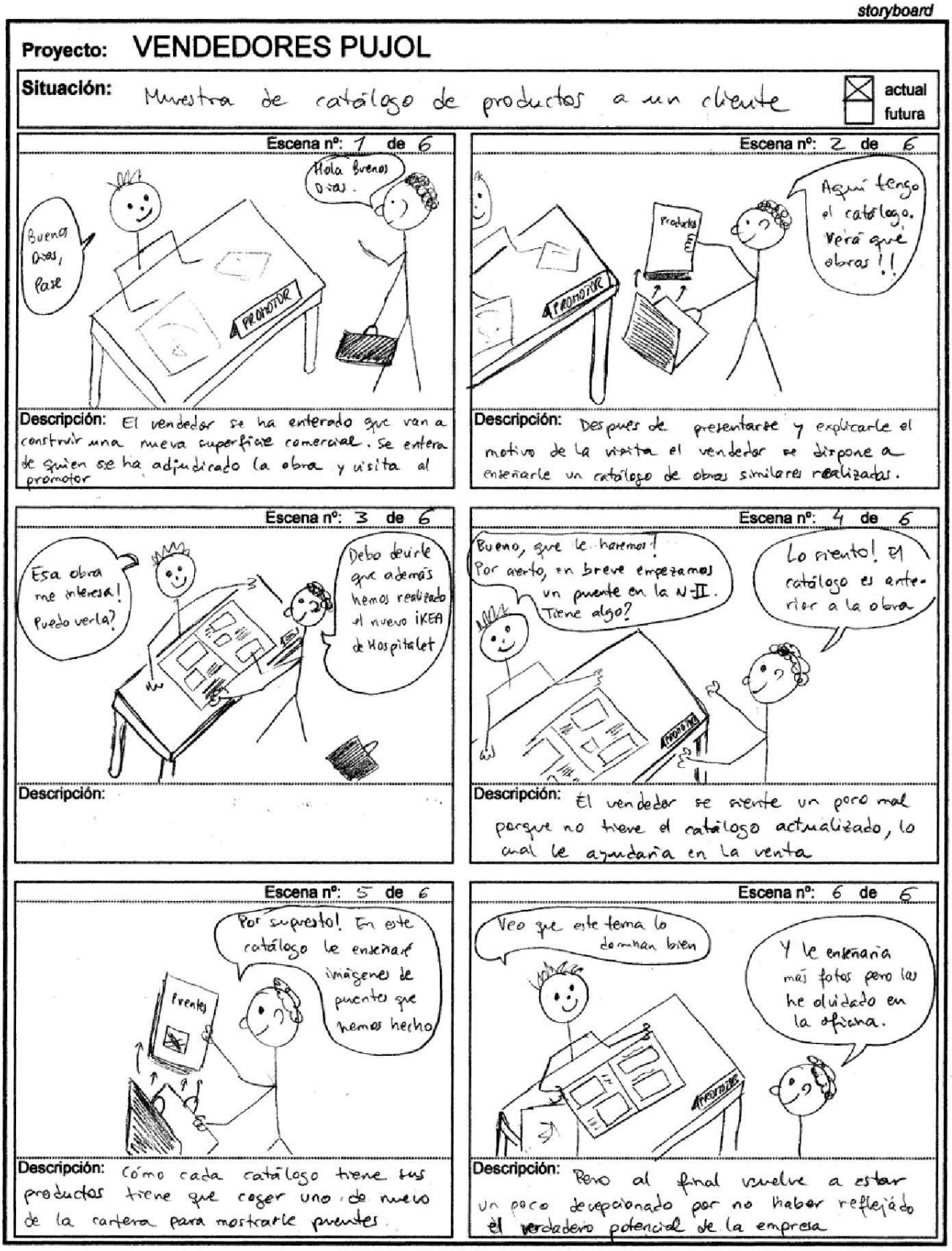


Figura 11 /Escenario representado en formato de storyboard que representa una situación típica tal y como se realiza actualmente.

Nota: Tomado de Granollers, Lorés y Perdrix (2002)

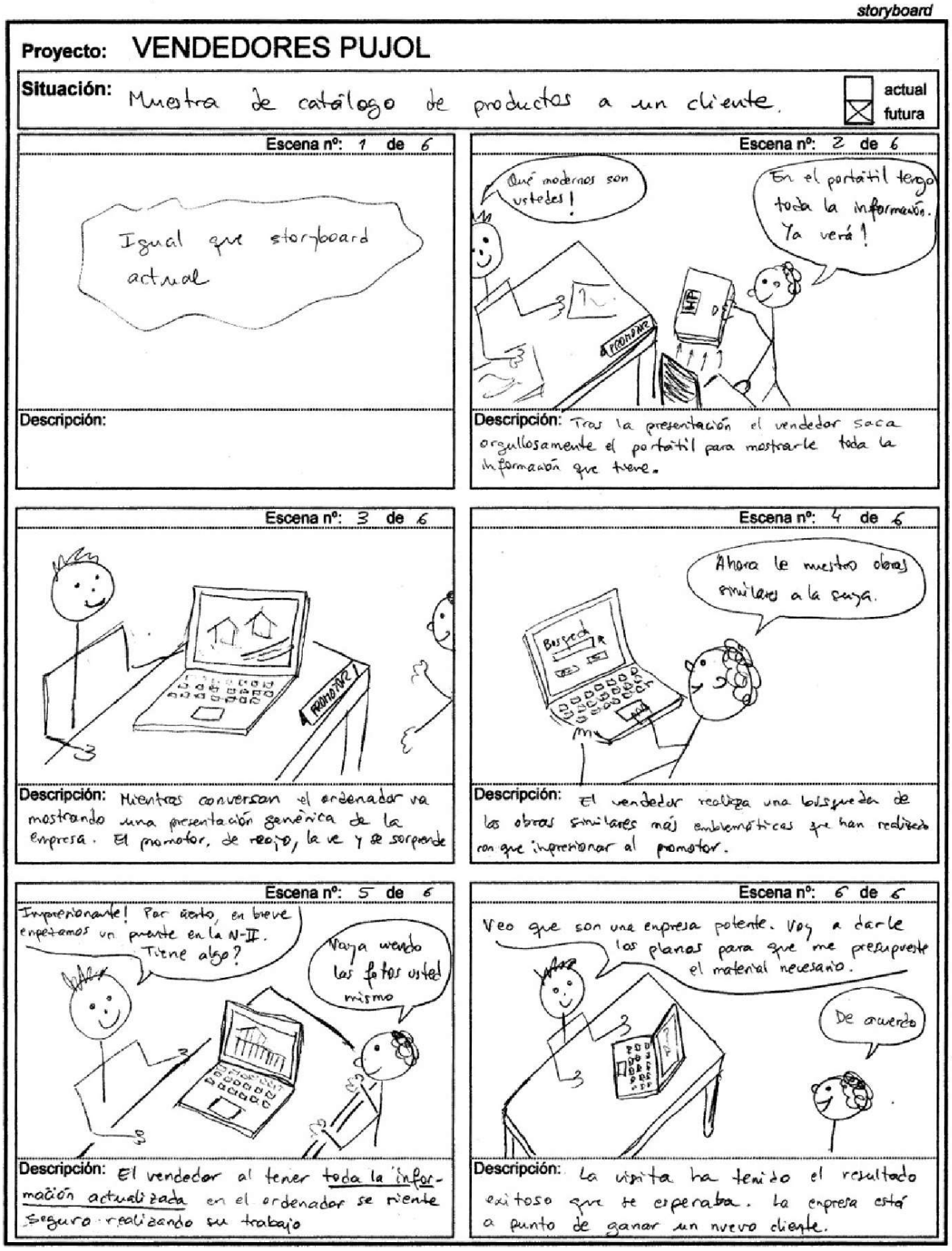


Figura 12 /Escenario representado en formato de storyboard que representa la misma situación anterior tal como quedará con la implementación del sistema.

Nota: Tomado de Granollers, Lorés y Perdrix (2002)

1.4. Herramientas de modelado

Estas herramientas permiten crear un "simulacro" del sistema.

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 (https://www.gliffy.com/).
- MagicDraw.
- Lucidchart.
- Papyrus Uml.
- Modelio.
- StarUml.
- Dia.
- Mono Uml.

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

Gliffy
ArgoUML
MagicDraw
StarUML
Lucidchart

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.

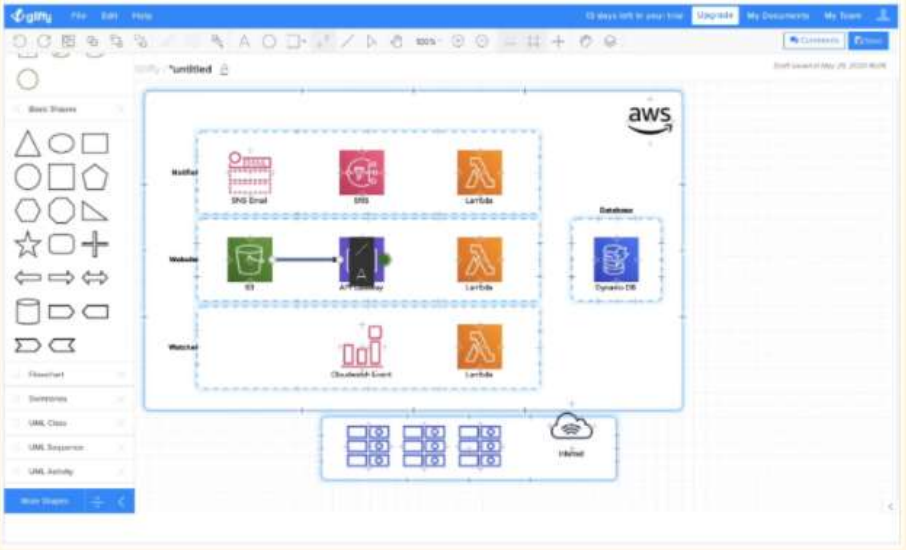


Figura 13 /Herramienta Gliffy.

Nota: Tomado de https://www.gliffy.com/

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.



1.1 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.

1.1.1 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.

1.1.2 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 donde 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 1 - 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.

1.1.3 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 2 - Tabla de referencia para la técnica urgente | Nota: Tomado de Porfirio (2021)

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

Luego se consideran los requerimientos de mayor prioridad que están en el sector de color rojo, luego los de color naranja, luego 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 3 - Ejemplo de aplicación técnica urgente

Requerimientos	Valor del negocio	Urgencia	Sector
R01	1	4	Verde
R02	2	4	Amarillo

Requerimientos	Valor del negocio	Urgencia	Sector
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, 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.

1.1.4 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.**



1.1.5 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.

1.1.6 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 4 - Ejemplo de aplicación técnica de matriz de priorización

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

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.

1.2 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).

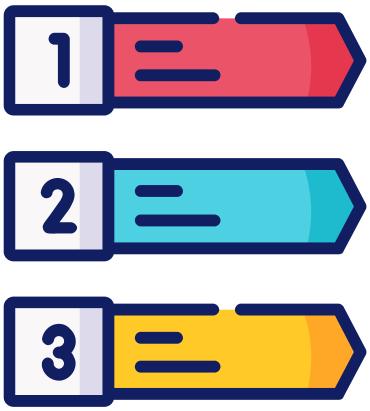


Este es un instrumento clave para el responsable del proyecto, en especial en el proceso de seguimiento y control de cambios, ya que permite analizar qué requerimientos, eventualmente, convendría modificar, eliminar o añadir; así, se puede ajustar la planificación de tareas pendientes del proyecto. También 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 la siguiente figura se propone el contenido base de una matriz de correlación dividida por secciones.

Identificación
Estado
Objetivo



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)

Figura 1 - Contenido base matriz correlación.

A continuación, se muestra el ejemplo de la estructura en Excel de una matriz de trazabilidad.

Tabla 5 - Ejemplo de la estructura de una matriz de trazabilidad.

IDENTIFICACIÓN				ESTADO				CARACTI	
Id	Código jerárquico	Descripción	Tipo	Versión	Estado	Fecha de estado	Responsable	Prioridad	Comp



Ejemplo de la estructura de una matriz de trazabilidad

Documento XLS

Descargar

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.**

1.3 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.

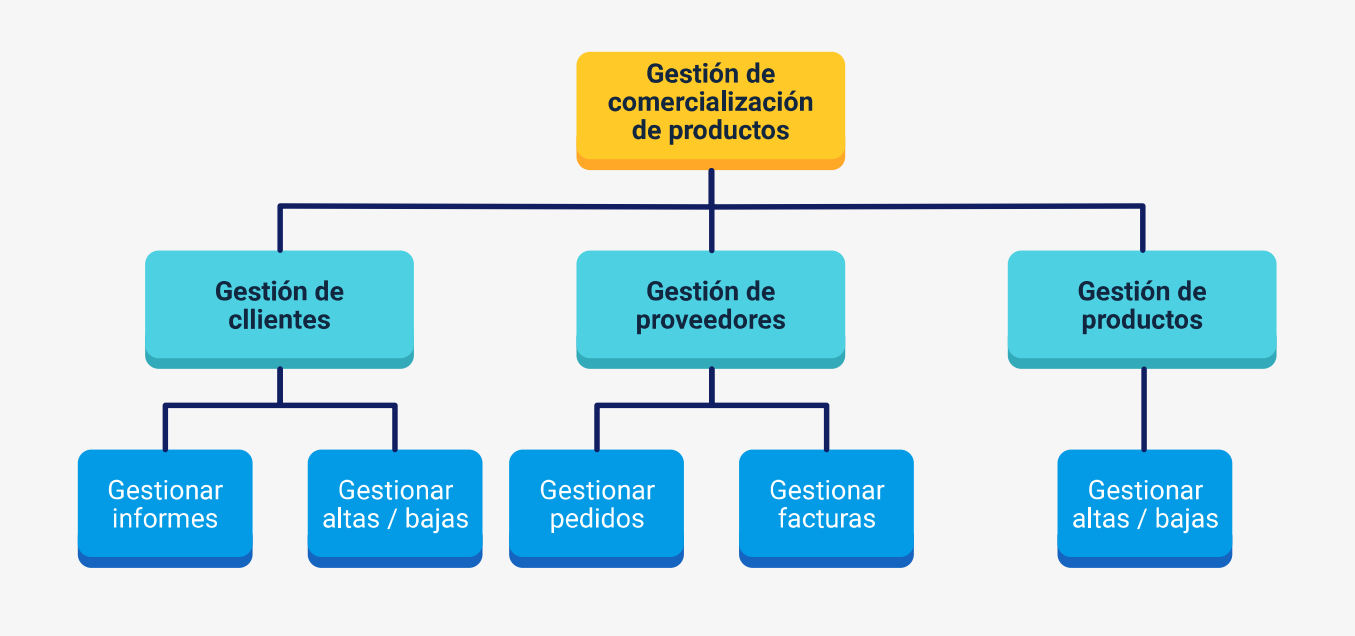


Figura 2 - Ejemplo de descomposición funcional de un sistema de gestión comercialización de productos (fragmento).

1 Validación de requerimientos

La validación de los requerimientos busca ratificar que los requerimientos realmente están especificando lo que el cliente desea y necesita. Este proceso es muy importante, pues un error en un documento de requerimientos puede ocasionar el desgaste importante de muchos recursos si estos errores son detectados en etapas más avanzadas del proyecto como diseño, construcción o despliegue a producción. Los costos asociados para el arreglo de problemas en los requerimientos siempre van a ser menores que en otras etapas, ya que un error en los requerimientos se propaga en cascada en todas las fases subsiguientes del ciclo de vida.

Según Sommerville (2011), en el proceso de validación de requerimientos se llevan a cabo las siguientes verificaciones:

Verificación de validez: los requerimientos son razonables e identifican realmente todas las funciones necesarias para cumplir con las necesidades del cliente.

Verificación de consistencia: los requerimientos no presentan contradicciones.

Verificaciones de completitud: se incluyen todas las funcionalidades y restricciones definidas por los usuarios del sistema.

Verificaciones de realismo: los requerimientos son realizables de acuerdo con la tecnología existente, el presupuesto y los tiempos definidos.

Verificabilidad: es posible demostrar la realización de cada requerimiento y que hace lo que debe hacer. Es decir, existe una forma clara en la que se le pueda realizar pruebas.



Infografía - Validación de requerimientos

La validación de los requerimientos busca ratificar que los requerimientos realmente están especificando lo que el cliente desea y necesita.

Descargar

Existen varias técnicas que pueden usarse para la validación de requisitos, como se puede revisar a continuación.

1.1. Revisiones de requerimientos

Las revisiones de los requerimientos es un proceso manual que involucra la participación de personas de parte de la organización constructora del software así como la de los clientes. Por lo general, en este proceso se revisa el documento de requerimientos tratando de encontrar alguna anomalía y/u omisiones en la escritura de estos.

Esta revisión se puede realizar de manera informal o formal (Sommerville, 2011). En una revisión informal, se requiere verificar, con tantos stakeholders como sea posible, el documento generado para recibir confirmación por parte de ellos de que lo escrito sí refleja su deseo. Esta revisión informal permite, de manera muy sencilla, detectar muchos problemas antes de establecer cualquier formalismo de revisión.

En una revisión formal, el equipo de desarrollo debe introducir al cliente por cada uno de los requerimientos establecidos, y explicando cada una de sus implicaciones. El equipo encargado de la revisión deberá verificar cada requerimiento de forma individual y realizar el análisis de conjunto. Además, en el proceso de verificación debe resolverse cada una de las siguientes preguntas:

Verificabilidad: ¿Es posible probar este requerimiento en la realidad?

Comprensibilidad: ¿Es claro lo que expresa este requerimiento para las personas que lo van a usar?

Rastreabilidad: ¿Es posible observar el origen del requerimiento y también evaluar los cambios que pueden generar en el sistema?

Adaptabilidad: ¿El cambio en el requerimiento impacta o no a gran escala sobre otros requerimientos?

1.2 Construcción de prototipos

Por otra parte, el uso inadecuado de los prototipos también puede generar algunos riesgos, entre los que se encuentran:

El prototipado puede estimular un número excesivo de cambios de peticiones.

Los prototipos operativos pueden inducir a pensar a la directiva y a los clientes que el producto final está prácticamente dispuesto para su salida al mercado.

Los prototipos pueden encarecer el producto.

Para la fase de verificación de requisitos se recomienda el uso de prototipos de baja fidelidad (presentación de escenarios con maquetas estáticas) y de tipo exploratorio (prototipos no reutilizables, usados únicamente para la clarificación e identificación de requerimientos).

Aunque un prototipo podría ser una sencilla representación gráfica del sistema en papel o la representación de interfaces del sistema usando herramientas de ofimática, existen varias herramientas gratuitas y de pago que facilitan la construcción de prototipos, algunas de ellas solo requieren un registro de correo electrónico para acceder a sus funcionalidades directamente desde el navegador, y otras requieren la descarga e instalación en el equipo donde se planean construir los prototipos.

Algunas herramientas recomendadas son las siguientes:



Adobe XD

www.adobe.com/la/products/xd.html



Marvel App

www.marvelapp.com



Moqups

www.moqups.com



LucidChart

www.lucidchart.com/pages/es



Proto.io

www.proto.io



Balsamiq

www.balsamiq.com



Wirify

www.wirify.com



Picodo

www.pidoco.com/en

1.3. Generación de casos de prueba

Desarrollar pruebas para los requerimientos permite evidenciar problemas antes de escribir cualquier línea de código. Normalmente si es complejo construir un caso de prueba para un requerimiento esto es un indicio de que el requerimiento puede ser difícil de implementar por lo que debería considerarse.

La validación de requisitos por casos de prueba requiere el desarrollo de cuatros fases: i) planeación de la prueba. Aquí se define la estrategia a utilizar, el alcance de la prueba a realizar y los tiempos requeridos para el desarrollo de la prueba; ii) diseño de los casos de prueba; iii) ejecución de los casos de prueba y iv) elaboración del informe final de la prueba donde se describen los aspectos más importantes y hallazgos de la ejecución de la prueba (Pantaleo, 2018).

De otra parte, el diseño de un caso de prueba requiere: i) la construcción de un instrumento donde se debe detallar para cada requerimiento si existen precondiciones, es decir, si se requieren de actividades o valores previos para poder iniciar la ejecución del requerimiento; ii) identificar los pasos a seguir para la validación de un requisito y iii) los resultados esperados de la realización de cada paso.

Formato de caso de prueba	
Objetivo del caso de prueba	

Formato de caso de prueba	
Identificador	
Nombre o identificador del requerimiento	
Precondiciones	
Pasos	**Resultados esperados**
1.	**1.**
2.	**2.**
3.	**3.**

Tabla 1 - Formato de caso de prueba | Nota: Adaptado de Pantaleo (2018).

1 Metodologías de desarrollo de *software*

Es importante aclarar que existe un gran número de definiciones sobre lo que es una metodología, para evitar cualquier confusión en este material se utiliza la definición dada en Maida y Pacienzia (2015), quienes indican que **una metodología hace referencia a un conjunto de procedimientos genéricos y lógicos que se utilizan para alcanzar un objetivo particular usando un conjunto de habilidades y conocimientos.**

Las metodologías de desarrollo de *software* siempre parten de un componente teórico y cuando son usadas por los equipos de trabajo conllevan a la utilización de un conjunto de técnicas y métodos que al final determinarán las tareas generales y específicas que se deberían realizar para alcanzar un objetivo.

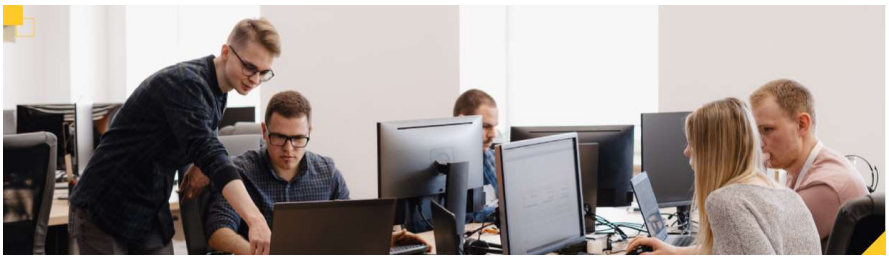


Existen diferentes tipos de metodologías de desarrollo de *software* que fueron ideadas pensando en problemas particulares presentados en la industria en contextos específicos, por lo cual es importante conocer sus diferentes características y contrastarlas con las necesidades particulares a las que se enfrenta a la hora de desarrollar un producto y servicio. Es decir, cada una de estas tiene ventajas y enfoques que pueden ser reutilizados en diferentes momentos.

Existen dos grandes clasificaciones de metodologías de desarrollo de *software* que se agrupan generalmente como marcos de trabajo tradicionales o marcos de trabajo ágiles que se verán a continuación.



2 Marcos de trabajo tradicionales



Para el de o de un buen producto de *software* se debe iniciar por un excelente proceso de planificación y gestión de este durante todas las etapas y actividad involucran transformar una idea o requerimiento en un producto o servicio que será usado por un cliente particular.

> Los marcos de trabajo tradicionales o metodologías tradicionales se caracterizan por centrar la mayor parte de su esfuerzo en la planeación y control del proceso, lo que conlleva a una documentación exhaustiva y precisa de los artefactos que describen los requisitos y los modelos del sistema en las etapas iniciales del desarrollo del proyecto
>
> (Maida y Pacienzia, 2015).



Lo anterior supone que este tipo de enfoques son óptimos en proyectos en los cuales los requisitos están plenamente identificados y delimitados, donde no se producirá ningún cambio en lo establecido mientras el proyecto es finalizado.

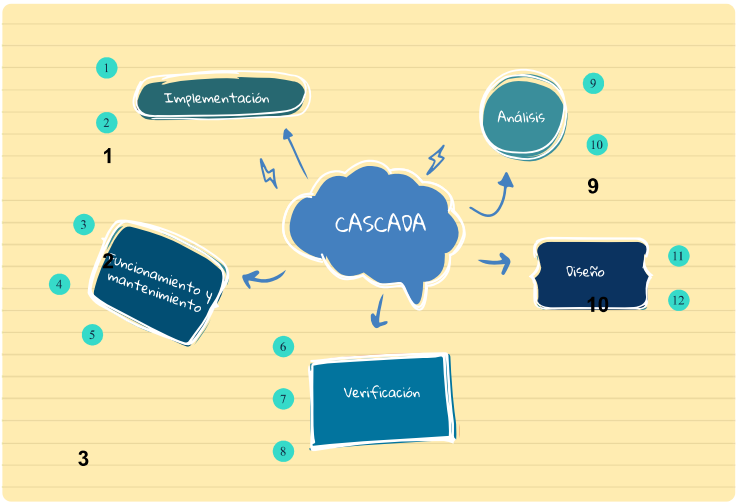
A continuación, se describen algunas metodologías que se enmarcan en los marcos tradicionales de desarrollo de *software*.

2.1 Cascada

Este es uno de los modelos genéricos más ampliamente conocido en la ingeniería de *software* y se deriva de procesos de ingeniería de sistemas más generales (Royce, 1970). Este modelo plantea un proceso lineal donde las actividades de desarrollo de un producto o servicios de *software* se agrupan en un conjunto de fases sucesivas donde estas son desarrolladas una única vez y los resultados de cada fase son la entrada requerida para cada fase subsiguiente, ninguna fase puede iniciar si la fase anterior no ha sido finalizada generalmente mediante un formalismo que puede ser un documento.

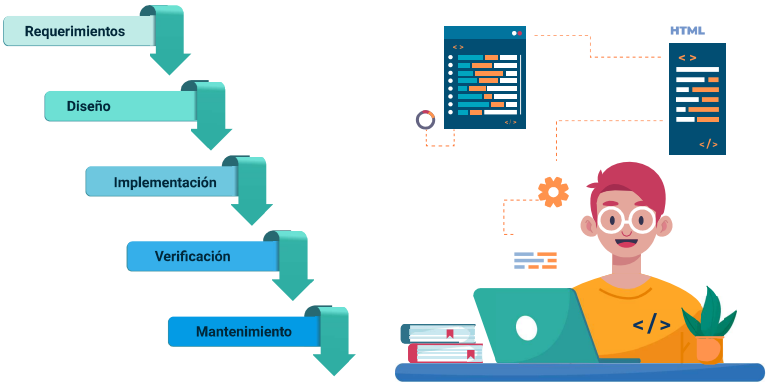
Según Sommerville (2005) el modelo en cascada se compone de cinco (5) etapas principales que se asocian con actividades fundamentales en el proceso de desarrollo de *software*, las cuales son:





Las anteriores fases se pueden visualizar en la siguiente figura donde se sintetiza la información anteriormente mencionada.

Etapas del modelo cascada



Siguiendo con el tema se presenta un cuadro comparativo donde se resumen las principales ventajas y desventajas de este modelo.

Tabla 1 Ventajas y desventajas del modelo cascada.

Ventaja	Desventaja
La definición clara de fases permite el desarrollo de una estructura sencilla que es ideal para proyectos sencillos y cortos	No se acopla muy bien a proyectos complejos con múltiples equipos trabajando en paralelo ya que las fases normalmente se traslapan y es difícil diferenciarlas.
Siguiendo este modelo se genera una muy buena documentación del proceso y es posible definir hitos claros.	Es difícil introducir cambios en el transcurso del proyecto.
Es más fácil realizar la estimación de costos al inicio del proyecto.	Los usuarios finales y clientes normalmente son integrados al final del proceso lo que impide tener realimentación y ajustes en etapas tempranas.
Es fácil elaborar cronogramas de trabajo en base al desarrollo de las actividades de cada fase.	Existen fallos que solo son detectados cuando el sistema entra en funcionamiento, lo que puede ser desastroso para un proyecto que sigue este modelo.

El modelo en cascada define cuatro grupos de roles típicos los cuales se mencionan a continuación:



Desarrolladores

Es el rol más importante en la metodología cascada ya que son los encargados directos de la creación de código.



Testers

Encargados de encontrar fallas en los productos finales y retornar el software a los desarrolladores para arreglar todos los defectos.



Analista del negocio

Encargado de la realización de estrategias de negocio que le permitan al producto software alcanzar popularidad en el mercado digital.



Administrador del proyecto

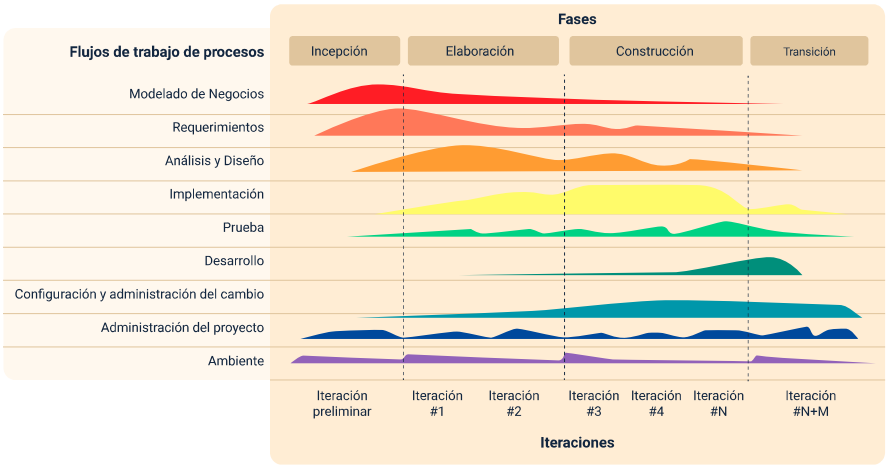
Es responsable de la calidad final del software. Administra el proyecto y lo subdivide en tareas entre los miembros del equipo.

2.2 Proceso Racional Unificado - RUP

RUP es una sigla en inglés equivalentes a Proceso Racional Unificado, el cual es un proceso de desarrollo de software tradicional basado en el modelo cascada y que fue desarrollado por la empresa Rational Software que es propiedad de IBM, esta metodología se centra en la arquitectura y es guía por casos de uso (requerimientos) (Kruchten, 2003).

RUP divide el proceso de desarrollo en cuatro grandes fases, dentro de las que se realizan algunas iteraciones donde se desglosan, en mayor o menor intensidad, un conjunto de disciplinas según la fase que se está abordando. A continuación, se observan las fases y disciplinas propuestas por RUP.

Fases y disciplinas de RUP



Nota. Tomada de Man, Hafriz, Nural, Maizura, Noor, Wan, Bakar y Man (2008).

A continuación, se describen las fases y su impacto en el proceso, por lo que se le invita a revisar el siguiente recurso de aprendizaje.

1 **Primera fase** 2 Segunda fase 3



La **primera gran fase** definida por RUP es la de **inicio**, en la cual se abordan actividades principalmente enfocadas en la comprensión del problema y el tipo de tecnología a utilizar, por lo que hay una gran carga en actividades relacionadas con la disciplina de modelado del negocio y especificación de requisitos.

La **segunda fase** es la elaboración, donde se centra la en la definición general de la arquitectura del sistema y requisitos y modelado del negocio. RUP generalmente modelado UML para el modelado del negocio y descrip



Es importante aclarar que dentro de cada una de las fases se desarrollan varias de las disciplinas, pero el esfuerzo y tiempo que se invierte en ellas tendrá mayor o menor enfoque dependiendo de la fase en la que se encuentre el proyecto.

Otro aspecto importante de esta metodología es que se apoya en un conjunto de artefactos que son desarrollados con el fin de especificar el proceso de análisis y diseño que soporta la construcción del software, entre los artefactos más comunes se encuentran:

Documento de visión.
Documento de especificación de requisitos.
Diagramas de casos de uso.
Modelos conceptuales (clases y entidad relación).

Diagramas que representan la vista de implementación como:

- Diagramas de secuencia.
- Diagramas de estados.
- Diagramas de colaboración, entre otros.



Las disciplinas, por otra parte, representan un conjunto de actividades relacionadas con un área específica del proyecto y están inspiradas en el modelo en cascada. RUP establece las siguientes disciplinas según Kruchten (2003):



- Modelado de negocios.
- Requerimientos.
- Análisis y diseño.
- Implementación.
- Pruebas.
- Despliegue.
- Configuración.
- Administración del cambio.
- Administración de proyectos y ambiente.

A continuación, se describen cada una de las disciplinas mencionadas y cómo estás aportan en el proceso.

RUP propone una categorización de roles encargados de la realización de actividades dentro de cada una de las disciplinas que son:

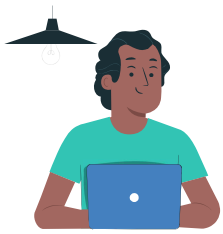
 **Analistas**	 Desarrolladores
 Probadores	 Encargados y otros actores.

A continuación se describen cada uno de los roles:

En los **roles específicos** clasificados en la categoría de analistas se encuentran:

- Analistas de procesos de negocio.
- Diseñadores del negocio.
- Analistas del sistema.
- Especificador de requisitos.
- Diseñadores de interfaces de usuario o similares.





Existen otras metodologías que son categorizadas como marcos de trabajo tradicionales, sin embargo, la idea de este espacio es mostrar las características principales de los más reconocidos en la industria de software. Si se desea profundizar en las ventajas y desventajas de estas metodologías y otras, se invita a explorar el apartado de materiales complementarios. A continuación, se detallarán las metodologías enmarcadas dentro de la categoría de marcos de trabajo ágiles.

3 Marcos de trabajo ágiles



> Los marcos de trabajo ágiles o metodologías ágiles para el desarrollo de *software* nacen como otra opción para abordar proyectos donde no es posible tener un detalle completo de los requerimientos y sus estimaciones en la primera fase del proyecto o donde es necesario hacer procesos de adaptabilidad a lo largo del proceso de desarrollo de *software*.
>
> (Maida y Pacienzia, 2015).

Las metodologías ágiles proveen un conjunto de pautas y principios que buscan facilitar y priorizar la entrega de producto sobre procesos de documentación exhaustiva, haciéndolos más simples, donde interactúa el cliente final desde las primeras etapas del proyecto. El inicio de las metodologías ágiles nació en el año 2001 a partir del manifiesto ágil de *software* donde se establecen cuatro valores fundamentales (Manifiesto Ágil, 2001):

Individuos e interacciones sobre procesos y herramientas.
Software funcionando sobre documentación extensiva.
Colaboración con el cliente sobre negociación contractual.
Respuesta ante el cambio sobre seguir un plan.

Adicionalmente a los valores ágiles anteriormente listados, el manifiesto ágil establece 12 principios ágiles para materializar los valores definidos (Manifiesto Ágil, 2001):



1. “Nuestra principal prioridad es satisfacer al cliente a través de la entrega temprana y continua de software de valor.
2. Son bienvenidos los requisitos cambiantes, incluso si llegan tarde al desarrollo. Los procesos ágiles se doblegan al cambio como ventaja competitiva para el cliente.
3. Entregar con frecuencia software que funcione, en periodos de un par de semanas hasta un par de meses, con preferencia en los períodos breves.
4. Las personas del negocio y los desarrolladores deben trabajar juntos de forma cotidiana a través del proyecto.
5. Construcción de proyectos en torno a individuos motivados, dándoles la oportunidad y el respaldo que necesitan y procurándoles confianza para que realicen la tarea.
6. La forma más eficiente y efectiva de comunicar información de ida y vuelta dentro de un equipo de desarrollo es mediante la conversación cara a cara.
7. El software que funciona es la principal medida del progreso.
8. Los procesos ágiles promueven el desarrollo sostenido. Los patrocinadores, desarrolladores y usuarios deben mantener un ritmo constante de forma indefinida.
9. La atención continua a la excelencia técnica enaltece la agilidad.
10. La simplicidad como arte de maximizar la cantidad de trabajo que se hace, es esencial.
11. Las mejores arquitecturas, requisitos y diseños emergen de equipos que se auto organizan.
12. En intervalos regulares, el equipo reflexiona sobre la forma de ser más efectivo y ajusta su conducta en consecuencia”.

A continuación, se invita a profundizar en el tema revisando la sección de material complementario donde encontrará un video que le permitirá ampliar el conocimiento, por otro lado, en los numerales siguientes podrá identificar otra serie de factores importantes que debe tener en cuenta.

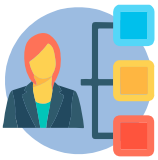
3.1 Programación Extrema - XP

XP es la abreviación comúnmente utilizada para referirse a Extreme Programming, que es un marco de desarrollo de software ágil que busca producir software de alta calidad en contextos con requisitos altamente cambiantes, riesgos que involucran tiempos fijo con tecnologías nuevas y equipos de trabajo pequeños ubicados en un mismo sitio.

En XP se definen cinco valores según Beck y Andrés (2004):



Comunicación: el desarrollo de software requiere de trabajo en equipo por lo cual es importante la transferencia de conocimientos y utilizar medios de comunicación apropiados, se propone la discusión cara a cara con herramientas que permitan dibujar o escribir, como, por ejemplo: un tablero.



Simplicidad: esto se refiere a hacer solo las cosas que sean absolutamente necesarias evitando el desperdicio, elaborar las cosas de forma que sea fácil entender por otros y abordar solo requisitos conocidos.



Retroalimentación: esto permite identificar áreas de mejora y revisión constante de las prácticas implementadas en el proceso de forma que se puedan establecer procesos de mejora permanentes.



Coraje: se refiere al actuar sobre situaciones que pueden ser retadoras para el equipo, como, por ejemplo:

Enfrentar problemas organizacionales.

Intentar implementar funcionalidades de formas diferentes cuando lo convencional no funciona y aceptar comentarios, etc.



Respeto, los miembros del equipo deben respetarse entre sí para comunicarse y trabajar en equipo. Cada persona contribuye en favor de lograr el objetivo del equipo.

Adicional a los valores XP se caracteriza por la definición de un conjunto de 12 prácticas de desarrollo de software que aunque pueden ser adoptadas de forma aislada tiene mayor relevancia cuando son desarrolladas en conjunto (Jeffries, 2011):



El juego de la planificación

Consiste en una reunión de planeación que realiza el equipo de desarrollo en conjunto con el cliente para discutir y probar las características a ser desarrolladas.



Pequeños lanzamientos

Esta práctica sugiere hacer iteraciones cortas con funcionalidades pequeñas que puedan ser probadas con el cliente de forma que se obtenga realimentación temprana y frecuente.



Metáfora

Los nombres usados para definir cualquier tipo de identificador en el sistema deben ser coherentes. El diseño y la estructura en general deben ser comprensibles para cualquier persona.



Diseño simple

Esto implica hacer un código que funcione y que sea a la vez la solución más simple posible, evitando duplicación de código, menor número de métodos y clases.



Pruebas

Sugiere el uso de técnicas donde se enfatice en el proceso de pruebas constantes antes de iniciar la construcción del código, como, por ejemplo: TTD (desarrollo basado en pruebas)



Refactorización

Eliminación de funciones innecesarias, desacoplar elementos y eliminar cualquier tipo de redundancia de forma que se mantenga un código lo más limpio posible, que sea fácil de entender y modifica.

Los roles fundamentales establecidos por este marco de trabajo ágil son los siguientes: cliente, programador, coach, tester y mánager.





Los clientes son los encargados del establecimiento de las prioridades del proyecto y las necesidades puntuales.



Los programadores se encargan de transformar esos requerimientos en bloques de código funcional, son el centro del marco de *Extreme Programming*.



Los *testers* se encargan de la aplicación de pruebas para garantizar la calidad de los productos y servicios desarrollados.



El *coach* es una figura encargada de brindar asesoría a los miembros del equipo y son los que definen el rumbo del proyecto.



El manager se encarga de la coordinación de actividades, del aseguramiento de los recursos requeridos para el proyecto y quien tiene la responsabilidad de la comunicación externa.

3.2 Desarrollo rápido de aplicaciones - RAD

RAD (Desarrollo Rápido de Aplicaciones) es otra metodología de desarrollo de software ágil que se centra en el desarrollo rápido de aplicaciones mediante la realización de iteraciones frecuentes y realimentación constante y fue inventado por James Martin en 1991.

Algunas características fundamentales de RAD son:

Mayor flexibilidad y adaptabilidad a cualquier ajuste que deba realizarse durante el proceso de desarrollo.

Iteraciones rápidas que reducen el tiempo de desarrollo y mantienen un ritmo de entrega acelerado.

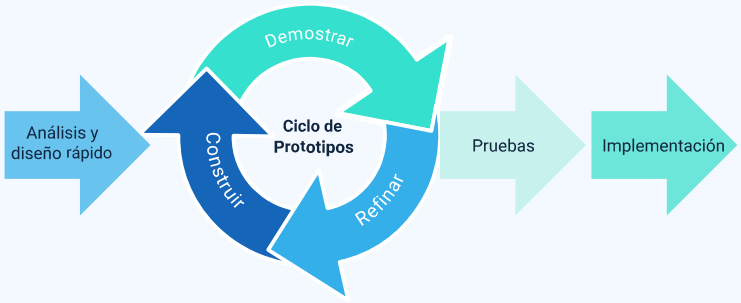
Se fomenta la reutilización de código.

Mejor gestión del riesgo, ya que las partes interesadas discuten y abordan cualquier vulnerabilidad mientras se construyen los productos.



A continuación, se describen las fases propuestas en RAD.

Figura 3 Fases definidas en RAD



Nota. Tomada de Blog Capterra (2019).

La **primera fase** definida corresponde a la **definición y finalización** de los requisitos del proyecto. En esta fase las partes interesadas se reúnen para definir los requisitos del proyecto definiendo entre otros casos:

Los objetivos.

Las expectativas.

Los plazos.

Presupuesto del proyecto.

La **segunda fase** aborda la construcción de **prototipos** los cuales son construidos, validados y mejorados a partir de la validación con los usuarios y una vez son aprobados pasan a la **tercera fase** donde estos prototipos son transformados en modelos funcionales.

En la **cuarta fase** se enfoca en la realización de **pruebas** exhaustivas para garantizar que todos los elementos construidos funcionan bien individualmente y también de forma colectiva. Finalmente, en la **última fase** se realizan todas las actividades de lanzamiento del producto lo que involucra el cargue inicial de datos y entrenamiento a los usuarios.



Según HKSAR (2009) los roles más importantes de la metodología RAD son: facilitador, escriba, equipo *Swat*, administrador del modelo, administrador de bases de datos, equipo de planificación de *Workshop*, equipo de diseño de usuario, equipo de soporte de construcción y equipo de transición, a continuación se mencionan las características más importantes de cada rol.



Facilitador: se encarga del aseguramiento de los objetivos, es responsable de los *Workshops* de captura de requerimientos, se asegura del alistamiento de materiales, es un mediador en procesos de resolución de conflictos.



Equipo Swat: equipo encargado del diseño y construcción del sistema.

Administrador de bases de datos: responsables del



rendimiento, integridad y seguridad de los datos de la organización



Equipo de diseño de usuario: proveen información detallada describiendo las funciones del negocio y procesos que serán afectados por el sistema propuesto.



Equipo de transición: desarrolla las tareas necesarias para preparar y llevar el sistema a entornos de producción.



Escriba: es responsable de la documentación de todas las salidas producto de los *Workshops* de captura de requerimientos y registra información de todo el proceso.



Administrador del modelo: encargado de coordinar el desarrollo y mantenimiento de las arquitecturas modelos resultantes del proyecto.



Equipo de planificación de los *Workshops*: asisten en la definición de requerimientos y el alcance propuesto para el sistema propuesto.



Equipo de soporte de construcción: equipo que se asegura que las necesidades del usuario final sean alcanzadas en el



sistema completo mediante la entrega de conocimiento detallado de actividades de negocio y realimentación en la usabilidad del sistema al equipo Swat.

3.3 Scrum

Scrum es un marco de trabajo ágil de muy amplio uso en la industria del software que se fundamenta en los valores y principios ágiles definidos en (Manifiesto Ágil, 2001) y donde se definen tres pilares fundamentales según (SCRUMstudy, 2013) los cuales se describen a continuación:

Transparencia

Hace referencia a que cualquier proceso de Scrum puede ser conocido por cualquiera. Esto es posible por medio de eventos como:

- Las reuniones de revisión y reuniones diarias.
- Artefactos como la pila de producto.
- Cronogramas de lanzamiento.
- Documentos de visión del proyecto.
- Instrumentos de seguimiento, como: el *burndown chart* o el tablero de *Scrum* (*Scrum board*).

Inspección

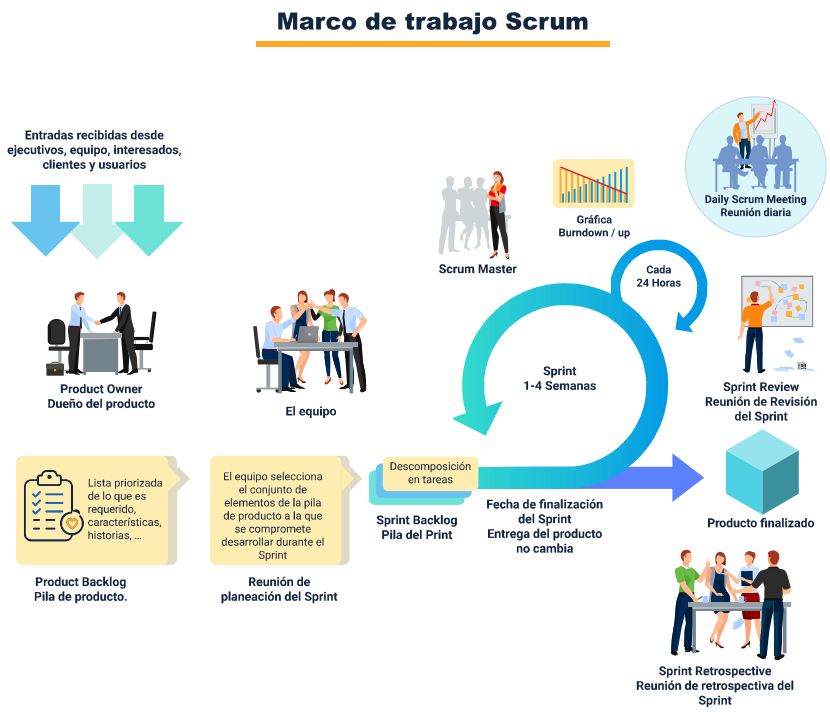
Permite que cualquiera pueda estar enterado de las actividades realizadas por otros y en general conocer el estado actual de los procesos.

Adaptación

Por medio de la transparencia y la inspección es posible fijar actividades de mejoras que permitan modificar todo tipo de proceso en pro de lograr más altos estándares de calidad.



Adicionalmente, este marco de trabajo ágil está estructurado por un conjunto de roles, eventos y artefactos que pueden ser observados a continuación:



Nota. Tomada de pngwing.com (s.f.)

Dentro de los roles hay una división en dos categorías fundamentales:



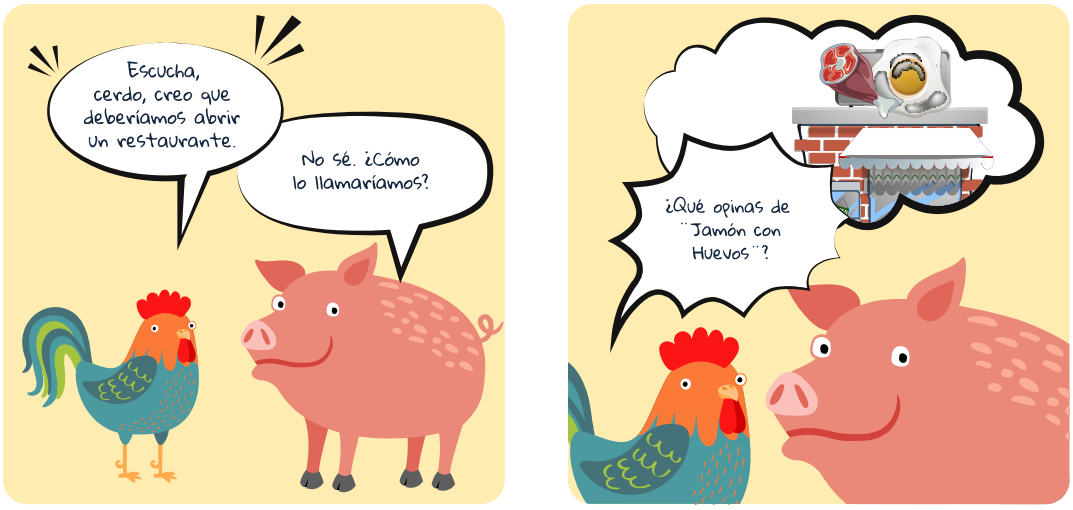
Los roles centrales que hacen referencia a los requeridos obligatoriamente para la creación de un producto, están altamente comprometidos y de los cuales depende el éxito o no de un proyecto.



Los roles no centrales que se refieren a todo el personal interesado en el proyecto, pueden interactuar con el equipo, pero no son los responsables del éxito del mismo, dentro de esta categoría entran los *stakeholders*, directivos, gerentes, *marketing*, asesores, etc.

Para representar mejor estas dos categorías se hace referencia a la historia del cerdo y la gallina que se observa en la siguiente figura, los roles centrales son el cerdo en la historia y las gallinas los roles no centrales.

Fábula roles de Scrum



Nota. Tomada de implementingscrum.com (2006).



Por otro lado hay tres roles centrales dentro del marco de trabajo de Scrum (SCRUMstudy, 2013) que se describen a continuación:

Roles de Scrum



Dueño del producto (*Product Owner*)

persona con amplio conocimiento en el negocio del cliente, sus necesidades y las tendencias del mercado para el área específica. Este rol está encargado de maximizar el valor de negocio entregado al cliente y es el único responsable del control del Product Backlog (requerimientos) y su priorización. Este también representa al cliente en algunos procesos de demostración de avances y determina cuándo aprobar o no una entrega.



Scrum Master

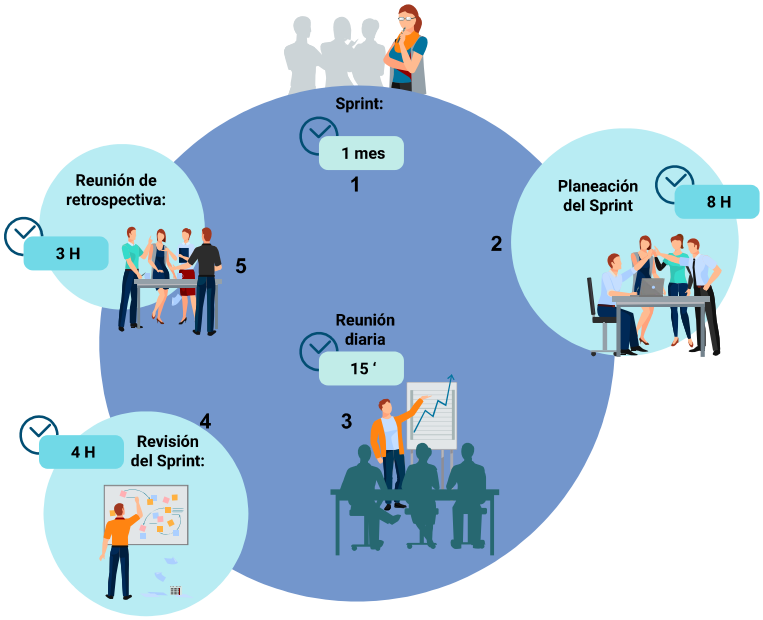
es un rol que se encarga de facilitar los procesos al interior del equipo de trabajo removiendo cualquier impedimento y apoyando procesos de empoderamiento personal, debe velar porque los elementos propios del marco de trabajo Scrum se apliquen de manera correcta.



Equipo de desarrollo (*Developer Team*)

son los responsables de la transformación de los requerimientos en código ejecutable a ser usado por el cliente, pero también son responsables de la planificación de las iteraciones, establecimiento de características para tener en cuenta en la verificación de un requerimiento terminado y presentación de avances a los clientes. Generalmente es un equipo autoorganizado y autogestionado.

Además de los roles, Scrum define un conjunto de eventos con participantes y objetivos claros que se desarrollan en momentos particulares del flujo general de Scrum, a continuación, se detalla cada uno de estos:



Finalmente, el marco de trabajo Scrum define un conjunto de artefactos que permiten registrar y gestionar información clave para asegurar los tres pilares fundamentales y proveen información valiosa durante todo el proceso de desarrollo de software. Entre los artefactos representativos de este se encuentran los siguientes:



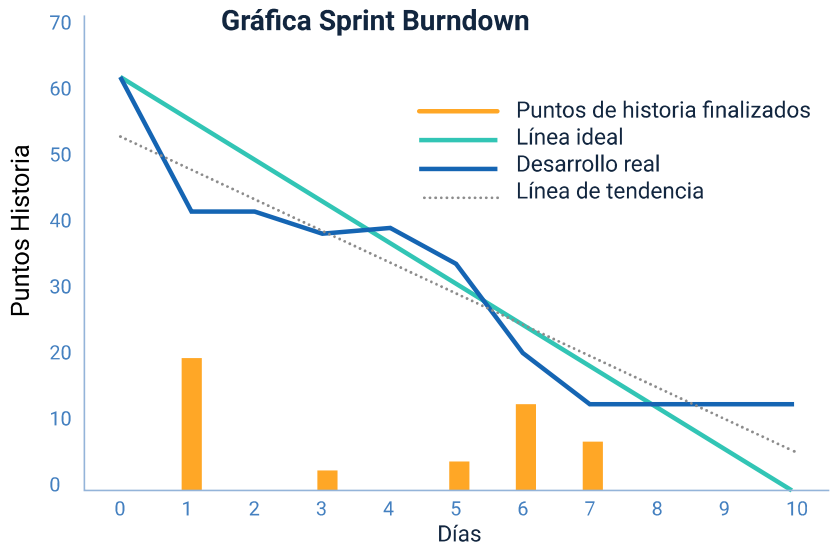
1. **Pila de producto (*Product Backlog*)**: lista priorizada de requerimientos generalmente descritos en formato de historias de usuarios que representa todas las características del sistema a construir.

2. **Pila del *Sprint* (*Sprint Backlog*)**: lista de requerimientos seleccionados desde el *Product backlog* por el equipo de trabajo para ser desarrollados durante un *Sprint* particular. Este es uno de los artefactos generados en la reunión de planeación del *Sprint*.

3. ***Burndown Chart***: es un gráfico visual de dos ejes que muestra a los equipos la cantidad de trabajo pendiente por completar (eje Y) y el tiempo disponible para hacerlo (eje X). Este generalmente se realiza por cada *Sprint* ubicando la cantidad de trabajo a realizar del *Sprint Backlog* (usualmente medido por puntos de historia u de horas de trabajo) en un tiempo 0 y por cada día finalizado se resta la cantidad de puntos de historia u horas de cada tarea completada.

Por otro lado, también es posible usar este mismo gráfico para representar el avance general del proyecto ubicando en el eje Y la cantidad total de horas o esfuerzo del *Product Backlog* y en el eje X la cantidad de *Sprint* proyectados. Cada uno de estos puntos se une por medio de una línea y es posible determinar visualmente si el flujo de trabajo está en una situación óptima o no respecto al tiempo restante para completar el *Sprint*.

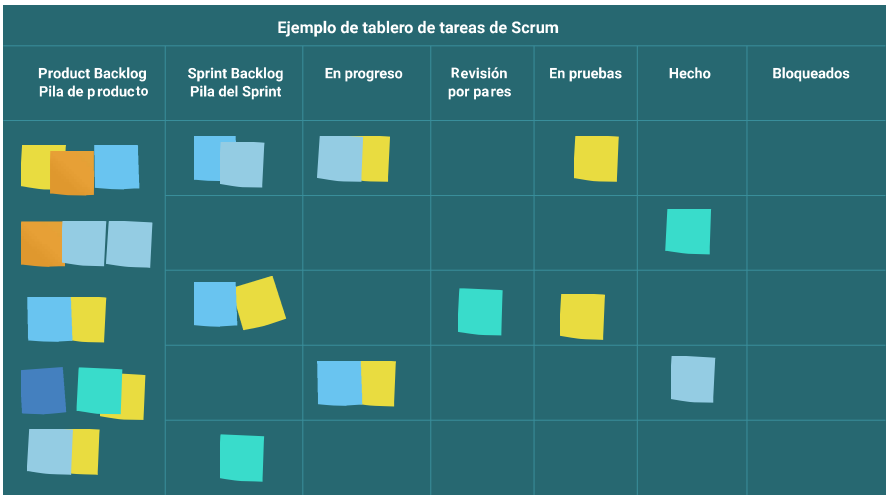
Figura 7 Burndown Chart



Tablero de *Scrum* (*Scrumboard*): es un elemento visual donde se integra la mayor parte de los elementos del marco de trabajo *Scrum*, en él se indica la carga de trabajo, el estado actual de cada una de las actividades y sus respectivos responsables. Este es un elemento que se sincroniza de manera permanente y facilita la implementación de los pilares de transparencia, inspección y adaptabilidad. Si bien se aconseja el uso de un tablero, existen diferentes tipos de herramientas digitales que permiten la implementación de un tablero de *Scrum*.

Nota. Tomada de intl-blog.imgix.net (2019).

Figura 8 Tablero de Scrum



Adicionalmente es de vital importancia mencionar que entre los principales beneficios del marco de trabajo Scrum se encuentran los siguientes elementos:

Es posible gestionar las expectativas del cliente de manera regular, ya que, este puede y debe participar en las reuniones de revisión, por lo que, está enterado todo el tiempo del estado actual del proyecto.

El cliente puede obtener resultados importantes y utilizables desde las primeras iteraciones, ya que, la lista de producto está priorizada para ofrecer mayor valor en el menor tiempo posible y porque cada finalización de Sprint debe tener como resultado una versión totalmente funcional.

El proyecto puede iniciar con requerimientos de muy alto nivel y es fácil administrar los cambios.

La participación constante del cliente permite mitigar riesgos del proyecto desde sus primeras etapas.

Los procesos de retrospectiva permiten establecer actividades permanentes de mejora continua en función de las experiencias del equipo.



Un elemento clave en su formación es la planeación de proyectos de software que aunado con los contenidos ya vistos le permite tener un panorama amplio sobre el tema.

Para profundizar le invito a revisar la sección de material complementario donde encontrará un video sobre el tema.