Iplacex
enovus

**UNIDAD I: DETECCIÓN Y PLANTEAMIENTO DE UN PROBLEMA DE ESPECIALIDAD INNOVADOR**

**ASIGNATURA: PROYECTO DE TÍTULO**

# Consideraciones Previas

## Sobre las fuentes utilizadas en el material

*El presente Material de Estudio constituye un ejercicio de recopilación de distintas fuentes, cuyas referencias bibliográficas estarán debidamente señaladas al final del documento. Este material, en ningún caso pretende asumir como propia la autoría de las ideas plantea-das. La información que se incorpora tiene como única finalidad el apoyo para el desarrollode los contenidos de la unidad correspondiente, respetando los derechos de autor ligados alas ideas e información seleccionada para los fines específicos de cada asignatura.*

# Introducción

¡Bienvenido/a a la asignatura de *Proyecto de Título*! Durante esta primera semana, daremos inicio al análisis y formulación de un proyecto innovador que aborde una problemática relevante en su entorno profesional, integrando soluciones creativas y efectivas en el ámbito de las tecnologías de la información (TIC).
En un mundo donde la tecnología avanza constantemente y las necesidades del entorno laboral son cada vez más complejas, los técnicos en informática tienen la oportunidad de liderar proyectos que trasciendan la solución de problemas cotidianos. Este desafío implica diseñar soluciones innovadoras, adaptables y con un impacto positivo en su contexto profesional. Esta semana, nos centraremos en cómo identificar esas problemáticas clave y transformarlas en proyectos estructurados y viables.

**¿Qué abordaremos esta semana?**

- Identificar problemáticas y oportunidades relevantes dentro de su especialidad, tomando en cuenta las demandas del entorno laboral.
- Diferenciar entre bases de datos relacionales y NoSQL, entendiendo sus características y cómo elegir la tecnología adecuada para su proyecto.
- Definir objetivos generales y específicos, asegurándonos de que sean claros, medibles, alcanzables, relevantes y con un tiempo definido (metodología SMART).
- Delimitar el alcance del proyecto, evaluando las necesidades a cubrir y el impacto esperado.
- Introducir herramientas de planificación como la Carta Gantt, fundamentales para organizar y visualizar las etapas del desarrollo del sistema.

**Propósitos de la Semana:**

- Comprender las etapas iniciales del ciclo de vida de un proyecto innovador en tecnologías de la información.
- Formular una problemática específica y relevante que guiará el desarrollo de su proyecto de título.
- Diseñar un marco inicial para el proyecto, considerando la factibilidad técnica, operativa y los requerimientos necesarios.

El objetivo de esta semana es sentar las bases para desarrollar una solución informática innovadora y fundamentada, que no solo cumpla con los requisitos técnicos, sino que también genere un impacto significativo en su entorno profesional. Este es el primer paso hacia la construcción de un proyecto que refleje sus conocimientos, habilidades y creatividad como técnicos en informática.

# Ideas fuerza

En esta semana abordaremos ideas clave que serán fundamentales para el desarrollo de su proyecto de título. Estas son los conceptos y habilidades a las que debemos prestar especial atención:

**Ciclo de vida de un proyecto innovador**

- Reconocer y describir cada etapa del ciclo de vida de un proyecto, comprendiendo cómo estas se aplican y contribuyen al desarrollo de soluciones tecnológicas innovadoras en el ámbito de las TIC.

**Identificación de problemáticas y oportunidades**

- Desarrollar la capacidad para observar y analizar el entorno laboral, identificando problemáticas concretas y oportunidades que puedan ser resueltas mediante enfoques creativos e innovadores.

**Bases de datos relacionales y NoSQL**

- Entender las diferencias clave entre los sistemas de bases de datos relacionales y NoSQL, explorando sus características principales y su aplicación específica en proyectos tecnológicos según las necesidades del contexto.

**Formulación de la problemática y planificación inicial**

- Plantear una problemática específica del área de especialidad, definiendo objetivos claros mediante la metodología SMART, delimitando el alcance del proyecto y estableciendo las bases para una planificación estructurada y eficiente.

# Índice

# Desarrollo

**PROYECTO DE TÍTULO**

**1. Definición y formulación del proyecto.**

Previo a la definición de la problemática como tal, comenzaremos con definir que es un proyecto ya que es precisamente en lo que debemos trabajar, en base a esto entonces vamos a definir un proyecto como *aquella asociación de esfuerzos que se encuentra establecido en un tiempo determinado, cuenta con un objetivo concreto y necesita contar con el acuerdo de un conjunto de especialidades y recursos*. Por otra parte, se puede definir como una *organización en un tiempo determinado que tiene como finalidad obtener un propósito definido*. En el momento en que los objetivos de un proyecto son alcanzados se entiende que el proyecto está completo.

En el caso de los proyectos informáticos, estos cumplen con las características generales de los proyectos, pero presentan particularidades clave, como:


**Figura 1:** Particularidades clave proyectos informáticos

**Fuente:** (Villamar, 2024)

El ciclo de vida de un proyecto informático comienza con la identificación y definición de problemas, oportunidades y objetivos. Esta etapa inicial es crucial para garantizar el éxito del proyecto, ya que establece los cimientos para todas las fases posteriores.

El técnico en informática debe realizar un análisis inicial para comprender la situación actual de la empresa u organización. Esto incluye:

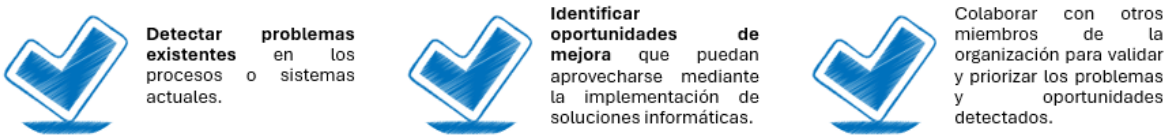


**Figura 2:** Consideraciones en análisis inicial.

**Fuente:** (Villamar, 2024)

La formulación de la problemática del proyecto se basa en las oportunidades identificadas y en cómo estas pueden resolverse o potenciarse mediante el uso de **sistemas de información** computarizados. Este enfoque permite no solo solucionar problemas actuales, sino también generar ventajas competitivas o establecer nuevos estándares en la industria.

**Identificación de objetivos**

Definir objetivos claros es un paso esencial en esta fase inicial. El técnico en informática debe:

- Comprender los objetivos estratégicos de la empresa.
- Evaluar cómo las soluciones tecnológicas pueden contribuir a alcanzar esos objetivos.
- Formular objetivos específicos para el proyecto, asegurándose de que sean medibles, alcanzables y relevantes.

En esta etapa, el técnico desempeña un rol fundamental al traducir las necesidades organizacionales en soluciones concretas, estableciendo un marco sólido para el desarrollo del proyecto de título.

En la primera fase del desarrollo de software, centrada en la identificación y definición de problemas, oportunidades y objetivos, las personas involucradas suelen ser las siguientes:

| PRINCIPALES ROLES | DESCRIPCIÓN | ROL EN LA FASE INICIAL | CONTRIBUCIÓN |
|---|---|---|---|
| 1. **TÉCNICO EN INFORMÁTICA** | Profesional encargado de analizar problemas, formular soluciones y colaborar en el desarrollo del proyecto. | Lidera el levantamiento de requerimientos técnicos y funcionales.<br>Analiza los datos recopilados de usuarios y administradores de sistemas para identificar problemas y oportunidades.<br>Proporciona una visión práctica para implementar soluciones innovadoras. | Actúa como puente entre usuarios y administradores, asegurando que las soluciones sean funcionales, factibles y alineadas con las necesidades del entorno. |
| **2. ADMINISTRADORES DE SISTEMAS** | Responsables de la infraestructura tecnológica y del correcto funcionamiento de los sistemas existentes. | Proveen información técnica sobre los sistemas actuales, incluyendo configuraciones, restricciones y rendimiento.<br>Identifican posibles desafíos técnicos o riesgos al implementar una nueva solución.<br>Validan la viabilidad técnica de las propuestas iniciales. | Garantizan que las soluciones propuestas sean compatibles con la infraestructura existente y viables desde el punto de vista técnico. |
| 3**. USUARIOS FINALES** | Personas que utilizarán directamente el sistema una vez implementado. | Proporcionan información sobre las dificultades y limitaciones del sistema actual.<br>Describen sus necesidades y expectativas respecto al nuevo sistema.<br>Participan en sesiones de levantamiento de requerimientos para garantizar que sus problemas sean considerados. | Aseguran que el proyecto esté enfocado en resolver problemas reales y facilitar las tareas cotidianas. |

**Figura 3:** Principales roles involucrados en la primera fase del desarrollo.

**Fuente:** (Villamar, 2024)

En esta fase, las actividades principales consisten en entrevistar a los usuarios finales y a los administradores de sistemas, recopilar información sobre los problemas existentes y necesidades identificadas, exponer de manera clara los hallazgos obtenidos, estimar el alcance preliminar del proyecto y documentar los resultados en un informe inicial. *El producto final de esta etapa es un informe de*

*viabilidad, que incluye la definición del problema y un resumen de los objetivos propuestos.* Con base en este informe, la administración o el responsable del proyecto decidirá si se continuará con el desarrollo del sistema.

Si los usuarios finales no cuentan con el presupuesto necesario, los problemas identificados no están directamente relacionados con la solución informática, o si las necesidades pueden ser resueltas de manera más eficiente mediante otro enfoque, es posible que el proyecto no continúe. En estos casos, se pueden proponer soluciones alternativas y detener el desarrollo del sistema computacional.

## 1.1 Diagnóstico e identificación de problemáticas específicas.

### Diagnóstico de la situación actual

Para comenzar el desarrollo del proyecto, es necesario realizar el diagnóstico de la situación actual, este diagnóstico es un estudio que se realiza previamente a toda planificación o proyecto y radica en reunir y tratar información importante de la organización con la finalidad de comprender su funcionamiento e identificar las debilidades y fortalezas existentes en la organización.

Al efectuar el análisis de la situación con respecto a un proyecto en particular, se podrán identificar aquellos problemas principales en torno al contexto analizado y definir la problemática central de la situación, junto con la visualización de la relación causa efecto. En un proyecto de desarrollo de software la definición general se encuentra enfocada en describir en que consiste el sistema o el desarrollo en sí, explicar la idea general y cuál es la funcionalidad central del proyecto y detallar el propósito del desarrollo.

En base a todos los requerimientos que han sido detectados por medio de este proceso, es posible entregar un diagnóstico en base a las necesidades que se encuentran presentes en la empresa. Los que posteriormente darán paso a la generación de los objetivos del proyecto.

### Definición del problema

Con respecto a la definición del proyecto se deben entregar detalles relacionados con la funcionalidad esperada, cuáles son los objetivos asociados a las necesidades existentes y definir quienes serán los usuarios del sistema, en detalle lo pasamos a revisar:

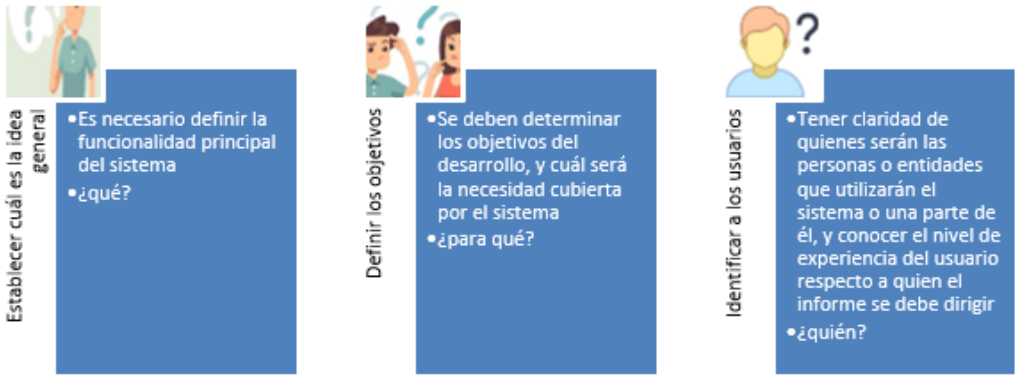


**Figura 4:** Definición del proyecto, aspectos relacionados.

**Fuente:** (Villamar, 2024)

### 1.2.- Formulación de propuesta de solución en el área de especialidad.

La formulación de una propuesta de solución en el área de especialidad de un técnico en informática implica diseñar un enfoque estructurado y práctico para resolver una problemática específica identificada en el entorno profesional. Este proceso comienza con la definición clara del problema, analizando sus causas, efectos y el contexto en el que ocurre, además de identificar el tipo de información y recursos necesarios para abordarlo de manera efectiva.

En esta etapa, es fundamental proponer una solución basada en herramientas y métodos propios de la especialidad, tales como el diseño e implementación de bases de datos, la automatización de procesos mediante scripts o aplicaciones, o el análisis de datos utilizando técnicas predictivas y de visualización. Este enfoque asegura que las soluciones sean relevantes, factibles y alineadas con las demandas del entorno laboral.

La propuesta debe ser:


**Figura 5:** Características de la propuesta.

**Fuente:** (Villamar, 2024).

### 1.3. Definición de propuesta de valor y objetivos SMART.

La formulación de una propuesta de valor en el área de especialidad de un técnico en informática consiste en definir el beneficio concreto y medible que el proyecto aportará a la organización o al cliente. Este valor debe responder a una necesidad específica del entorno laboral y destacar por su relevancia, aplicabilidad y capacidad de solución.

En este proceso, es esencial identificar cómo la propuesta mejora procesos, optimiza recursos, automatiza tareas o facilita la toma de decisiones mediante el uso eficiente de herramientas informáticas, como bases de datos, sistemas de gestión o soluciones de análisis de datos. La propuesta de valor debe ser clara y enfocada, demostrando cómo la solución puede transformar datos en conocimiento práctico, reducir costos operativos, aumentar la eficiencia o mejorar la competitividad de la organización en su sector.

Esta formulación asegura que todos los involucrados comprendan el propósito y la relevancia del proyecto, alineando esfuerzos hacia objetivos concretos que generen un impacto positivo y justifiquen la inversión en términos de tiempo, recursos y expectativas de resultados.

**Formulación de objetivos generales y específicos.**

**Objetivo General**

Si bien es cierto se dispone de fórmulas para la redacción de los objetivos, siempre es necesario adaptarlas de acuerdo con el tipo de proyecto que se esté desarrollando de manera particular y a los recursos que se disponga.

Generalmente, se indica que un objetivo general debe:

- Estar limitado en el tiempo.
- Comenzar con un verbo en infinitivo que pueda asociarse a un logro, una actividad que se ejecuta eficientemente, pero no lleva a ningún logro, no se puede considerar un objetivo alcanzado. Hay objetivos que comienzan con más de un verbo cuando es necesario llevar a cabo múltiples actividades para alcanzar el logro deseado.
- Ser alcanzable, es importante realizar un análisis de cuáles serán los obstáculos durante el proceso, al igual que las ventajas y recursos a favor para planificar el objetivo general con base en ello. Con datos claros, es

posible adaptase mejor a la realidad.

- Ser claro y fácil de comprender, no hay que olvidar que un objetivo general es un resumen de la idea central del proyecto. Si se formula de forma sencilla, será más fácil comunicarlo a los miembros del equipo.

Vamos a considerar como ejemplo, el siguiente caso:

**InvControl: Sistema Integral de Gestión de Inventarios**

Diseño e implementación de un sistema de gestión de inventarios para una pequeña empresa, que incluya una interfaz web interactiva desarrollada con HTML, CSS y Flask, bases de datos SQL y NoSQL para manejar registros y consultas avanzadas, y características de seguridad basadas en normas de ciberseguridad.

Revisaremos en base al caso anterior, un objetivo general bien redactado y un objetivo general mal redactado o incorrecto:


**Figura 6:** Ejemplo objetivo general.

**Fuente:** (Villamar, 2024).

**Objetivos específicos**

El objetivo general de un proyecto se puede descomponer en objetivos específicos. Para alcanzar la meta principal, se precisa listar el “paso a paso” y las tareas sucesivas que nos llevarán a buen término. Dicho de otra manera, para alcanzar un objetivo general es necesario completar distintos objetivos específicos.

*Podemos afirmar que la suma de todos los objetivos específicos es la concreción del objetivo general.*

Al momento de determinar un objetivo específico es necesario considerar que debe comenzar con un verbo en infinitivo y ser:

- **Concreto:** debe postular acciones reales y detalladas.
- **Medible:** porque ha de contemplar instrumentos que permitan evaluar el progreso.
- **Alcanzable:** al momento de formularlo, se debe considera su alcance. Debe ser coherente con los recursos y estrategias actuales.
- **Relevante:** los objetivos específicos de un proyecto deben ser importantes en el camino hacia el logro del objetivo general, se deben ajustar y estar orientados hacia la meta final, ya que de lo contrario serán un obstáculo.
- **Limitado en el tiempo**: para se pueda medir y que efectivamente sirva a la concreción del proyecto, los objetivos específicos deben ceñirse a un esquema temporal.

Lo anterior basado en la metodología **SMART** que se puede resumir según se aprecia en la siguiente figura:


**Figura 7:** Objetivos eficaces, a través de la metodología SMART.

Fuente: (Guerrero, s.f.)

Vamos a considerar el caso anterior *InvControl: Sistema Integral de Gestión de Inventarios*, para revisar algunos ejemplos de objetivos específicos correctos basados en la metodología SMART y un par de ejemplos incorrectos:

**Objetivos específicos correctos (basados en SMART):**

1. Implementar una base de datos SQL que permita registrar, consultar y actualizar los productos en inventario, asegurando un tiempo de respuesta inferior a 2 segundos en el servidor local, antes de la tercera semana del proyecto.

   *(Específico, Medible, Alcanzable, Relevante, Temporal)*

2. Diseñar una interfaz web con HTML, CSS y Flask que facilite la gestión de inventarios, incluyendo formularios estilizados para agregar y modificar productos, finalizando esta funcionalidad en un plazo de cuatro semanas.

   *(Específico, Medible, Alcanzable, Relevante, Temporal)*

3. Configurar una base de datos NoSQL para manejar un historial de transacciones de inventario, permitiendo consultar al menos 1.000 registros de forma simultánea, completando la integración en un mes.

   *(Específico, Medible, Alcanzable, Relevante, Temporal)*

4. Implementar mecanismos de ciberseguridad que incluyan validación de usuarios, roles y cifrado de contraseñas, garantizando el cumplimiento de estándares de seguridad antes del despliegue del sistema.

   *(Específico, Medible, Alcanzable, Relevante, Temporal)*

5. Realizar pruebas funcionales y de seguridad en el sistema completo, asegurando que el tiempo de carga de cada módulo no supere los 5 segundos, y documentar los resultados en un informe final durante las dos últimas semanas del proyecto.

   *(Específico, Medible, Alcanzable, Relevante, Temporal)*

**Objetivos específicos mal definidos:**

1. **Hacer una base de datos para los productos.**
   - Este objetivo no es medible ni específico, no tiene un tiempo definido y es demasiado general.
2. **Mejorar la seguridad del sistema.**
   - Este objetivo es ambiguo y carece de detalles sobre qué se entiende por "mejorar", cómo se logrará o en cuánto tiempo se realizará.

### 1.4 Delimitación y alcance del proyecto

La delimitación y alcance del proyecto es una etapa fundamental en la planificación, ya que permite establecer los límites específicos del proyecto y definir de forma clara qué aspectos serán abordados y cuáles no. Esto incluye detallar las funciones, características, tareas y resultados esperados, asegurando que todos los involucrados tengan una comprensión compartida de lo que el proyecto pretende lograr y hasta dónde llegará su intervención.

Determinar o delimitar el alcance del proyecto está enfocado en establecer un marco de trabajo en el que se debe enfocar el desarrollo, debemos tener en consideración que no todas las funcionalidades que se esperan implementar cuentan con el mismo nivel de prioridad o criticidad, es importante comprender que con la delimitación no estamos estableciendo límites al sistema, pues en base a la misma evolución del sistema, la aplicación se encaminará respecto de las expectativas de los usuarios.

El proyecto se debe centrar en lo que es realmente importante, y dejar para más adelante lo demás. Lo más probable es que no se cuente con presupuesto para todo principalmente en etapas iniciales, por lo mismo es fundamental cubrir lo que es efectivamente necesario. Debemos considerar que, al no delimitar el alcance del proyecto, estamos dejando muchos puntos abiertos que pueden dar pie a que el cliente o los usuarios quieran retomarlos, en estos casos, lo que debemos hacer es centrarnos en lo que tenemos estipulado y al finalizar el proyecto se podrá evaluar el ir incorporando o complementando el sistema en base a lo que el cliente pueda ir solicitando.

Al delimitar el alcance, se previenen desvíos o expansiones innecesarias en los objetivos (conocido como *scope creep*[1]) que podrían comprometer el tiempo, los recursos y el éxito del proyecto. Esta claridad facilita la asignación eficiente de recursos y establece expectativas realistas tanto para el equipo de trabajo como para los beneficiarios del proyecto.

[1] En el ámbito de la gestión de proyectos, el scope o alcance del proyecto describe los requisitos y los entregables de un proyecto. La definición del alcance tiene lugar al comienzo del proceso de planificación y debe registrarse en el plan del proyecto, hoja de ruta o brief del proyecto. Una corrupción o arrastre del alcance sucede cuando las solicitudes y los entregables exceden el alcance preestablecido del proyecto. Fuente: (Martins, 2024)

Delimitación y alcance del proyecto en base al caso *InvControl: Sistema Integral de Gestión de Inventarios*:

**Alcance del proyecto:**

Diseño e implementación de una interfaz web amigable que permita a los administradores visualizar, agregar, modificar y eliminar productos del inventario mediante formularios estilizados.

Creación de una base de datos SQL para almacenar y gestionar la información de los productos, con soporte para consultas rápidas y organizadas por categorías.

Configuración de roles y usuarios para garantizar que solo los administradores tengan acceso a las funciones de modificación del inventario, incluyendo validación de credenciales y auditoría de accesos.

Realización de pruebas funcionales y de carga para garantizar el rendimiento del sistema, acompañado de una guía de uso para el administrador del inventario.

Instalación del sistema en un servidor local o en un entorno virtualizado con Docker, asegurando que esté listo para su uso en un plazo máximo de tres meses.

**Delimitación:**

El sistema será diseñado exclusivamente para la gestión del inventario de una tienda de electrónica que maneja un catálogo de hasta 5,000 productos, incluyendo detalles como nombre, precio, cantidad en stock y categoría.

No se integrarán módulos para la gestión de ventas ni para la facturación; únicamente se enfocará en la administración del inventario.

La aplicación será desarrollada para funcionar en un entorno web accesible desde navegadores modernos (Chrome, Firefox, Edge), pero no incluirá una versión móvil específica ni funcionalidades de aplicaciones híbridas.

El proyecto no considerará la integración con sistemas externos, como pasarelas de pago o ERPs, durante esta fase de desarrollo.

El mantenimiento del sistema y la actualización de contenido serán responsabilidad del administrador de inventarios, configurado previamente en el sistema.

Esta delimitación establece claramente los elementos que serán desarrollados y aquellos que están fuera del alcance actual del proyecto, evitando confusiones o expectativas adicionales.

## Pregunta de reflexión

- *¿Cuento con los conocimientos necesarios, para definir correctamente los objetivos del proyecto cumpliendo con la metodología SMART?*
- *¿Soy capaz de identificar el límite del sistema?*
- *¿Puedo establecer adecuadamente el alcance del proyecto de manera tal que cubra realmente la necesidad que he detectado?*

Ilustración 3: Pregunta de reflexión

## 2. Análisis de factibilidad y requerimientos

El análisis de factibilidad y requerimientos es una etapa clave en la planificación de un proyecto, donde se evalúa la viabilidad de su desarrollo y se identifican las condiciones necesarias para su éxito.

Este análisis considera dos aspectos principales:

- **Factibilidad técnica:** Determina si los recursos tecnológicos y las competencias del equipo son suficientes para llevar a cabo el proyecto de acuerdo con los objetivos planteados.
- **Factibilidad operativa:** Analiza si el proyecto puede integrarse eficazmente en el entorno laboral y si satisface las necesidades de los usuarios.

Además, en esta etapa se especifican los **requerimientos funcionales** (funciones y características que debe cumplir el sistema) y **no funcionales** (como seguridad, rendimiento y usabilidad), para asegurar que el producto final sea efectivo y cumpla con los estándares esperados.

**2.1 Factibilidad técnica (tecnologías, herramientas, lenguajes, SGBD).**

El análisis de factibilidad técnica y operativa consiste en evaluar la viabilidad del proyecto en cuanto a los recursos tecnológicos y la capacidad operativa del equipo. En este caso, se toma en cuenta la simplicidad y accesibilidad de las tecnologías, las competencias del equipo de desarrollo y los objetivos específicos del proyecto.

Este análisis nos permite seleccionar herramientas y tecnologías que optimicen el proceso de desarrollo, aseguren el correcto funcionamiento de la aplicación y permitan un mantenimiento accesible en el futuro.


**Figura 8:** Análisis de factibilidad técnica y operativa.

**Fuente:** (Villamar, 2024).

**2.1. Factibilidad técnica (tecnologías, herramientas, lenguajes, SGBD)**

La factibilidad técnica es el análisis que determina si un proyecto puede ser desarrollado y operado con éxito en términos de tecnología y recursos técnicos. Evalúa si las herramientas, infraestructuras, y conocimientos necesarios están disponibles o son accesibles, así como si son adecuados para cumplir con los objetivos del proyecto.

Este análisis incluye aspectos como la compatibilidad y disponibilidad de tecnologías y herramientas, la experiencia y competencias del equipo, los requerimientos de infraestructura, la escalabilidad, el mantenimiento, y la seguridad

del sistema. El objetivo principal de la factibilidad técnica es asegurar que el proyecto pueda implementarse de manera eficiente y sostenible desde el punto de vista tecnológico.

La factibilidad técnica debe considerar los siguientes aspectos:

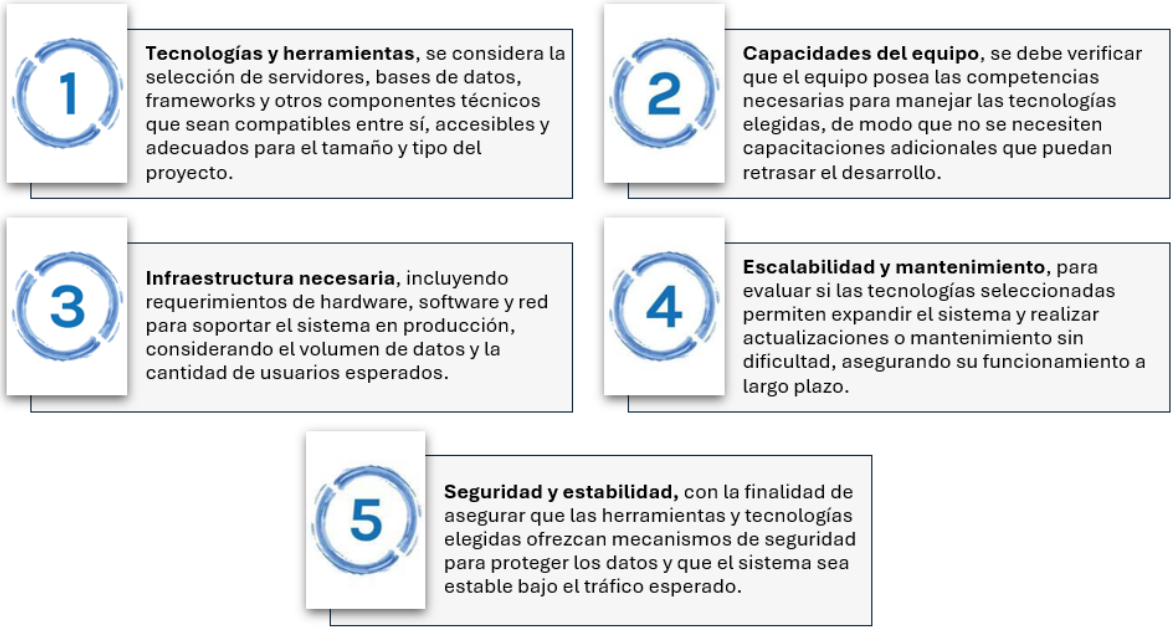


**Figura 6:** Aspectos que considerar en un estudio de factibilidad técnico.

**Fuente:** Elaboración propia.

Estos elementos ayudan a determinar si el proyecto puede ser implementado con éxito desde el punto de vista técnico.

Ejemplo *InvControl: Sistema Integral de Gestión de Inventarios* factibilidad técnica y operativa

A continuación, revisaremos como ejemplo, la factibilidad técnica de *InvControl:*
En este caso se evalúa si *InvControl*, una aplicación web para la gestión de inventarios, puede desarrollarse de manera eficiente en términos de tecnología y operaciones.

**Factibilidad Técnica y Operativa:**

La factibilidad técnica y operativa de *InvControl: Sistema Integral de Gestión de Inventarios* se evalúa desde la perspectiva tecnológica y operativa, considerando las herramientas, tecnologías, habilidades y recursos necesarios para desarrollar y operar el sistema de manera efectiva.

**Factibilidad Técnica:**

La factibilidad técnica del proyecto se centra en la capacidad de desarrollar el sistema *InvControl* con las tecnologías y herramientas necesarias, evaluando las competencias del equipo de desarrollo y la viabilidad de implementar las funcionalidades del sistema.

**Conocimientos y habilidades**:

- El equipo de desarrollo debe estar capacitado en HTML, CSS, y Flask para la creación de interfaces web atractivas y funcionales.
- Deben dominar la integración de bases de datos SQL (PL/SQL), que incluyen la programación de excepciones, cursores explícitos, triggers y funciones avanzadas.
- El manejo de bases de datos NoSQL para operaciones transaccionales, especialmente para gestionar inventarios y transacciones de forma eficiente.
- Competencias en ciberseguridad para garantizar la integridad de los datos almacenados y la protección de la información sensible.
- Capacidad para desarrollar aplicaciones híbridas utilizando Flask para la funcionalidad web y contenedores Docker para la portabilidad.

**Recursos tecnológicos**:

- El equipo debe contar con un entorno de desarrollo compatible con Visual Studio Code, servidores de prueba y almacenamiento en la nube para escalar según los requerimientos del cliente.
- El sistema utilizará SQL Server para la base de datos relacional y MongoDB para la base de datos NoSQL, asegurando flexibilidad y alta disponibilidad de los datos.

**Integración de funcionalidades**:

- Se garantizará la integración de seguridad en las interfaces mediante el uso de autenticación robusta y acceso restringido.

- El sistema podrá realizar consultas avanzadas, gestiones de inventarios, y reportes de datos mediante la integración de BPMN para diagramar flujos de trabajo y procesos de gestión.

**Capacidades del servidor**:

- Se evaluará si los servidores seleccionados (en la nube o físicos) pueden manejar el volumen esperado de usuarios concurrentes y el almacenamiento de grandes cantidades de datos relacionados con inventarios y transacciones.

**Configuración Mínima:**

La configuración mínima define los recursos básicos para el funcionamiento inicial de *InvControl*, permitiendo una implementación a pequeña escala con un costo bajo, y suficiente capacidad para operar con los usuarios y registros básicos.

- Servidor web compartido o en la nube básica que soporte Flask y otras dependencias mínimas.
- Base de Datos SQL o MongoDB en su versión 5.7 o superior, capaz de gestionar las tablas básicas necesarias para controlar los inventarios y las transacciones.
- Almacenamiento en disco duro tradicional de 20 GB para gestionar la base de datos de inventarios con un crecimiento controlado.
- Hosting compartido con capacidad de manejar hasta 50 usuarios concurrentes, con SSL básico para seguridad.
- En cuanto a seguridad Certificado SSL para asegurar las comunicaciones HTTPS.
- Framework y Librerías, uso de Bootstrap para el diseño responsive de las interfaces sin la necesidad de personalización avanzada.

**Configuración Óptima:**

La configuración óptima incluye recursos avanzados y tecnologías que optimizan el rendimiento, la escalabilidad y la seguridad del sistema *InvControl*, permitiendo el manejo de un mayor volumen de usuarios y datos.

- Servidor AWS EC2 o Google Cloud, con escalabilidad automática para manejar aumentos en la carga de usuarios concurrentes.

- Base de Datos SQL Server o MongoDB en la nube, con replicación y backups automáticos para alta disponibilidad y confiabilidad.
- Almacenamiento SSD de 100 GB o más para mayor rendimiento y capacidad de escalabilidad.
- Hosting dedicado o con balanceo de carga que permita manejar miles de usuarios simultáneos sin comprometer el rendimiento.
- En cuanto a seguridad Certificado SSL avanzado, con autenticación de dos factores (MFA) para una mayor protección de los usuarios.
- Framework y Librerías, uso de React o Vue.js para una interfaz dinámica y reactiva, junto con Flask en el backend, Docker para la creación de contenedores que permiten una mayor portabilidad de la aplicación.

**Versiones de Herramientas Recomendadas:**

Para garantizar la estabilidad y compatibilidad en las diferentes fases del proyecto, se recomienda usar versiones estables de las siguientes herramientas:

**Configuración Mínima:**

1. Lenguajes y Frameworks:
    - Flask: Versión 2.1.
    - HTML/CSS/JavaScript: Últimas versiones estables.
    - Bootstrap: Versión 5.0.
2. Base de Datos:
    - MySQL: Versión 5.7 o superior.
    - MongoDB: Versión 4.4.
3. Entorno de Desarrollo:
    - Visual Studio Code: Versión 1.65+.
    - XAMPP: Versión 7.4+ para desarrollo local.
4. Control de Versiones:
    - Git: Versión 2.30 o superior.

**Configuración Óptima:**

1. Lenguajes y Frameworks:
    - Flask: Versión 2.2.
    - React/Vue.js: React 17.0.2 o Vue 3.2.
2. Base de Datos:
    - SQL Server: SQL Server 2019 o superior.
    - MongoDB: Versión 5.0.
3. Entorno de Desarrollo:
    - Visual Studio Code: Última versión con extensiones avanzadas.
    - Docker: Versión 20.10+.
4. Control de Versiones y CI/CD:
    - Git: Versión 2.35+.
    - GitHub Actions o GitLab CI/CD para automatización de pruebas y despliegue.
5. Seguridad:
    - MFA configurado mediante Firebase Authentication o Auth0.

Esta estructura permite que el proyecto *InvControl* se desarrolle de manera eficiente y escalable, desde una versión básica hasta una completamente optimizada para alto rendimiento.

**- Selección de tecnologías y herramientas.**

Para la **selección de tecnologías y herramientas** en la factibilidad técnica, es importante evaluar los siguientes puntos relevantes:

Estos criterios aseguran que las tecnologías seleccionadas cumplan con las necesidades actuales del proyecto y sean sostenibles a largo plazo.

| | |
|---|---|
| **Compatibilidad y Soporte** | Asegurarse de que las herramientas y tecnologías sean compatibles entre sí (por ejemplo, que el lenguaje de programación elegido funcione bien con el servidor web y el sistema de gestión de bases de datos).<br>Verificar que las tecnologías seleccionadas sean compatibles con los sistemas operativos y dispositivos que usará el público objetivo. |
| **Escalabilidad** | Elegir tecnologías que permitan escalar el sistema si aumenta el número de usuarios, el volumen de datos o la funcionalidad en el futuro.<br>Evaluar si las herramientas soportan configuraciones que se adapten al crecimiento del sistema sin comprometer su rendimiento. |
| **Facilidad de Mantenimiento y Actualización** | Seleccionar herramientas que ofrezcan facilidad para actualizaciones, parches de seguridad, y mantenimiento general.<br>Considerar tecnologías con comunidades activas y soporte frecuente, lo que facilita la resolución de problemas y la obtención de actualizaciones. |
| **Costo y Accesibilidad** | Evaluar los costos asociados a las herramientas (licencias, suscripciones) y optar por soluciones de código abierto o de bajo costo si se alinean con los requerimientos del proyecto.<br>Asegurarse de que los recursos y herramientas necesarios para la implementación estén disponibles y accesibles para el equipo de desarrollo. |
| **Curva de Aprendizaje y Competencias del Equipo** | Considerar tecnologías con una curva de aprendizaje adecuada al nivel de experiencia del equipo, para asegurar una implementación ágil.<br>Seleccionar herramientas que el equipo pueda manejar sin requerir capacitación extensa o contratar habilidades adicionales. |
| **Seguridad** | Evaluar si las tecnologías y herramientas ofrecen mecanismos robustos de seguridad, como encriptación de datos, autenticación y autorizaciones seguras, para proteger la información de los usuarios. |
| **Soporte de Comunidad y Documentación** | Preferir tecnologías con una comunidad activa y buena documentación, lo que facilita la resolución de problemas y el acceso a soluciones probadas en otros proyectos similares. |

Ahora bien, habiendo realizado el análisis de factibilidad técnica y revisado si los recursos tecnológicos y las habilidades del equipo son adecuados para implementar este sistema, revisaremos como se realiza la selección de tecnología y herramientas aplicado al proyecto *InvControl*, aplicación web para el control de inventario.

La elección de las tecnologías y herramientas en este caso se basa en la naturaleza del proyecto y en los requisitos de uso y escalabilidad. Las tecnologías seleccionadas son ampliamente accesibles y se encuentran dentro de las competencias del equipo, permitiendo reducir costos y tiempos de desarrollo.

En este análisis, se determinó que la empresa cuenta con los recursos necesarios y habilidades técnicas adecuadas para implementar el sistema. Al cumplir con estos requisitos, el proyecto es técnicamente factible, siempre que se mantenga la infraestructura de soporte, se sigan las mejores prácticas de desarrollo y se realicen exhaustivas pruebas de calidad continuas.

**- Selección de lenguaje de programación justificación elección.**

Para el desarrollo de *BookBuddy*, se ha seleccionado PHP 7.4 como el lenguaje de programación principal para el backend de la aplicación.

Pasaremos a revisar la justificación de la elección:

En primer lugar se ha considerado la ***simplicidad y acceso a recursos*** ya que PHP es un lenguaje accesible, con una sintaxis sencilla, ideal para aplicaciones de pequeño y mediano tamaño, por otro lado, es ampliamente usado en aplicaciones de intercambio y sitios web dinámicos.

Se ha evaluado además la ***compatibilidad*** ya que PHP se integra fácilmente con MySQL, permitiendo la implementación de un sistema de base de datos relacional sin requerir configuraciones complejas.

Por otro lado, se ha analizado el ***manejo de recursos***, PHP es eficiente en entornos de hosting compartido y servidores web básicos como Apache o Nginx, minimizando los costos operativos del proyecto.

También se ha considerado la ***disponibilidad de recursos y soporte***, ya que al ser un lenguaje de código abierto y popular, existen abundantes bibliotecas y documentación para soporte, lo que facilita el desarrollo y el mantenimiento.

**- Selección de SGBD: Justificación de la Elección**

Para almacenar y gestionar la información relacionada con el inventario de productos, usuarios y transacciones, se ha seleccionado MongoDB como el SGBD para *InvControl*.

Las razones que justifican esta elección son las siguientes:

- Escalabilidad y Flexibilidad: MongoDB es una base de datos NoSQL que permite un almacenamiento flexible de datos, dado que el sistema de inventario puede involucrar tipos de datos variables y registros que no siempre se ajustan a un modelo estrictamente relacional, MongoDB proporciona una estructura de documentos que facilita la ampliación del sistema sin necesidad de reestructurar la base de datos cada vez que se agregan nuevos tipos de información o características.
- Rendimiento en Entornos de Alta Carga: MongoDB está diseñado para manejar grandes volúmenes de datos de forma eficiente, gracias a su capacidad de distribuir la carga entre múltiples servidores (sharding), puede escalar horizontalmente, lo que es fundamental cuando el inventario o el número de usuarios crece significativamente.
- Integración con Node.js: MongoDB se integra de manera excelente con Node.js, utilizando bibliotecas como MongoDB para gestionar esquemas y facilitar la interacción con la base de datos, esta integración asegura una comunicación fluida entre el backend y la base de datos, optimizando el rendimiento y reduciendo el tiempo de desarrollo.
- Alta Disponibilidad y Resiliencia: MongoDB ofrece características avanzadas de replicación y alta disponibilidad, lo que garantiza que la base de datos esté siempre accesible, incluso en caso de fallos de hardware o software, esto es esencial para asegurar la continuidad del servicio en aplicaciones críticas como un sistema de control de inventarios.
- Seguridad y Gestión de Accesos: MongoDB proporciona mecanismos robustos de autenticación, control de acceso basado en roles y cifrado de datos en reposo, lo que asegura la protección de la información sensible tanto de los usuarios como del inventario gestionado.
- Comunidad y Soporte: Al ser una de las bases de datos NoSQL más populares, MongoDB cuenta con una gran comunidad de desarrolladores, recursos educativos, y documentación disponible, esto facilita la resolución

de problemas y el mantenimiento del sistema, permitiendo a los desarrolladores encontrar soluciones de manera rápida y eficiente.

Con esta selección de tecnologías, el proyecto *InvControl* está preparado para ser una solución escalable, flexible y eficiente, adaptándose tanto a las necesidades actuales como a posibles expansiones futuras.

### 2.2. Factibilidad Operativa

La factibilidad operativa es el análisis que determina si un proyecto puede ser implementado y mantenido con éxito dentro de las operaciones diarias de una organización o comunidad. Evalúa si los recursos, procesos, y capacidades operativas actuales son suficientes para soportar el proyecto una vez desarrollado y si este es viable y útil para sus usuarios.

Este análisis incluye aspectos como la disponibilidad de personal para operar y mantener el sistema, la adaptabilidad de los procesos existentes para integrar el nuevo proyecto, la capacitación requerida para los usuarios, el soporte y la asistencia técnica, y la alineación del proyecto con las necesidades y expectativas de los usuarios. Su objetivo es asegurar que el proyecto pueda integrarse sin problemas y mantenerse operativo a largo plazo, cumpliendo su propósito eficazmente.

A modo de ejemplo revisaremos la factibilidad operativa de *InvControl:*

En cuanto a las capacidades y competencias del equipo, se ha establecido que el equipo de desarrollo tiene experiencia en las tecnologías seleccionadas (Node.js, MongoDB, HTML, CSS, JavaScript y herramientas relacionadas con la administración de servidores). Esto garantiza que el desarrollo se llevará a cabo de manera eficiente, y que los miembros del equipo podrán solucionar problemas de implementación sin necesidad de capacitaciones adicionales, optimizando el tiempo y recursos del proyecto.

Con respecto a los requerimientos de infraestructura, para el servidor web se necesita un servicio de alojamiento confiable que soporte Node.js, MongoDB y servicios relacionados con aplicaciones web escalables. Además, el sistema debe ser capaz de gestionar grandes volúmenes de datos y tráfico creciente, por lo que se recomienda una infraestructura basada en cloud computing, como AWS o Google Cloud, que ofrezca capacidades de escalabilidad bajo demanda.

Las características necesarias incluyen soporte para aplicaciones web

modernas, infraestructura que permita la implementación de bases de datos NoSQL como MongoDB, y una red de servidores distribuidos para garantizar alta disponibilidad y resiliencia. Además, se requiere que el servicio de alojamiento ofrezca certificados SSL para asegurar las comunicaciones entre el usuario y el sistema, protegiendo los datos sensibles del inventario y los usuarios.

En cuanto al mantenimiento y soporte, debido a que Node.js y MongoDB son tecnologías de código abierto, las actualizaciones y el mantenimiento del sistema podrán realizarse de manera periódica sin incurrir en costos elevados. Esto permite mantener el sistema actualizado con las últimas características de seguridad y optimización, asegurando una operación continua sin grandes inversiones adicionales en infraestructura o licencias.

Lo que debemos tener presente al momento de realizar la factibilidad técnica y operativa del proyecto es lo siguiente:

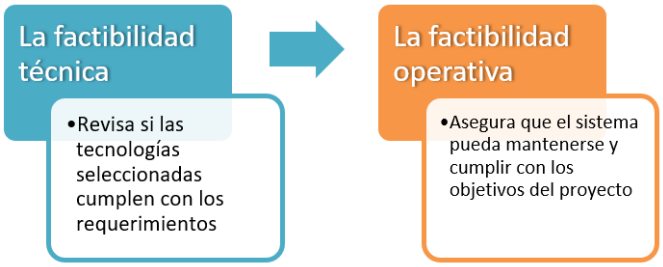


**Figura 7:** Consideraciones en análisis de factibilidad.

**Fuente:** (Villamar, 2024)

### 2.3 Definición de requerimientos funcionales y no funcionales.

#### Requerimientos Funcionales

Describen la funcionalidad o los servicios que se espera que el sistema proveerá. Dependen del tipo de software, del sistema que se desarrolle y de los posibles usuarios.

Veremos a continuación como se definen los requerimientos funcionales según sean expresados:

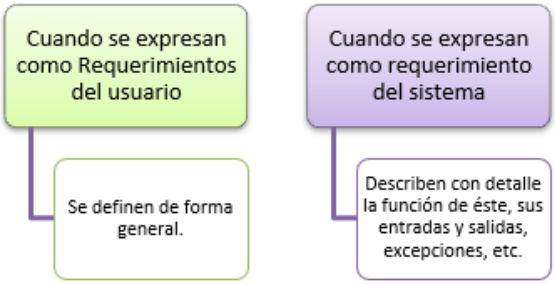


**Figura 8:** Descripción de cómo se expresan los requerimientos.

**Fuente:** (Villamar, 2024)

Los requisitos funcionales son declaraciones de los servicios que proporcionará el sistema, la manera en que reaccionará a entradas específicas.

**¿Cuáles pueden ser estas entradas?**

Ya que al mencionar las entradas no solamente se hace referencia a las entradas de los usuarios, podemos considerar:

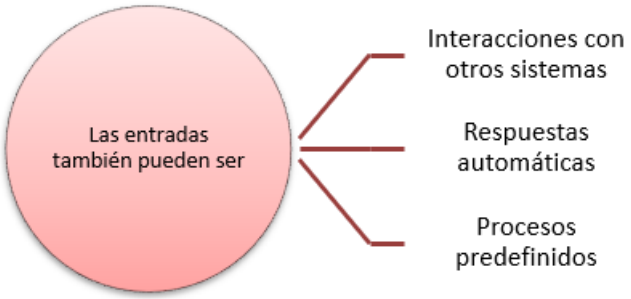


**Figura 9:** Tipos de entradas.

**Fuente:** (Villamar, 2024)

Hay veces en las que los requisitos funcionales de los sistemas también deben indicar de manera clara lo que el sistema no debe hacer. Es importante tener en consideración que un requerimiento funcional puede ser también una declaración negativa, siempre y cuando el resultado de su comportamiento implique una respuesta funcional al usuario, o bien, a otro sistema, es correcto y necesario definirlo.

El origen de muchos problemas en el desarrollo de proyectos se haya en las especificaciones, debido a que se encuentran incorrectas, incompletas, o ambiguas, lo que da pie a que el analista deba suponer ciertas cosas, lo que generalmente es realizado considerando la alternativa de solución más sencilla para simplificar su implementación, pero no necesariamente se ajusta a lo esperado.

Vamos a tomar como ejemplo una historia de usuario desde (medium, 2018):

> *Como usuario, quiero que la aplicación sea fácil de usar, de modo que no tenga que pasar mucho tiempo aprendiendo a usarla.*
>
> - *¿Qué significa ser "fácil de usar"?*
> - *¿Fácil para quién?*
> - *¿Cómo se mide?*
> - *¿Cómo lo rastreas?*
> - *¿Cómo se prueba? ¿Contra qué criterios?*

Podemos apreciar que lo que nos dice el usuario resulta muy ambiguo y general, lo que da pie a que cada persona lo pueda interpretar de manera distinta, asumiendo algo que para otros puede ser distintos, en la definición de requerimientos es precisamente lo que debemos evitar.


**Figura 10:** Características de la especificación de los requisitos funcionales.

**Fuente:** (medium, 2018)

Vamos a revisar el ejemplo que plantea (ingenieriadesoftware, s.f.) en el que se presentan los requerimientos funcionales de una cafetería:

- El cliente podrá ver los productos que se encuentran disponibles.
- Los precios de los alimentos estarán visibles.
- Se pedirá los alimentos en un lugar específico (tanto empaquetados como por preparar) y pagaran en el mismo lugar (caja).
- Los alimentos empaquetados se entregarán en la caja.
- Los alimentos a preparar se pedirán en su lugar asignado.
    - Fruta, gelatinas, etc.
    - Tortas, hamburguesas, etc.
- Los clientes recibirán sus productos, una vez terminado el proceso.
- Se podrá pedir el servicio de calentamiento para alimentos.
- El administrador podrá ver las ganancias.
- Los vendedores entregaran los alimentos empaquetados.
- Los ayudantes de los cocineros entregaran los alimentos para preparar.
- El administrador se encargará de los proveedores.

Lo anterior se puede representar de acuerdo con la siguiente imagen:

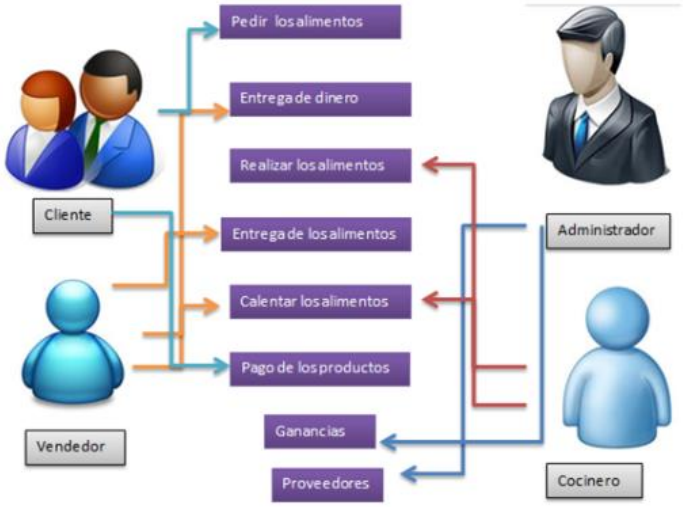


**Figura 11:** Ejemplo requerimientos cafetería.

**Fuente:** (ingenieriadesoftware, s.f.)

**Requerimientos No Funcionales**

Los requerimientos no funcionales son aquellos que se encuentran enfocados en las cualidades, características o restricciones del sistema como son la fiabilidad, tiempo de respuesta o usabilidad, no hay una mención directa respecto de las funciones específicas que debe realizar el sistema, es decir, no hablan de “lo que” hace el sistema, sino de “cómo” lo hace. Encontramos que una gran cantidad de requerimientos no funcionales hacen una referencia del sistema como un todo más que a características particulares del mismo.

Los requerimientos no funcionales podemos catalogarlos como mucho más críticos al momento de haber cualquier incumplimiento, ya que si nos encontramos frente a un incumplimiento de un requerimiento funcional lo que genera es una degradación del sistema en sí, en cambio sí nos enfrentamos a un incumplimiento de un requerimiento no funcional veremos que lo inutiliza.

Los requerimientos no funcionales se clasifican según su implicancia:

- Del producto
- Organizacionales
- Externos

Encontramos que generalmente los requerimientos no funcionales son difíciles de verificar, lo ideal es los requerimientos no funcionales se detallen cuantitativamente empleando métricas que permitan ser probadas de forma objetiva, algo que resulta difícil y posee un costo muy alto asociado, por lo mismo debemos ser muy cuidadosos al momento de documentarlos.

Existen diferentes tipos de requisitos y se clasifican según sus implicaciones.

- Requisitos del producto
- Requisitos organizativos
- Necesidades externas

Al igual que sucede con los requisitos funcionales, generalizar nunca será una buena alternativa, pero en particular en este caso resulta ser aún más importante ya que el resultado de un desarrollo de requisitos no funcionales puede resultar que no es explicito para el usuario final.

A continuación, revisaremos un mal ejemplo y un buen ejemplo de requisito no funcional que presenta (medium, 2018):

*Mal Ejemplo de requisito no funcional:*

*El sistema debe ser seguro.*

- *¿Qué tan seguro es "seguro"?*
- *¿En qué situaciones?*
- *¿Existe una norma a cumplir?*
- *¿En qué secciones?*
- *¿Qué debe suceder si el sistema no puede funcionar tan rápido como se requiere?*

*Buen ejemplo de requisito no funcional:*

*Todas las comunicaciones externas entre los servidores de datos, la aplicación y el cliente del sistema deben estar cifradas utilizando el algoritmo RSA (sistema criptográfico de clave pública).*

- *Saber qué tipo de comunicaciones necesitan ser encriptadas.*
- *Se sabe qué algoritmo usar y validar.*

A continuación, revisaremos por medio de la figura los requerimientos no funcionales:

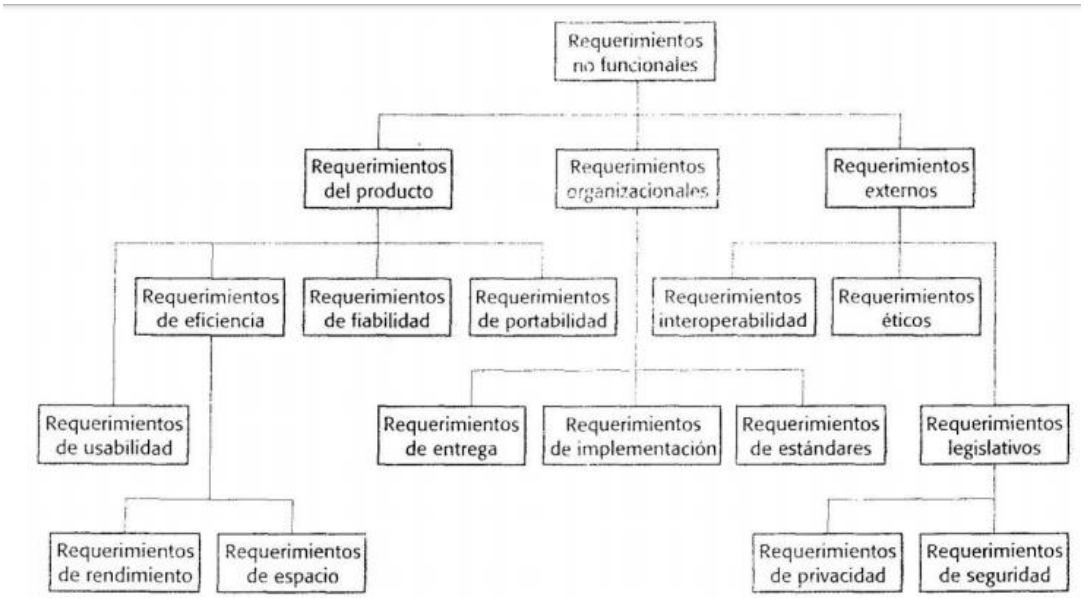


**Figura 12:** Tipos de requerimientos no funcionales.
**Fuente:** (Sommerville, 2005)

La ingeniería de requerimientos[2] comprende siete tareas distintas: concepción, indagación, elaboración, negociación, especificación, validación y administración. Es importante tener en consideración que algunas de estas tareas se realizan paralelamente y la totalidad de ellas se adecúan a las necesidades de cada proyecto.

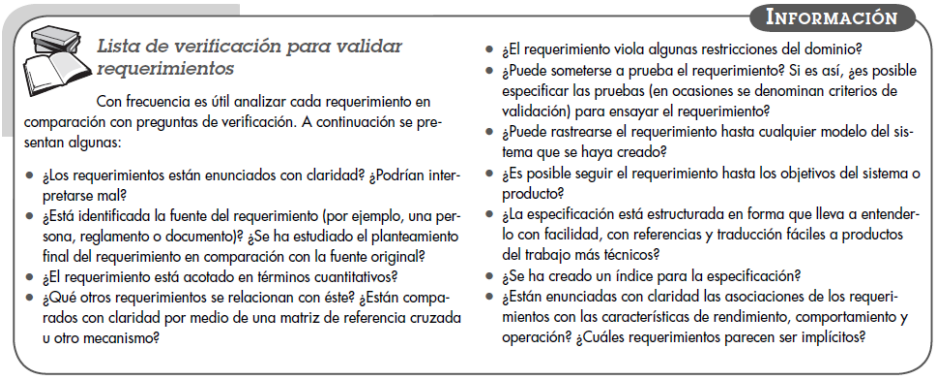


**Figura 13:** Lista de verificación para validar requerimientos.
**Fuente:** (Pressman, 2010)

Para sintetizar lo anterior y podamos comprender la importancia y el impacto de esta etapa, se resume de la siguiente manera:

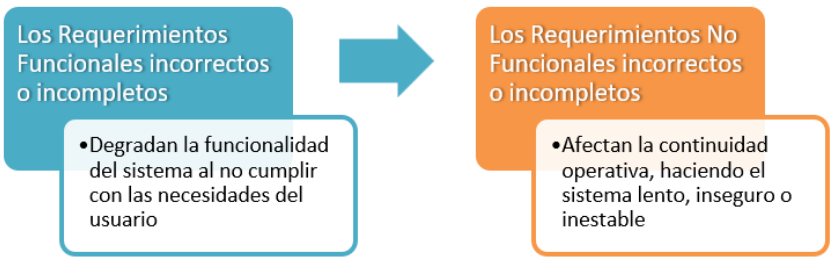


**Figura 14:** Impacto de RF y RNF incompletos o incorrectos.
**Fuente:** Elaboración propia.

[2] Proporciona el mecanismo apropiado para entender lo que desea el cliente, analizar las necesidades, evaluar la factibilidad, negociar una solución razonable, especificar la solución sin ambigüedades, validar la especificación y administrar los requerimientos a medida que se transforman en un sistema funcional. Fuente: (Pressman, 2010)

### 2.4 Planificación preliminar (Carta Gantt).

Una carta Gantt es una herramienta de planificación de proyectos por medio de la cual se enseñan las diferentes tareas con sus tiempos asociados y personas a cargo.

Este tipo de documentos permiten conocer en todo momento qué tareas se deben realizar de forma muy sencilla.

En la actualidad es uno de los métodos de organización del trabajo más utilizados en la gestión de proyectos y de equipos de trabajo. Se puede considerar como la hoja de ruta o la carta de navegación principal del equipo, en la que cada participante puede ver qué es lo que tiene que hacer.

Al utilizar una planificación por medio de una carta Gantt podemos:

- Plasmar el Plan de Acción
- Ordenar las actividades
- Definir el tiempo asignado para realizar cada tarea
- Definir con claridad las tareas asignadas a cada persona
- Entrega una visión general
- Es fundamental para buen plan de acción ya que permite desglosar en detalle las acciones

A continuación, la figura nos presenta un ejemplo de planificación por medio de una Carta Gantt.

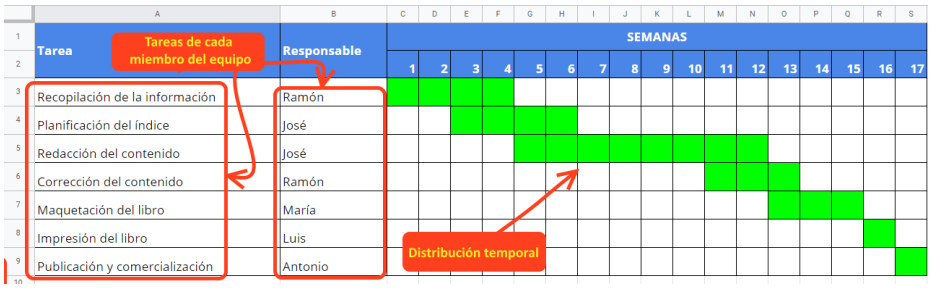


**Figura 15:** Ejemplo Carta Gantt
**Fuente:** (redelaldia, s.f.)

## Preguntas de reflexión

- *Cuando nos encontramos en las primeras etapas de un proyecto de desarrollo ¿en qué nos debemos basar para escoger la tecnología que vamos a utilizar?*
- *¿Cómo podría afectar en el desarrollo del proyecto el considerar requisitos incorrectos o incompletos?*

# Conclusión

Durante esta semana, hemos estudiado aspectos esenciales para la definición de un proyecto de desarrollo, desde la perspectiva de un técnico en informática. El principal desafío ha sido identificar y satisfacer una necesidad específica de los usuarios mediante una propuesta innovadora, integrando de manera estructurada los aspectos técnicos y metodológicos clave que guiarán cada fase del proyecto.

Este proceso requiere que la propuesta no solo responda a esa necesidad, sino que también esté fundamentada en principios técnicos claros y en metodologías adecuadas para garantizar un desarrollo eficiente y alineado con los objetivos del proyecto. Es crucial que la definición del proyecto sea concreta y libre de ambigüedades, para asegurar una planificación precisa y una implementación efectiva de lo aprendido.

Con estos elementos, avanzaremos en la especificación de los requerimientos del sistema, así como en su diseño e implementación. De esta forma, nos aseguramos de que cada etapa del proyecto esté alineada con los objetivos iniciales y logre el impacto esperado para los usuarios finales, contribuyendo al éxito de la solución propuesta.

# Referencias


1. Guerrero, M. (s.f.). *Monge Guerrero*. Obtenido de http://mongeguerrero.com/objetivos-eficaces-a-traves-de-la-metodologia-smart

2. Martins, J. (02 de 03 de 2024). *asana*. Obtenido de https://asana.com/es/resources/what-is-scope-creep

3. Pressman, R. S. (2010). *Ingenieria de Software Un Enfoque Practico.* Mcgraw-Hill.

4. Sommerville, I. (2005). *Ingenieria de Software.* Madrid: Pearson.

# Iplacex

enovus

4 AÑOS

INSTITUCION ACREDITADA

NIVEL AVANZADO

Hasta octubre 2025

Comisión Nacional de Acreditación

CNA - Chile

GESTIÓN INSTITUCIONAL Y DOCENTE DE PREGRADO