



INGENIERÍA DE REQUISITOS

# ANÁLISIS DE NECESIDADES

# ANÁLISIS DE NECESIDADES

## Diferencia entre necesidades, deseos y requerimientos



El análisis de necesidades es uno de los componentes fundamentales en el desarrollo de software y la ingeniería de sistemas, dado que permite determinar lo que verdaderamente se necesita para que el sistema o producto, sea efectivo. Para entender mejor este proceso, es crucial distinguir entre tres conceptos claves: necesidades, deseos y requerimientos, los cuales son fundamentales para definir qué debe hacer el sistema.

### Necesidades

Las necesidades son las características fundamentales que el sistema debe satisfacer para ser considerado útil y eficaz. Son imperativas para el funcionamiento del sistema y, en general, están relacionadas con las problemáticas más profundas de los usuarios o stakeholders. Estas necesidades son las que se derivan directamente de los objetivos y problemas a resolver y, en muchos casos, no son negociables.

Ejemplo. En un sistema de gestión de inventarios, una necesidad puede ser que el sistema permita realizar un seguimiento en tiempo real del stock disponible para evitar desabastecimientos. Esto es esencial para el funcionamiento del negocio.

### Deseos

Los deseos, por otro lado, son aspectos adicionales que los usuarios pueden considerar importantes, pero que no son estrictamente necesarios para que el sistema cumpla con su función principal. Son características que mejorarían la experiencia del usuario, pero que no afectan la operatividad básica del sistema. Normalmente, los deseos pueden quedar fuera del alcance en fases iniciales de desarrollo, puesto que no son imprescindibles.

Ejemplo. Siguiendo el ejemplo del sistema de gestión de inventarios, un deseo podría ser la integración con redes sociales para realizar promociones en tiempo real sobre productos con poca rotación. Aunque este deseo es atractivo, no es esencial para el funcionamiento del sistema.

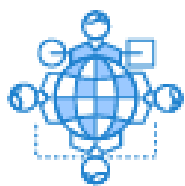
### Requerimientos

Los requerimientos, finalmente, son las especificaciones detalladas que describen cómo debe comportarse el sistema para cumplir con las necesidades identificadas. Estos son los elementos concretos que se desarrollarán y medirán en el sistema. Un requerimiento es un conjunto de características específicas, que describen lo que el sistema debe hacer, y puede ser funcional o no funcional.

Ejemplo. Un requerimiento funcional para el sistema de inventarios podría ser: "El sistema debe permitir al usuario agregar, editar y eliminar, productos del inventario a través de una interfaz de usuario intuitiva". Un requerimiento no funcional podría ser: "El sistema debe ser capaz de procesar hasta 1000 transacciones de inventario por minuto, sin afectar su rendimiento".

Comprender la diferencia entre necesidades, deseos y requerimientos, es fundamental para realizar un análisis eficaz y enfocado. Las necesidades dictan lo que es esencial, los deseos agregan valor, y los requerimientos especifican cómo debe cumplirse todo ello. Este enfoque ayuda a priorizar y gestionar las expectativas de los stakeholders y asegurar que el proyecto cumpla con lo esencial, sin desviarse en funcionalidades innecesarias.

## Técnicas de priorización de necesidades (MoSCoW, Kano, matriz de impacto)



En el análisis de necesidades, no todas las necesidades y requerimientos tienen la misma importancia. Algunas son esenciales para el éxito del sistema, mientras que otras pueden ser implementadas en fases posteriores. La priorización de necesidades es el proceso mediante el cual se determinan cuáles son los elementos más críticos para el proyecto. Existen varias técnicas que ayudan a llevar a cabo, esta priorización de manera eficiente.

### MoSCoW

La técnica MoSCoW es un enfoque ampliamente utilizado para priorizar requerimientos en proyectos de desarrollo de software. El acrónimo MoSCoW se desglosa en las siguientes categorías:

**M (Must have).** Necesidades imprescindibles que el sistema debe tener. Son requisitos esenciales sin los cuales el sistema no podría funcionar.

**S (Should have).** Requisitos importantes, pero no críticos. Son funcionalidades que mejoran el sistema, pero que no afectan su funcionamiento básico.

**C (Could have).** Funcionalidades deseables, pero no necesarias. Se incluyen si el tiempo y los recursos lo permiten.

**W (Won't have).** Requisitos que no se incluirán en esta fase del proyecto, pero que podrían ser considerados en el futuro.

**Ejemplo.** Para un sistema de compras en línea, un requisito "Must have" podría ser la capacidad de realizar pagos, mientras que un "Should have" podría ser una recomendación de productos basada en el historial de compras.

### Kano

El modelo Kano es otra técnica popular para priorizar requisitos, enfocándose en la satisfacción del usuario.

Para poderlo categorizar, se hace a través del análisis de las preguntas recolectadas. La siguiente tabla, es un ejemplo de cómo se puede hacer el análisis a partir de las respuestas de los usuarios. Hay otros ejemplos de tablas, pero todas tienen un significado casi igual y lo único que cambia son las nomenclaturas.

**Tabla 1.** Ejemplo de Modelo Kano

| Respuesta a una pregunta funcional | 1. Me gusta   | 2. Lo espero | 3. Soy neutral | 4. Puedo tolerarlo | 5. No me gusta |
|------------------------------------|---------------|--------------|----------------|--------------------|----------------|
| 1. Me gusta.                       | Cuestionable. | Delighter.   | Delighter.     | Delighter.         | Performance.   |
| 2. Lo espero.                      | Inverso.      | Indiferente. | Indiferente.   | Indiferente.       | Básico.        |
| 3. Soy neutral.                    | Inverso.      | Indiferente. | Indiferente.   | Indiferente.       | Básico.        |
| 4. Puedo tolerarlo.                | Inverso.      | Indiferente. | Indiferente.   | Indiferente.       | Básico.        |
| 5. No me gusta.                    | Inverso.      | Inverso.     | Inverso.       | Inverso.           | Cuestionable.  |

Tal como muestra la tabla, las respuestas a las preguntas están divididas entre funcionales (cómo se sentirían los usuarios si una funcionalidad específica estuviera presente) y disfuncionales (cómo se sentirían si no estuviera presente). Según la opción de respuesta que seleccione el usuario de las cinco opciones posibles:

1. Me gusta,
2. Lo espero,
3. Soy neutral,
4. Puedo tolerarlo,
5. No me gusta, sabremos en cuál de las categorías podemos poner cada una de las funcionalidades sobre las cuales hemos hecho preguntas (una pregunta funcional y una disfuncional por cada funcionalidad).

**Requerimientos básicos.** Son aquellos que se dan por supuestos. Si no están presentes, los usuarios estarán insatisfechos, pero su presencia no genera satisfacción adicional.

**Requerimientos de desempeño.** Son aquellos que los usuarios esperan y cuya mejora aumenta la satisfacción.

**Requerimientos encantadores.** Son características que no se esperan, pero que pueden causar una gran satisfacción si se incluyen.

**Requerimientos indiferentes.** Son características que no tienen un impacto significativo en la satisfacción del usuario.

**Requerimientos reversos.** Son aquellos que pueden causar insatisfacción si se implementan.

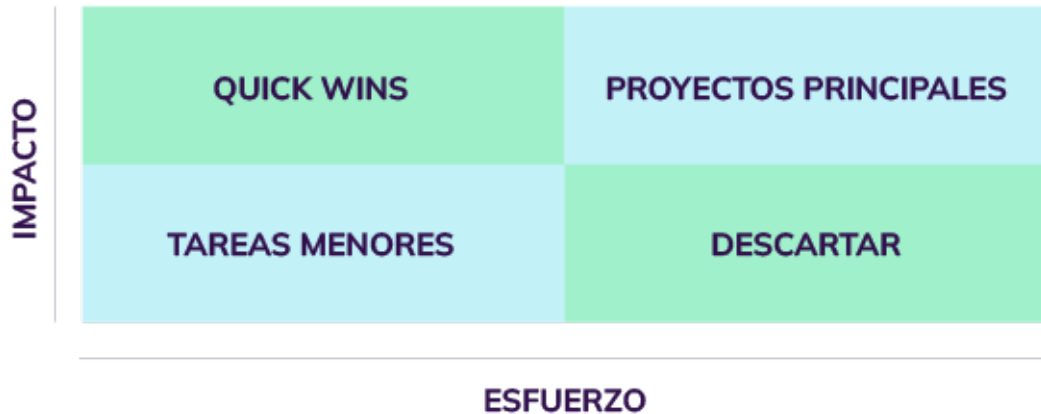
**Ejemplo.** En un sistema de mensajería, un requerimiento básico sería poder enviar y recibir mensajes. Un requerimiento encantador podría ser la capacidad de personalizar el fondo de las conversaciones.

## Matriz de impacto



La matriz de impacto permite evaluar y priorizar los requisitos en función de su impacto en los objetivos del proyecto y la facilidad con la que pueden ser implementados. En esta matriz, se consideran dos factores claves: el impacto de cada requisito en los objetivos del negocio y el esfuerzo necesario para implementarlo.

**Figura 1.** Matriz de impacto



**Ejemplo.** En un proyecto de desarrollo de un sistema de reservas, un requisito de "pago seguro" tendría un alto impacto, pero también podría ser difícil de implementar. Un requisito de "historial de reservas" podría tener un menor impacto, pero sería más fácil de implementar.

La priorización de las necesidades permite tomar decisiones fundamentadas sobre qué requisitos implementar primero, optimizando los recursos y alineando el proyecto con los objetivos estratégicos. Herramientas como MoSCoW, Kano y la matriz de impacto, son esenciales para tomar decisiones informadas y garantizar que el sistema cumpla con las expectativas de los stakeholders.

## Identificación de problemas actuales y oportunidades



La identificación de problemas y oportunidades, es un paso crucial dentro del análisis de necesidades, porque permite detectar las áreas que requieren intervención en el sistema actual. Este proceso ayuda a comprender cómo los usuarios experimentan los problemas y qué mejoras son percibidas como valiosas. Además, la identificación precisa de problemas y oportunidades permite enfocar los esfuerzos del proyecto en las áreas con mayor impacto.

### Identificación de problemas actuales

Para identificar los problemas actuales, es necesario recopilar información tanto de los usuarios como de los sistemas existentes. Se deben realizar análisis exhaustivos sobre los procesos actuales, las interacciones de los usuarios con el sistema y las fallas o insuficiencias observadas.



**Ejemplo.** En una empresa de atención al cliente, un problema identificado podría ser la lentitud con la que los agentes procesan las solicitudes, lo que lleva a largos tiempos de espera para los clientes. Este problema es una ineficiencia que debe ser abordada en el diseño de la solución, ya sea automatizando ciertas tareas o mejorando la interfaz de usuario del sistema.

## Identificación de oportunidades

Las oportunidades se refieren a las áreas en las que se pueden lograr mejoras significativas que no solo resuelvan problemas, sino que también ofrezcan valor añadido a los usuarios o al negocio. Para identificar estas oportunidades, se debe tener en cuenta la retroalimentación de los usuarios y los avances tecnológicos que puedan ser aprovechados para mejorar el sistema.

**Ejemplo.** Una oportunidad en el mismo sistema de atención al cliente, podría ser la implementación de un chatbot que responda a preguntas frecuentes, liberando a los agentes para que se concentren en problemas más complejos. Este tipo de mejora no solo resolvería un problema de capacidad, sino que también aumentaría la eficiencia del sistema.

La identificación de problemas y oportunidades es esencial para determinar las áreas de intervención que mejorarán la experiencia del usuario y la eficiencia del sistema. Este análisis contribuye para que el proyecto se enfoque en las soluciones más relevantes y de mayor impacto.

## Modelado de procesos actuales (AS-IS)

El modelado de procesos AS-IS, es una técnica utilizada para representar y analizar el estado actual de un sistema o proceso. Permite visualizar cómo funcionan los procesos en la realidad y detectar áreas de mejora antes de implementar una solución. El modelo AS-IS es fundamental para comprender las operaciones existentes y las interacciones de los usuarios con el sistema, sirviendo como base para el desarrollo de procesos futuros (TO-BE).

Para entenderlo mejor, se muestra la siguiente imagen de un proceso AS IS básico de un registro de un usuario.

**Figura 2.** Diagrama de proceso de registro de un usuario



**Nota.** Tomado de Nuvaweb (2023). <https://nuvaweb.com/que-es-el-mapeo-de-procesos-as-is-mejora-la-eficiencia-de-tu-negocio>

Como se puede ver, el diagrama de flujo, AS-IS intenta mapear el proceso que se sigue actualmente, esto permite analizar el proceso desde una visión más global y poder establecer opciones de mejoras añadiendo más opciones.

En el ejemplo de la imagen, se puede ver diferentes opciones de mejora y esto se ha de plasmar en un mapeo de procesos To be.

## Herramientas para modelar procesos AS-IS



Existen diversas herramientas y metodologías que permiten modelar procesos de manera efectiva, tales como diagramas de flujo, diagramas de actividades, o el modelado BPMN (Business Process Model and Notation). Estas herramientas visualizan las actividades del proceso, los actores involucrados, las decisiones y los flujos de información.

## Detección de brechas (gap analysis)

La detección de brechas, o gap analysis, es una técnica utilizada para comparar el estado actual (AS-IS) de un sistema con el estado deseado (TO-BE). Este análisis permite identificar las diferencias entre lo que el sistema actual ofrece y lo que los usuarios o el negocio necesitan para lograr sus objetivos. El gap analysis es esencial para enfocar los esfuerzos del proyecto en las áreas más críticas que requieren intervención.

## Pasos para realizar un gap analysis

El proceso de análisis de brechas, generalmente incluye los siguientes pasos:

- ✓ **Definir el estado futuro (TO-BE).** Es necesario tener claridad sobre lo que se quiere lograr, es decir, el estado ideal del sistema. Este estado debe estar alineado con los objetivos estratégicos del proyecto.
- ✓ **Evaluar el estado actual (AS-IS).** A continuación, se realiza una evaluación del sistema actual para entender cómo funciona en su estado actual.
- ✓ **Identificar las brechas.** Se comparan el estado actual y el futuro para identificar las diferencias. Estas brechas representan las áreas donde el sistema no cumple con las expectativas o necesidades.
- ✓ **Desarrollar un plan para cerrar las brechas.** Una vez identificadas las brechas, se debe desarrollar un plan de acción para abordarlas y cerrar esas diferencias.

**Ejemplo.** Supongamos que un hospital desea mejorar su sistema de gestión de pacientes. El estado deseado (TO-BE) podría ser un sistema que permita a los médicos acceder a la información del paciente en tiempo real desde cualquier dispositivo. El estado actual (AS-IS) podría ser que los médicos solo tienen acceso a los registros a través de estaciones de trabajo fijas. La brecha entre estos dos estados es la falta de acceso móvil a los registros médicos, lo que requiere una solución tecnológica para permitir la movilidad.

El gap analysis es una herramienta vital para identificar las diferencias entre el estado actual y los objetivos futuros. Ayuda a definir claramente las áreas que requieren intervención y permite planificar las soluciones necesarias para mejorar el sistema y cumplir con las expectativas de los usuarios.

## Redacción de requisitos iniciales

La redacción de requisitos iniciales es un paso crucial dentro del análisis de necesidades, debido a que establece las bases sobre las cuales se desarrollarán las funcionalidades del sistema. Estos requisitos reflejan las necesidades y expectativas de los usuarios y stakeholders, y deben ser claros, detallados y comprensibles para garantizar una implementación efectiva.

## Características de los requisitos iniciales

Los requisitos iniciales deben ser específicos, medibles, alcanzables, relevantes y limitados en el tiempo, siguiendo el principio SMART. Además, deben ser comprensibles tanto para los desarrolladores como para los usuarios, evitando ambigüedades que puedan generar confusión durante el proceso de desarrollo.





**Ejemplo.** Un requisito inicial podría ser: "El sistema debe permitir a los usuarios registrar nuevos productos en la base de datos mediante una interfaz gráfica en un plazo no mayor a 5 minutos por producto". Este requisito es específico (registro de productos), medible (tiempo máximo de 5 minutos), alcanzable, relevante (mejora la eficiencia del proceso) y limitado en el tiempo (5 minutos).

## Proceso de redacción de requisitos

El proceso de redacción de requisitos generalmente comienza con la recopilación de información de los stakeholders y la identificación de sus necesidades. A continuación, los requisitos deben ser documentados de manera formal, utilizando una terminología clara y precisa. La validación y revisión de estos requisitos son cruciales para asegurar que se alineen con los objetivos del proyecto y las expectativas de los usuarios.

La redacción de requisitos iniciales es un proceso clave que debe garantizar la claridad y precisión en las especificaciones del sistema. Estos requisitos servirán como la base para el desarrollo y las pruebas, asegurando que el producto final cumpla con las necesidades y expectativas de los usuarios.

## BIBLIOGRAFÍA

-  Pérez Agüera, J. R. (2024). Técnicas básicas de priorización. Gemba.  
<https://www.gemba.es/p/tecnicas-basicas-de-priorizacion>
-  Visure (s.f.). Ingeniería de Requerimientos: Paso a Paso. Visure Solutions.  
<https://visuresolutions.com/es/blog/proceso-de-ingenieria-de-requisitos>
-  nuvaweb. (2023). ¿Qué es el mapeo de procesos AS IS? Mejora la eficiencia de tu negocio. <https://nuvaweb.com/que-es-el-mapeo-de-procesos-as-is-mejora-la-eficiencia-de-tu-negocio>
-  Busquets, C. (s.f.). MoSCoW: Qué es y cómo utilizarlo para priorizar. Ui-From mars-.  
<https://www.uifrommars.com/priorizacion-metodo-moscow/>