TECNOLÓGICA DEL ORIENTE
INSTITUCIÓN DE EDUCACIÓN SUPERIOR

INGENIERÍA DE REQUISITOS

PRIORIZACIÓN DE REQUISITOS

PRIORIZACIÓN DE REQUISITOS

La priorización de requisitos representa una etapa estratégica dentro del proceso de ingeniería de requisitos, en la que se determinan cuáles funcionalidades o características del sistema deben ser desarrolladas primero, en función de su valor para los usuarios, viabilidad técnica y alineación con los objetivos del negocio. Este proceso permite enfocar los recursos disponibles en lo que realmente aporta mayor beneficio, facilitando una toma de decisiones informada ante limitaciones de tiempo, presupuesto o cambios en el entorno del proyecto. A través de la priorización, se optimiza la entrega incremental de valor y se fortalece la planificación de versiones, minimizando riesgos y mejorando la satisfacción de los stakeholders a lo largo del desarrollo del software (Quintero, 2006).

Importancia de la priorización de requisitos

La importancia de la priorización de requisitos, radica en su capacidad para dirigir los esfuerzos del equipo de desarrollo hacia aquellas funcionalidades que generan el mayor valor para el cliente y los usuarios finales. Priorizar los requisitos permite gestionar eficazmente los recursos disponibles: tiempo, personal, presupuesto, al enfocar el trabajo en lo esencial, especialmente en proyectos con restricciones estrictas o con plazos ajustados. En este contexto, la priorización actúa como una herramienta clave para la toma de decisiones estratégicas y la planificación de entregas parciales o incrementales (Quintero, 2006).

Desde una perspectiva práctica, la priorización ayuda a responder adecuadamente a los cambios del entorno o del negocio. Por ejemplo, en un sistema de gestión hospitalaria, si los usuarios finales indican que el módulo de registro de pacientes es más crítico que el de generación de reportes estadísticos, entonces el desarrollo debe centrarse en garantizar la funcionalidad del primero. Esta decisión no solo refleja las prioridades reales del usuario, sino que también reduce el riesgo de entregar un producto que no cumpla con las expectativas esenciales.

Además, la priorización permite establecer acuerdos claros con los stakeholders, evitando malentendidos sobre lo que se entregará primero y asegurando una visión compartida del progreso del proyecto (Echeverri et al., 2013). En entornos ágiles, esta práctica se vuelve aún más relevante, dado que cada iteración o sprint debe entregar un valor tangible, y la correcta selección de los requisitos a desarrollar, es determinante para ello.

En definitiva, priorizar no es simplemente ordenar tareas, sino alinear los requerimientos con los objetivos estratégicos del proyecto y del negocio, lo que repercute directamente en el éxito del sistema desarrollado.

Criterios comunes para priorizar requisitos

Los criterios comunes para priorizar requisitos se centran en los factores claves que guían la selección y el ordenamiento de los requisitos más relevantes dentro de un proyecto de desarrollo de software. En la ingeniería de requisitos, establecer criterios objetivos para priorizar permite tomar decisiones fundamentadas y evitar juicios arbitrarios que podrían afectar el valor del producto final. Estos criterios proporcionan una base para negociar con los interesados y asignar recursos de manera estratégica (Echeverri et al., 2013).

La priorización de requisitos es una actividad clave en la ingeniería de software, puesto que permite establecer un orden lógico y estratégico para su implementación, con base en parámetros objetivos y necesidades del negocio.

A continuación, se describen los criterios más utilizados para llevar a cabo esta tarea:

1. **Valor para el negocio.** Este criterio evalúa la contribución directa de cada requisito a los objetivos estratégicos de la organización. Se priorizan aquellos que generan mayor impacto positivo en los procesos internos, en la rentabilidad o en la satisfacción del cliente.

   - ☑ **Ejemplo.** En un sistema bancario, el desarrollo de transferencias inmediatas tiene mayor prioridad que la personalización de temas visuales, debido a su alta demanda por parte de los usuarios.

2. **Urgencia o criticidad temporal.** Algunos requisitos deben ser implementados con prontitud debido a fechas límite contractuales, compromisos legales o dependencias externas. Este criterio ayuda a identificar lo que no puede postergarse.

   - ☑ **Ejemplo.** Un módulo para emitir certificados electrónicos, se vuelve prioritario si existe una regulación gubernamental que exige su uso a partir de una fecha específica.

3. **Riesgo técnico o complejidad.** Se da prioridad a aquellos requisitos que presentan mayores desafíos técnicos, incertidumbre o posibles bloqueos en fases posteriores. Abordarlos temprano permite mitigar riesgos y garantizar una arquitectura estable.

   - ☑ **Ejemplo.** Si un requisito implica integrar inteligencia artificial con un sistema heredado, puede adelantarse para validar su viabilidad técnica.

4. **Dependencias entre requisitos.** Algunos requisitos no pueden desarrollarse sin que otros estén previamente implementados. Este criterio establece un orden lógico para la ejecución (Quintero, 2006).

   - ☑ **Ejemplo.** Antes de diseñar un reporte mensual de ventas, se requiere contar con el módulo de registro de transacciones completamente funcional.

5. **Frecuencia de uso o recurrencia.** Se priorizan aquellos requisitos que serán utilizados con mayor frecuencia por los usuarios finales, porque su impacto en la experiencia general del sistema, es más significativo.

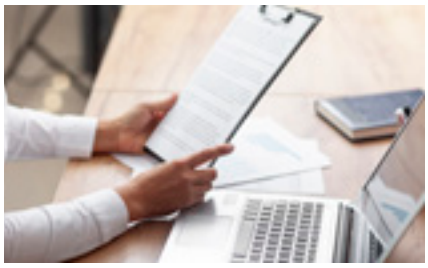
   - ☑ **Ejemplo.** La pantalla de inicio de sesión, debe desarrollarse antes que una funcionalidad de configuración avanzada que se usa esporádicamente.

6. **Expectativas y satisfacción del usuario.** Este criterio considera la percepción de valor desde la perspectiva del usuario final. A veces, aunque un requisito no sea técnicamente prioritario, puede serlo por su impacto en la aceptación del producto.

   - ☑ **Ejemplo.** Una interfaz intuitiva de búsqueda puede recibir mayor prioridad por parte de los usuarios, incluso si no afecta directamente la lógica de negocio.

7. **Costo estimado de implementación.** La estimación de esfuerzo, recursos y costos, puede influir en la decisión de priorizar requisitos más económicos y de alto valor, sobre otros más costosos y de menor impacto.

   - ☑ **Ejemplo.** Si se cuenta con un presupuesto ajustado, podrían desarrollarse primero funcionalidades claves de bajo costo, postergando aquellas que requieran licencias especiales o herramientas adicionales.



Cada uno de estos criterios puede utilizarse de forma independiente o en combinación, según la naturaleza del proyecto, los actores involucrados y los objetivos establecidos (Quintero, 2006). Aplicar una priorización estructurada y basada en estos parámetros, permite tomar decisiones más transparentes, negociadas y alineadas con las metas del desarrollo de software.

Técnicas cualitativas de priorización

Las técnicas cualitativas de priorización, permiten clasificar los requisitos de software con base en criterios subjetivos, valoraciones expertas o consensos entre los actores involucrados. A diferencia de los métodos cuantitativos, no dependen de métricas numéricas ni cálculos formales, sino del juicio humano y el entendimiento compartido del contexto del proyecto. Estas técnicas resultan especialmente útiles en las fases iniciales del análisis, cuando aún no se dispone de información precisa o suficiente para una evaluación numérica.

A continuación, se describen las principales técnicas cualitativas utilizadas en la priorización de requisitos:

1. **Técnica MoSCoW.**

   Esta técnica divide los requisitos en cuatro categorías que representan distintos niveles de prioridad:

   - ☑ **Must have (Debe tener):** requisitos esenciales que son imprescindibles para el funcionamiento básico del sistema.
   - ☑ **Should have (Debería tener):** requisitos importantes, pero no críticos; pueden esperar si hay limitaciones de tiempo o recursos.
   - ☑ **Could have (Podría tener):** funcionalidades deseables, cuya ausencia no afecta el resultado principal del proyecto.
   - ☑ **Won't have (No tendrá por ahora):** requisitos que se acuerda explícitamente dejar fuera del alcance actual.

   Ejemplo. En un sistema de comercio electrónico, la pasarela de pagos puede clasificarse como "Must have", el chat en línea como "Should have", las reseñas de productos como "Could have", y la integración con redes sociales como "Won't have".

2. Priorización por votación o método del grupo nominal.

Se reúne a los interesados y se les otorgan votos (generalmente en forma de puntos, etiquetas o adhesivos) que pueden asignar a los requisitos, según su percepción de importancia. Al final, los requisitos con mayor número de votos se consideran prioritarios (Echeverri et al., 2013).

- **Ejemplo.** En un taller de levantamiento de requisitos con usuarios finales, cada participante dispone de 5 votos para distribuir entre los 10 requisitos más relevantes. Los que reciben más votos pasan a ser considerados de alta prioridad.

3. Clasificación ordinal o jerárquica.

Los requisitos se ordenan de mayor a menor prioridad sin necesidad de asignarles una puntuación. Esta técnica se basa en la comparación relativa entre ellos, permitiendo establecer un ranking simple.

- **Ejemplo.** Un equipo de desarrollo ordena los siguientes requisitos según su impacto: autenticación → historial de compras → filtrado de productos → diseño personalizado. Este orden orienta las etapas de implementación.

4. Método de los tres niveles (alta, media, baja prioridad).

Esta técnica agrupa los requisitos en tres grandes categorías. Aunque es menos precisa, permite una visión general y rápida del nivel de atención que debe recibir cada funcionalidad.

- **Ejemplo.** En un sistema para gestión académica, la inscripción de estudiantes se clasifica como prioridad alta, el envío de recordatorios como media y la personalización de la interfaz como baja.

5. Análisis por consenso.

En este enfoque, los involucrados en el proyecto discuten cada requisito hasta alcanzar un acuerdo común sobre su nivel de prioridad. Esta técnica promueve la colaboración y reduce conflictos posteriores (Echeverri et al., 2013).

- **Ejemplo.** Durante una reunión entre cliente, analistas y desarrolladores, se revisan los requisitos uno por uno y se aprueba su nivel de importancia, en función de los objetivos del negocio y las restricciones técnicas.

Las técnicas cualitativas de priorización, son valiosas cuando el juicio experto y la percepción de valor son más relevantes que los números. Aunque presentan un grado de subjetividad, permiten establecer un punto de partida sólido, especialmente en proyectos ágiles o de naturaleza evolutiva. Para aumentar su efectividad, es recomendable documentar las razones detrás de cada decisión y complementar estas técnicas con métodos cuantitativos cuando el proyecto avance hacia etapas más maduras.

Técnicas cuantitativas de priorización

Las técnicas cuantitativas de priorización, se fundamentan en la asignación de valores numéricos a los requisitos para determinar de manera objetiva su nivel de importancia, urgencia o valor estratégico. A diferencia de los métodos cualitativos, estas técnicas permiten comparar y justificar decisiones mediante cálculos, fórmulas o ponderaciones que reducen la subjetividad en el proceso de priorización. Se utilizan con frecuencia cuando se dispone de información suficiente y se busca transparencia y trazabilidad en la toma de decisiones.

A continuación, se presentan algunas de las técnicas cuantitativas más utilizadas:

1. **Análisis de valor-beneficio vs. costo (Value vs. Cost Analysis).**

   Esta técnica consiste en asignar un valor a cada requisito, según el beneficio que aporta al negocio y otro valor según el costo de su implementación. Se priorizan aquellos requisitos que ofrecen un mayor beneficio por un menor costo (Echeverri et al., 2013).

   - **Ejemplo.** En una aplicación de reservas de citas, el requisito "automatizar recordatorios por correo" tiene un beneficio estimado de 8 y un costo de 3, mientras que "agregar animaciones al perfil del usuario" tiene un beneficio de 3 y un costo de 5. El primero se prioriza por su mayor retorno.

2. **Método de análisis AHP (Analytic Hierarchy Process).**

   El AHP se basa en la comparación por pares de requisitos, donde cada uno se compara con los demás según criterios como valor al usuario, urgencia o riesgo. Luego, se calcula un puntaje ponderado para establecer la prioridad final.

   - **Ejemplo.** Tres requisitos se comparan entre sí en función de su valor para el cliente. Mediante una matriz de comparación, se determina que el requisito A tiene mayor prioridad sobre B y C. El análisis genera una puntuación que refleja esta relación jerárquica.

3. **Priorización basada en puntajes ponderados (Weighted Scoring Model).**

   En esta técnica, se definen criterios (por ejemplo: impacto, urgencia, viabilidad técnica) y se asigna a cada uno un peso según su importancia. Luego, cada requisito se evalúa bajo estos criterios y se calcula una puntuación total.

   - **Ejemplo.** Un equipo define tres criterios: valor al cliente (peso 0.5), facilidad de implementación (0.3), y urgencia (0.2). El requisito X recibe las siguientes calificaciones: 8 en valor, 6 en facilidad y 9 en urgencia. La puntuación total se calcula como (8×0.5) + (6×0.3) + (9×0.2) = 7.3.

4. **Técnica Kano.**

   Aunque originalmente se usa para entender la satisfacción del cliente, esta técnica también se puede aplicar cuantitativamente asignando a cada requisito una categoría (básico, de rendimiento, encantador) y puntuaciones, según encuestas a usuarios.

- **Ejemplo.** Un sistema de banca móvil incluye el requisito "ver estado de cuenta" (básico), "recibir alertas en tiempo real" (de rendimiento) y "modo oscuro" (encantador). A través de una encuesta, se pondera su valor según la reacción de los usuarios y se prioriza en consecuencia.

5. **Cost of Delay (CoD) y Weighted Shortest Job First (WSJF).**

Estas técnicas son muy comunes en metodologías ágiles. El Cost of Delay estima la pérdida económica de retrasar un requisito, y el WSJF divide ese costo entre el esfuerzo necesario para implementarlo. Se prioriza el que tenga mayor relación beneficio/tiempo (Echeverri et al., 2013).

- **Ejemplo.** Si retrasar un requisito genera una pérdida de $10,000 por semana y puede implementarse en dos semanas, su WSJF es 10,000 ÷ 2 = 5,000. Se comparan varios requisitos y se selecciona aquel con el WSJF más alto.

El uso de técnicas cuantitativas de priorización dota al proceso de toma de decisiones de una base sólida, analítica y medible. Aunque requieren tiempo, experiencia y datos confiables, permiten justificar con claridad por qué se priorizan ciertos requisitos sobre otros (Quintero, 2006). En combinación con enfoques cualitativos, las técnicas cuantitativas facilitan una priorización equilibrada, alineada con los objetivos del negocio, las restricciones técnicas y las expectativas de los usuarios.

Participación de los interesados en la priorización

La priorización de requisitos en el desarrollo de software no puede considerarse completa ni precisa sin la activa participación de los interesados (stakeholders). Estos actores, que incluyen usuarios finales, clientes, patrocinadores, responsables técnicos y representantes del negocio, aportan perspectivas fundamentales que enriquecen el proceso de toma de decisiones. Su involucramiento garantiza que las funcionalidades más relevantes sean atendidas primero, asegurando así que el producto final responda a necesidades reales y aporte el máximo valor posible.

Razones claves para involucrar a los interesados:

1. **Diversidad de perspectivas.**

Cada interesado percibe el sistema desde un ángulo distinto. Mientras que un usuario final valora la facilidad de uso, un gerente de proyecto puede enfocarse en la viabilidad técnica y los plazos. La conjunción de estas visiones permite una priorización más equilibrada.

- **Ejemplo.** En el desarrollo de un sistema para una biblioteca digital, los bibliotecarios priorizan la facilidad para gestionar catálogos, mientras que los estudiantes consideran esencial la función de búsqueda avanzada. Ambas opiniones son valiosas para establecer prioridades reales.

2. **Compromiso con el proyecto.**

   Cuando los interesados participan en las decisiones, se sienten escuchados y valorados, lo que aumenta su compromiso y colaboración continua a lo largo del proyecto.

   - **Ejemplo.** En un proyecto de automatización de procesos administrativos, al incluir a los jefes de área en la priorización, estos se muestran más receptivos al cambio tecnológico y facilitan la adopción del sistema.

3. **Validación de valor de negocio.**

   Solo los interesados pueden determinar qué requisitos aportan un valor tangible. Su intervención ayuda a enfocar el desarrollo en funcionalidades que impactan directamente los objetivos estratégicos (Quintero, 2006).

   - **Ejemplo.** Una empresa de e-commerce consulta a su equipo de ventas y al área de atención al cliente para decidir si es más urgente implementar una herramienta de seguimiento de pedidos o mejorar el proceso de devolución.

Formas de participación de los interesados:

1. **Entrevistas y talleres colaborativos.** Se convocan sesiones donde los interesados discuten los requisitos y asignan prioridades en conjunto, generando consenso.

   - **Ejemplo.** Un equipo ágil organiza un workshop con usuarios y desarrolladores donde cada participante vota por los requisitos que considera más valiosos.

2. **Encuestas y cuestionarios ponderados.** Se recurre a formularios que permiten a los interesados calificar la importancia de cada requisito, obteniendo resultados cuantificables.

   - **Ejemplo.** Un equipo recopila los puntajes de 50 usuarios sobre funcionalidades de una app móvil y utiliza esos datos como base para definir el backlog de trabajo.

3. **Tableros de priorización (como MoSCoW o Kano).** Los interesados ayudan a clasificar los requisitos en categorías como "imprescindible", "deseable", "opcional" o "no necesario".

   - **Ejemplo.** En una institución educativa, docentes y alumnos clasifican características de un nuevo sistema académico usando el método MoSCoW.

4. **Métodos democráticos como votación o “dot voting”.** A cada participante se le asigna un número limitado de votos para distribuir entre los requisitos que considera prioritarios.

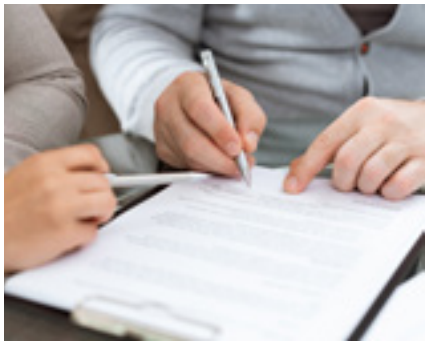
   - **Ejemplo.** Durante una reunión de priorización, los interesados colocan adhesivos en una pizarra para señalar los requisitos que consideran más urgentes.

Retos en la participación de los interesados

Aunque su inclusión es esencial, también presenta desafíos como:

- ☑ Conflictos de interés entre áreas que compiten por recursos.
- ☑ Dificultades para lograr consenso, especialmente en organizaciones grandes.
- ☑ Falta de disponibilidad o interés, lo cual puede sesgar la priorización.

Por ello, el rol del analista de requisitos o facilitador es clave para guiar el proceso, resolver discrepancias y asegurar que todas las voces relevantes sean escuchadas.



La participación de los interesados en la priorización de requisitos no es una opción, sino una necesidad estratégica. Su experiencia, visión y expectativas nutren el proceso con datos reales y alineados al negocio (Quintero, 2006). Cuando se integran sus aportes de manera estructurada y transparente, se incrementa la calidad del producto final, se fortalece la toma de decisiones y se construyen soluciones más satisfactorias y efectivas.

Priorización en proyectos ágiles

En el contexto de los proyectos ágiles, la priorización de requisitos adquiere un enfoque dinámico, iterativo y centrado en la entrega de valor continuo. A diferencia de los enfoques tradicionales donde los requisitos se definen de forma completa al inicio del proyecto, en los entornos ágiles la priorización es un proceso constante que guía la selección de funcionalidades a implementar en cada iteración o sprint. Este modelo busca adaptarse a los cambios del entorno, a las necesidades del cliente y a la evolución del producto, priorizando aquellas tareas que generan un mayor impacto en el negocio o en el usuario final (Echeverri et al., 2013).

Características distintivas de la priorización ágil

1. **Iterativa y flexible.** La priorización no se realiza una sola vez, sino que se ajusta con cada sprint o ciclo de desarrollo. Los requisitos pueden subir o bajar de prioridad según los comentarios del cliente, las condiciones del mercado o cambios técnicos.

   - ☑ **Ejemplo.** En un proyecto de desarrollo de una app de mensajería, se prioriza primero la funcionalidad básica de envío de texto. Luego, tras recibir retroalimentación de usuarios, se adelanta el desarrollo de stickers animados antes de integrar las videollamadas.

2. **Centrada en el valor del negocio.** La principal guía para priorizar es el valor que cada funcionalidad aporta al usuario o a los objetivos comerciales del cliente.

   - ☑ **Ejemplo.** En una startup financiera, se da prioridad al módulo de registro y verificación de identidad, ya que es indispensable para atraer a los primeros usuarios e iniciar operaciones.

3. **Colaborativa.** El equipo de desarrollo, el Product Owner y los interesados trabajan juntos para decidir qué historias de usuario son más importantes, fomentando la transparencia y el consenso.

   - ☑ **Ejemplo.** Durante una reunión de planificación de sprint, el Product Owner consulta al equipo técnico sobre la complejidad de ciertas tareas, mientras que los representantes de usuarios opinan sobre su valor percibido.

Técnicas comunes de priorización en proyectos ágiles

1. **MoSCoW (Must, Should, Could, Won't).** Esta técnica divide los requisitos en cuatro categorías:

   - ☑ **Must have:** esenciales para el éxito del proyecto.
   - ☑ **Should have:** importantes pero no esenciales.
   - ☑ **Could have:** deseables si hay tiempo.
   - ☑ **Won't have:** no se desarrollarán por ahora.

   Ejemplo. En un sistema de reservas en línea, el proceso de pago seguro es un "Must", la función de reseñas de usuarios es un "Should", y la integración con redes sociales es un "Could".

2. **Valor vs. Esfuerzo (Value vs. Effort).** Se pondera el valor de negocio frente al esfuerzo técnico necesario para implementar cada requisito. Se priorizan las funcionalidades de alto valor y bajo esfuerzo.

   - ☑ **Ejemplo.** Si la función "recordar contraseña" requiere poco tiempo de desarrollo y es muy demandada por los usuarios, se prioriza sobre mejoras complejas pero poco urgentes.

3. **Kano Model.** Clasifica las funcionalidades, según el grado de satisfacción que generan: básicas, de desempeño o encantadoras. Este enfoque permite priorizar elementos que generen satisfacción rápida.

   - ☑ **Ejemplo.** En una aplicación educativa, el acceso a contenido es una necesidad básica, mientras que una interfaz adaptativa con logros gamificados puede ser una característica que sorprende y encanta.

4. **WSJF (Weighted Shortest Job First).** Utilizado en marcos como SAFe (Scaled Agile Framework), este método calcula una puntuación en función del costo del retraso y la duración estimada. Se priorizan los elementos con mayor puntuación.

   - ☑ **Ejemplo.** En una empresa que trabaja con múltiples equipos ágiles, se evalúan las historias de usuario en función de su urgencia comercial y complejidad para decidir qué abordar primero.

Ventajas de la priorización ágil

- ☑ Permite adaptarse al cambio de forma rápida y controlada.
- ☑ Facilita la entrega temprana de funcionalidades clave.

- ☑ Promueve una mayor alineación con los objetivos del negocio.
- ☑ Reduce el riesgo de desarrollar funciones innecesarias.

La priorización en proyectos ágiles se convierte en una herramienta estratégica para maximizar el valor entregado en cada iteración. Su carácter colaborativo, enfocado en el cliente y orientado al valor permite a los equipos construir productos útiles, relevantes y ajustados a las condiciones reales del entorno (Echeverri et al., 2013). A través del uso de técnicas ágiles de priorización, se logra una toma de decisiones más ágil, consciente y efectiva, garantizando que el producto evolucione en la dirección correcta sprint tras sprint.

Bibliografía

- ✍🏻 Echeverri, J., Aristizábal, M. & González, L. (2013). Reflexiones sobre ingeniería de requisitos y pruebas de software. Corporación Universitaria Remington. https://elibro.net/es/lc/tecnologicadeloriente/titulos/68913
- ✍🏻 Quintero, J. B. (2006). Un estudio comparativo de herramientas para el modelado con UML. Red Universidad Eafit. https://elibro.net/es/lc/tecnologicadeloriente/titulos/5285