TECNOLÓGICA DEL ORIENTE
INSTITUCIÓN DE EDUCACIÓN SUPERIOR

INGENIERÍA DE REQUISITOS

ESTRATEGIAS CLAVES PARA EL LEVANTAMIENTO DE REQUISITOS EN EL DESARROLLO DE SISTEMAS

ESTRATEGIAS CLAVES PARA EL LEVANTAMIENTO DE REQUISITOS EN EL DESARROLLO DE SISTEMAS

En el proceso de desarrollo de sistemas, el levantamiento de requisitos representa una etapa crítica que determina la calidad y pertinencia del producto final. Para lograr una comprensión profunda de las necesidades de los usuarios y alinear los objetivos del proyecto con las expectativas de los stakeholders, se emplean diversas estrategias. Entre las más destacadas se encuentran el prototipado, los talleres de trabajo colaborativo (workshops) y las técnicas ágiles como las user stories, personas y story mapping. Cada una de estas metodologías ofrece ventajas específicas para la identificación, validación y priorización de funcionalidades, permitiendo que los equipos de desarrollo construyan soluciones más coherentes, eficientes y orientadas al usuario. A continuación, se presentan ejemplos y consideraciones clave para la aplicación efectiva de estas técnicas.

1. Ejemplo de aplicación del prototipado

Imaginemos que se está desarrollando una aplicación móvil para la gestión de tareas y proyectos. Los desarrolladores pueden comenzar con un prototipo de baja fidelidad, utilizando herramientas como bocetos en papel o software como Balsamiq. Este primer prototipo puede incluir una pantalla de inicio, un listado de tareas y botones de acción básicos, sin la necesidad de codificar la funcionalidad completa.

Posteriormente, al obtener comentarios de los usuarios (por ejemplo, sobre la ubicación de los botones, el diseño de la interfaz o la funcionalidad de las tareas recurrentes), el prototipo se convierte en una versión de alta fidelidad utilizando herramientas como Figma o Adobe XD, donde se integran características interactivas y se simula una experiencia de usuario más realista.

Con la retroalimentación obtenida de esta versión, se pueden realizar ajustes, como la adición de nuevas funcionalidades o el rediseño de la interfaz, antes de pasar a la fase de desarrollo real.

Ventajas del prototipado

Las ventajas del prototipado son numerosas, entre las cuales destacan:

1. **Reducción de riesgos.** Al ofrecer una representación visual de los requisitos, el prototipo permite a los desarrolladores identificar posibles fallos de diseño o funcionalidad antes de que se invierta un gran esfuerzo de desarrollo.

2. **Mejor comprensión de los requisitos.** Los prototipos ayudan a los usuarios a expresar sus necesidades de forma más clara, dado que les permiten experimentar con el sistema en un contexto más interactivo. Esto es particularmente útil en proyectos donde los requisitos no están completamente definidos al principio o son ambiguos.

3. **Facilita la comunicación con los stakeholders.** La representación tangible del sistema facilita las discusiones con los stakeholders, quienes pueden ser reacios a comprometerse con descripciones abstractas de las funcionalidades. Un prototipo permite que se concreten las expectativas y se ajuste el alcance del proyecto.

4. **Mayor satisfacción del cliente.** Dado que los usuarios pueden involucrarse de manera activa en el desarrollo del sistema, proporcionando comentarios directos sobre el prototipo, se incrementa la probabilidad de que el producto final cumpla con sus expectativas.

5. **Aceleración del proceso de desarrollo.** El prototipo sirve como base para el desarrollo continuo, reduciendo el tiempo de desarrollo al proporcionar una guía visual clara y una base de funcionalidades probadas en etapas tempranas.

Desventajas y desafíos

Aunque el prototipado ofrece muchas ventajas, también presenta algunas desventajas que deben tenerse en cuenta:

1. **Tiempo y recursos en la creación de prototipos.** El desarrollo de prototipos, especialmente de alta fidelidad, puede consumir tiempo y recursos, lo cual puede no ser justificable en proyectos de bajo presupuesto o con plazos ajustados.

2. **Expectativas poco realistas.** Los usuarios pueden malinterpretar el prototipo como la versión final del producto, lo que podría llevar a expectativas irreales sobre el nivel de funcionalidad. Es importante comunicar claramente que el prototipo es una herramienta para la exploración y no la versión definitiva.

3. **Cambios frecuentes en el diseño.** Si los usuarios piden cambios significativos a lo largo de las fases de prototipado, el proceso puede volverse iterativo en exceso, lo que podría ralentizar el avance del proyecto.

Ejemplo de prototipado en el sector de comercio electrónico

En un proyecto de desarrollo de una nueva plataforma de comercio electrónico, los diseñadores y desarrolladores pueden utilizar prototipos para validar la experiencia de usuario y la interfaz antes de escribir líneas de código. Un prototipo de baja fidelidad puede incluir pantallas en blanco y negro, que muestren cómo se organizarán los productos, las opciones de pago y el proceso de compra.

Después de recibir comentarios sobre la disposición de los elementos en la página, el prototipo de alta fidelidad puede incluir imágenes reales, colores y funcionalidades básicas, como botones interactivos, para simular el flujo de navegación y la compra de productos.

Al final de este proceso, los comentarios del usuario permiten ajustar detalles importantes, como el diseño de las fichas de producto, la disposición del carrito de compras y la accesibilidad de los métodos de pago. Este enfoque iterativo, mejora la calidad final del sistema, asegurando que se cumplan las expectativas de los usuarios y se maximice la usabilidad.

En conclusión, el prototipado es una técnica invaluable en el levantamiento de requisitos, que permite obtener retroalimentación temprana y precisa del usuario, y facilita la identificación de mejoras en etapas iniciales del desarrollo. Su capacidad para reducir riesgos, aclarar requisitos y mejorar la comunicación con los stakeholders lo convierte en una herramienta esencial para el diseño de sistemas orientados a la satisfacción del usuario final. Sin embargo, debe usarse con cuidado, gestionando las expectativas de los usuarios y ajustando su alcance según los recursos y los objetivos del proyecto.

2. Talleres de trabajo colaborativo (workshops)

Los talleres de trabajo colaborativo, comúnmente conocidos como workshops, son una de las técnicas más efectivas para la recolección de requisitos, especialmente en escenarios donde se requiere una interacción directa y dinámica entre los usuarios, los stakeholders y los desarrolladores del sistema. Esta técnica se basa en la participación activa de todos los involucrados, facilitando la generación de ideas, la resolución de dudas y la toma de decisiones en un ambiente colaborativo. Los workshops son particularmente útiles, cuando se necesitan definir requisitos complejos, resolver conflictos o fomentar el consenso en torno a los objetivos del proyecto.

Características de los workshops

Los workshops se caracterizan por ser reuniones intensivas, en las que un grupo de participantes trabaja de manera conjunta durante un tiempo determinado para abordar problemas específicos, generar soluciones y definir aspectos clave del sistema. Estos talleres se realizan en sesiones de trabajo interactivas, donde todos los participantes tienen voz y pueden aportar sus opiniones y conocimientos sobre el tema en cuestión. Los workshops son generalmente facilitados por un moderador o un experto en la materia, que asegura que las discusiones sean productivas y que todos los puntos de vista sean considerados.

Un aspecto crucial de los workshops es que suelen tener un enfoque práctico, orientado a la creación de resultados tangibles. Esto puede incluir la elaboración de diagramas, mapas de procesos, historias de usuario, prototipos rápidos o la identificación de prioridades de desarrollo. Este enfoque práctico asegura que el taller no solo sea una discusión teórica, sino una oportunidad para obtener productos concretos que pueden ser utilizados para guiar el desarrollo del sistema.

Ejemplo de un workshop en el desarrollo de un sistema de gestión de inventarios

Imaginemos que un equipo de desarrollo está trabajando en un sistema de gestión de inventarios para una empresa minorista. Un workshop puede convocarse para reunir a los siguientes participantes:

- ☑ Gerentes de la tienda.
- ☑ Encargados de almacén.
- ☑ Personal de ventas.
- ☑ Desarrolladores y diseñadores del sistema.

El objetivo del taller sería definir las funcionalidades claves del sistema y alinear las expectativas de todos los participantes. Durante el workshop, se podrían realizar las siguientes actividades:

1. **Mapeo de procesos.** Los participantes pueden trabajar en conjunto para crear un diagrama de flujo que represente el proceso actual de gestión de inventarios. Esto permite identificar ineficiencias y áreas de mejora que el nuevo sistema debe abordar.

2. **Priorización de requisitos.** A través de una técnica como el voting o el MoSCoW (Must have, Should have, Could have, Won't have), el grupo puede priorizar los requisitos más importantes para el sistema, asegurando que los recursos se asignen a las funcionalidades críticas.

3. **Generación de historias de usuario.** Los desarrolladores pueden guiar a los usuarios en la creación de historias de usuario, como: "Como encargado de almacén, quiero poder verificar el stock de un producto por código de barras, para asegurarme de que la cantidad registrada es correcta".

4. **Prototipado rápido.** Con la ayuda de herramientas como lápiz y papel, o incluso software de prototipado digital, los participantes pueden diseñar rápidamente, pantallas del sistema que representen la interfaz y las interacciones claves.

Ventajas de los workshops

El uso de workshops en el levantamiento de requisitos, presenta una serie de ventajas, entre las que destacan:

1. **Fomento de la colaboración y el consenso.** Al reunir a diferentes partes interesadas en un espacio común, los workshops promueven la colaboración y el trabajo en equipo. Esto es especialmente importante en proyectos que involucran múltiples departamentos o usuarios con diferentes perspectivas. Los workshops permiten que los participantes comprendan las necesidades y los puntos de vista de otros, ayudando a generar consenso sobre los requisitos.

2. **Definición clara de requisitos.** Dado que los participantes tienen la oportunidad de expresar sus ideas y dudas de forma directa, los workshops permiten definir de manera más precisa y clara, los requisitos del sistema, lo que reduce la posibilidad de malinterpretaciones en etapas posteriores del desarrollo.

3. **Reducción de la brecha entre usuarios y desarrolladores.** Los talleres facilitan la comunicación entre los usuarios y los desarrolladores, eliminando la jerarquía tradicional que puede existir entre estos grupos. Esto permite que los desarrolladores comprendan mejor el contexto y las necesidades de los usuarios, mientras que los usuarios pueden entender las limitaciones tecnológicas y los desafíos asociados al desarrollo del sistema.

4. **Generación rápida de prototipos y soluciones.** Durante los workshops, se pueden generar soluciones tangibles, como prototipos rápidos o diagramas, que sirven como base para el desarrollo posterior del sistema. Estos prototipos permiten a los usuarios evaluar la funcionalidad del sistema de manera temprana, lo que facilita la toma de decisiones informadas.

5. **Eficiencia en la toma de decisiones.** Al reunir a todos los participantes claves en un solo lugar, los workshops permiten tomar decisiones rápidamente y evitar la acumulación de retrasos derivados de la comunicación asincrónica. Esto es crucial en proyectos con plazos ajustados.

Desafíos y consideraciones

Aunque los workshops ofrecen numerosas ventajas, también presentan ciertos desafíos que deben ser considerados:

- **Gestión del tiempo.** Los workshops pueden ser intensivos y consumir una cantidad significativa de tiempo. Es fundamental que el facilitador gestione eficazmente las sesiones, para evitar que las discusiones se desvíen de los objetivos y para asegurar que todos los puntos clave sean cubiertos en el tiempo disponible.
- **Participación activa de todos.** Asegurarse de que todos los participantes, incluidos los usuarios y stakeholders, se involucren activamente puede ser un desafío, especialmente si algunos grupos son menos expresivos o tienen más dificultades para comunicar sus necesidades.
- **Generación de expectativas altas.** Dado que los workshops son altamente interactivos, los participantes pueden generar expectativas elevadas sobre la rapidez con que se solucionarán sus problemas. Es importante aclarar que el objetivo del workshop es definir requisitos y soluciones preliminares, no entregar el producto final en una sola sesión.

Ejemplo adicional. Workshop para un sistema de gestión de recursos humanos

En el caso de un sistema de gestión de recursos humanos, se podría convocar un workshop con gerentes de personal, representantes de nómina y responsables de capacitación. Durante el taller, se podría mapear el proceso de reclutamiento, identificar las características esenciales del sistema (como la capacidad de almacenar información de candidatos, generar informes o automatizar la asignación de tareas), y desarrollar historias de usuario como: "Como reclutador, quiero poder filtrar candidatos por experiencia laboral y habilidades, para agilizar el proceso de selección". Estos talleres permiten obtener información valiosa para crear un sistema alineado con las expectativas y necesidades de todos los grupos.

Los talleres de trabajo colaborativo, son una técnica poderosa en el levantamiento de requisitos, puesto que facilitan la participación activa de todas las partes interesadas en la definición de los requisitos del sistema. A través de un enfoque dinámico y práctico, los workshops permiten identificar necesidades, resolver problemas y tomar decisiones rápidamente, lo que resulta en soluciones más acordes con las expectativas de los usuarios. Si bien existen desafíos en su implementación, los beneficios de colaboración, claridad en los requisitos y generación de soluciones tangibles hacen de los workshops una herramienta esencial en el desarrollo de sistemas.

3. Técnicas ágiles (user Stories, personas, story mapping)

Las técnicas ágiles se han convertido en una metodología estándar para el desarrollo de software, promoviendo flexibilidad, iteración constante y la participación activa

de los usuarios y stakeholders a lo largo del proceso. En este contexto, las user stories (historias de usuario), las personas y el story mapping, son herramientas fundamentales que facilitan la recolección de requisitos y la definición de funcionalidades que responden a las verdaderas necesidades de los usuarios finales. Estas técnicas son especialmente valiosas en proyectos donde el cambio es constante y las expectativas de los usuarios pueden evolucionar con el tiempo.

User stories (historias de usuario)

Las user stories son una técnica clave en el marco ágil que permite capturar los requisitos desde la perspectiva del usuario final. Una historia de usuario describe una función o característica del sistema que el usuario necesita o desea, y se escribe en un formato simple y comprensible para todos los involucrados en el proyecto. El formato más común para una historia de usuario, es el siguiente:

- Como [tipo de usuario], quiero [realizar una acción] para [alcanzar un objetivo].

Este formato ayuda a enfocar las discusiones en los resultados que los usuarios desean obtener, más que en las características técnicas del sistema. El uso de historias de usuario también favorece la creación de un lenguaje común entre los desarrolladores y los usuarios, evitando malentendidos y asegurando que las funcionalidades se alineen con las expectativas del usuario.

Ejemplo de historia de usuario:

- Como usuario administrador del sistema de gestión de inventarios, quiero poder agregar nuevos productos al inventario con su código de barras y cantidad, para facilitar la actualización rápida del stock.

En este caso, la historia de usuario es clara y concisa, identificando al tipo de usuario (administrador del sistema), la acción deseada (agregar nuevos productos), y el objetivo (actualización rápida del stock). Este tipo de descripción ayuda al equipo de desarrollo a comprender el "por qué" detrás de la funcionalidad y a priorizar el trabajo, de acuerdo con las necesidades del usuario.

Personas

La técnica de personas es una herramienta utilizada para representar a los usuarios típicos de un sistema, con el fin de entender sus necesidades, motivaciones y comportamientos. Las personas son representaciones semificticias de los usuarios que podrían utilizar el sistema, basadas en datos reales obtenidos de la investigación y entrevistas con usuarios. Cada persona tiene atributos específicos como nombre, edad, ocupación, metas, frustraciones, habilidades y limitaciones. Al definir estas características, los desarrolladores pueden crear soluciones más alineadas con las necesidades de los usuarios reales.

Ejemplo de una persona:

- Nombre: Laura Gómez.
- Edad: 34 años.
- Ocupación: Gerente de compras.

- Metas: quiere reducir el tiempo que dedica a revisar inventarios y obtener alertas automáticas cuando el stock de productos críticos está por debajo de un umbral determinado.
- Frustraciones: se siente abrumada por la cantidad de tareas repetitivas y carece de tiempo para revisar manualmente el inventario.
- Comportamiento: Laura usa su teléfono móvil para realizar tareas de gestión mientras se desplaza, por lo que requiere que el sistema de inventarios sea accesible desde dispositivos móviles.

Al desarrollar personas como Laura, el equipo de desarrollo puede empatizar con las necesidades y limitaciones de los usuarios, lo que ayuda a crear un sistema que les sea más útil y fácil de usar. Esta técnica también es valiosa para tomar decisiones de diseño y priorización, ya que asegura que las funcionalidades respondan a las necesidades específicas de los usuarios.

Story mapping (mapeo de historias)

El story mapping es una técnica que permite visualizar el flujo de trabajo completo de un sistema, organizando las historias de usuario en un mapa que refleja cómo los usuarios interactúan con el sistema a lo largo del tiempo. Esta técnica ayuda a los equipos a ver la relación entre las distintas funcionalidades y a identificar dependencias, prioridades y puntos de dolor en la experiencia del usuario. El objetivo del story mapping es asegurar que el desarrollo del sistema siga una secuencia lógica y que se entreguen funcionalidades de manera incremental, permitiendo obtener resultados rápidamente y ajustarse a los cambios que puedan surgir.

El proceso de story mapping comienza con la identificación de las historias de usuario clave, que luego se organizan en un mapa visual. En la parte superior del mapa, se colocan las historias de usuario más importantes o de alto nivel (las funcionalidades más amplias o los objetivos principales del sistema), mientras que en la parte inferior se incluyen las historias de usuario más detalladas y específicas. Este enfoque facilita la identificación de tareas y funcionalidades críticas, y permite al equipo de desarrollo enfocarse en entregar las partes más valiosas del sistema primero.

Ejemplo de story mapping:

Supongamos que el equipo está desarrollando un sistema de ventas en línea. En el story mapping, las historias de usuario principales podrían incluir:

- Registrar una cuenta de usuario.
- Agregar productos al carrito.
- Realizar un pago.

Bajo cada una de estas historias principales, se agregan historias de usuario más específicas, como:

- Registrar una cuenta de usuario:
    - Como usuario, quiero proporcionar mi dirección de correo electrónico y crear una contraseña.

- Como usuario, quiero recibir un correo de confirmación para validar mi cuenta.

- Agregar productos al carrito:
  - Como usuario, quiero buscar productos por categoría.
  - Como usuario, quiero poder ver una vista previa del producto al pasar el ratón sobre la imagen.
- Realizar un pago:
  - Como usuario, quiero elegir entre varias opciones de pago (tarjeta de crédito, PayPal).
  - Como usuario, quiero recibir una notificación de confirmación del pago.

Este mapeo visual permite al equipo de desarrollo ver claramente las funcionalidades del sistema y planificar la entrega incremental del producto, asegurando que se cubran las necesidades esenciales de los usuarios desde el principio.

Ventajas de las técnicas ágiles

Las técnicas ágiles como user stories, personas y story mapping, ofrecen varias ventajas clave en el levantamiento de requisitos:

1. **Enfoque centrado en el usuario.** Estas técnicas aseguran que el desarrollo del sistema esté alineado con las necesidades y expectativas de los usuarios finales, lo que mejora la usabilidad y la satisfacción del usuario.
2. **Flexibilidad.** Las técnicas ágiles permiten ajustes rápidos y fáciles a medida que surgen nuevos requisitos o cambios en las expectativas del usuario. Esto es especialmente valioso en proyectos donde los requerimientos no están completamente definidos desde el inicio.
3. **Priorización clara.** Al usar user stories y story mapping, el equipo puede identificar qué funcionalidades son más importantes para los usuarios y priorizarlas en las primeras iteraciones del desarrollo, asegurando que el proyecto avance de manera eficiente.
4. **Colaboración constante.** Las técnicas ágiles fomentan la comunicación continua entre desarrolladores y usuarios, lo que facilita la resolución de problemas y la toma de decisiones informadas.

Desafíos de las técnicas ágiles

A pesar de sus numerosas ventajas, las técnicas ágiles también presentan algunos desafíos, como la dificultad de gestionar los cambios frecuentes en los requisitos y la necesidad de una colaboración constante entre todos los involucrados en el proyecto. Además, la implementación exitosa de estas técnicas requiere de un equipo comprometido y bien capacitado, que entienda el valor de la flexibilidad y la adaptabilidad en el proceso de desarrollo.

Las técnicas ágiles, como user stories, personas y story mapping, son herramientas fundamentales para el levantamiento de requisitos en el desarrollo de sistemas. Al

centrarse en las necesidades de los usuarios y facilitar la colaboración constante, estas técnicas permiten crear soluciones más alineadas con las expectativas de los usuarios, al tiempo que fomentan la flexibilidad y la adaptabilidad en el proceso de desarrollo. La implementación adecuada de estas técnicas contribuye a la entrega de un producto de mayor calidad y más satisfactorio para los usuarios finales.

BIBLIOGRAFÍA

- Busquets, C. (s.f.). MoSCoW: Qué es y cómo utilizarlo para priorizar. Ui–From mars–. https://www.uifrommars.com/priorizacion-metodo-moscow/
- Fresno Chávez, C. (2018). ¿Cómo gestionar la información científico-técnica? Ciudad Educativa. https://elibro.net/es/lc/tecnologicadeloriente/titulos/36730
- Genero Bocco, M., Cruz Lemus, J. A. & Piattini Velthuis, M. G. (2014). Métodos de investigación en ingeniería del software. RA-MA Editorial. https://elibro.net/es/lc/tecnologicadeloriente/titulos/106450
- Gil Pascual, J. A. (2016). Técnicas e instrumentos para la recogida de información. UNED - Universidad Nacional de Educación a Distancia. https://elibro.net/es/lc/tecnologicadeloriente/titulos/48876
- Pérez Agüera, J. R. (2024). Técnicas básicas de priorización. Gemba. https://www.gemba.es/p/tecnicas-basicas-de-priorizacion
- Jurado Antón, O. A. (2021). Métodos Multicriterio para la Priorización de requisitos de software. Universidad Politécnica Salesiana. https://dspace.ups.edu.ec/bitstream/123456789/20957/1/UPS-GT003395.pdf
- Zapata Jaramillo, C. M. (2006). Alineación entre metas organizacionales y elicitación de requisitos del software. Red Dyna. https://elibro.net/es/lc/tecnologicadeloriente/titulos/304