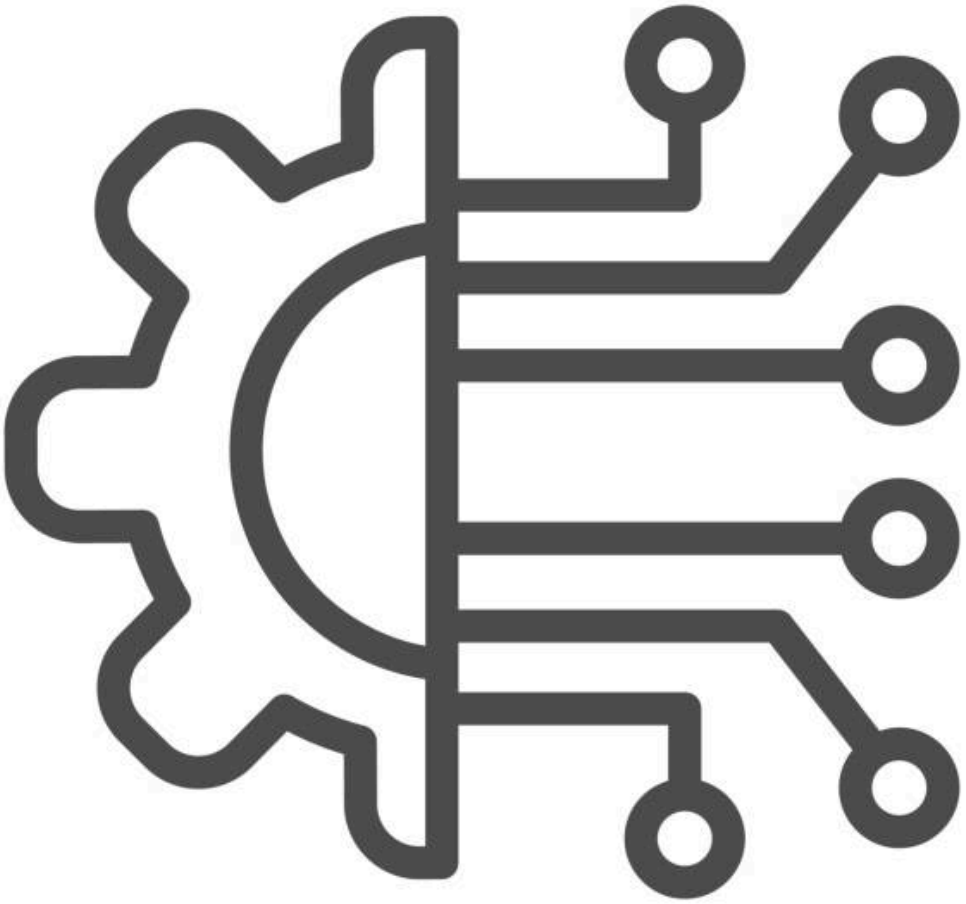
Sin título

TECNOLÓGICO UNIVERSITARIO PLAYACAR

VALORES · CULTURA · VANGUARDIA EDUCATIVA

INGENIERÍA DE REQUISITOS

PROF.-MARCO ANTONIO ANGEL GALEANA



Datos del Estudiante

- Nombre: Jesus Uriel Santana Oliva

- **Curso**: INGENIERÍA DE REQUISITOS
- Grado : 2 -B
- **Profesor**: PROF.-MARCO ANTONIO ANGEL GALEANA
- **Institución**: Tec Playacar
- Ubicación : Playa del Carmen
- Equipo : Jesus Uriel Santana Oliva - Cristian Alejandro

**Actividad

- [x] ~~Revisión de formato APA. ✅ 2025-01-17~~
- [x] ~~Finalizar la bibliografía. ✅ 2025-01-17~~
- [x] ~~Verificar coherencia en la argumentación. ✅ 2025-01-17~~
- [x] ~~Integración de Fuentes por Conceptos ✅ 2025-01-17~~
- [x] ~~Insertar gráficos relevantes. ✅ 2025-01-21~~

✓ Terminada

Tarea Terminada

Jesús Uriel Santana Oliva
215 Ingenieria de Reequisitos
2 - A-B

Introducción a los Requisitos de Software

Los requisitos de software constituyen un componente crítico en el desarrollo de cualquier producto digital. Se definen como las especificaciones detalladas que describen las funciones, características, restricciones y comportamientos esperados de un sistema informático. Estos requisitos son esenciales porque sirven como la base para el diseño, la implementación, las pruebas y el mantenimiento del software, asegurando que este cumpla con las expectativas de los usuarios y los objetivos del proyecto.

La calidad de los requisitos determina el éxito de un proyecto de software. Un mal manejo en esta fase puede resultar en sobrecostos, retrasos e insatisfacción del cliente. Por lo tanto, es imprescindible adoptar técnicas y herramientas adecuadas para su recopilación, análisis, especificación, validación y gestión. Según Glinz (2019), la gestión eficiente de requisitos no solo impacta en el desarrollo inicial del

sistema, sino que también afecta directamente su mantenimiento y evolución en el tiempo (Glinz, 2019).

Marco Conceptual de los Requisitos en Software

El marco conceptual de los requisitos de software abarca diversas definiciones y clasificaciones que permiten comprender su naturaleza y relevancia en el desarrollo de sistemas. A continuación, se describen los conceptos clave:

1. Tipos de requisitos

Los requisitos de software se dividen principalmente en tres categorías:

- **Requisitos funcionales:** Describen lo que el sistema debe hacer. Especifican las funciones o servicios que el software debe proporcionar al usuario. Por ejemplo, "el sistema debe permitir a los usuarios registrar nuevas cuentas" es un requisito funcional.
- **Requisitos no funcionales:** Detallan las propiedades del sistema, como el rendimiento, la seguridad, la usabilidad o la escalabilidad. Un ejemplo sería "el sistema debe procesar 10.000 solicitudes por minuto sin degradación en el rendimiento".
- **Requisitos del dominio:** Se refieren a las restricciones o especificaciones impuestas por el contexto específico del negocio o industria. Estos incluyen regulaciones legales o estándares específicos del sector (Sommerville, 2019).

2. Proceso de ingeniería de requisitos

La ingeniería de requisitos es un proceso estructurado que consta de cinco etapas principales:

1. **Elicitación de requisitos:** Implica la identificación de las necesidades y expectativas de los interesados (stakeholders).
2. **Análisis de requisitos:** Consiste en refinar, clasificar y priorizar los requisitos recopilados.
3. **Especificación de requisitos:** Se trata de documentar los requisitos de manera clara y detallada.
4. **Validación de requisitos:** Verificar que los requisitos especificados reflejan correctamente las necesidades del cliente.
5. **Gestión de requisitos:** Supervisar y controlar los cambios en los requisitos durante el ciclo de vida del proyecto (IEEE, 2018).

3. Herramientas para la gestión de requisitos

Existen diversas herramientas que facilitan el manejo de requisitos en proyectos complejos, como IBM Rational DOORS, Jira y Trello. Estas

herramientas ayudan en la trazabilidad, priorización y documentación efectiva de los requisitos (Bourque & Fairley, 2019).

Selección Metodológica

La selección de la metodología para la gestión de requisitos de software depende de varios factores, como el tamaño del proyecto, la complejidad del sistema, los recursos disponibles y el nivel de incertidumbre. A continuación, se describen dos enfoques metodológicos ampliamente utilizados:

1. Metodologías tradicionales

Las metodologías tradicionales, como el modelo en cascada, siguen un enfoque lineal y secuencial, donde cada fase del proyecto debe completarse antes de pasar a la siguiente. Estas metodologías son adecuadas para proyectos bien definidos y con requisitos estables. En este enfoque, la especificación detallada de los requisitos se realiza al inicio del proyecto y sirve como base para el desarrollo posterior.

Según Sommerville (2019), este modelo es eficaz cuando los requisitos son claros y no están sujetos a cambios frecuentes (Sommerville, 2019).

2. Metodologías ágiles

En contraste, las metodologías ágiles, como Scrum y Kanban, adoptan un enfoque iterativo e incremental. En lugar de especificar todos los requisitos al inicio, estos se desarrollan y refinan a lo largo del proyecto mediante ciclos cortos de retroalimentación (sprints). Este enfoque es ideal para entornos dinámicos donde los requisitos pueden cambiar con frecuencia.

Beck et al. (2020) señalan que las metodologías ágiles promueven la colaboración entre los interesados y el equipo de desarrollo, mejorando así la calidad del producto final (Beck et al., 2020).

Conclusión

El éxito de cualquier proyecto de software depende en gran medida de una gestión adecuada de los requisitos. Es fundamental comprender el marco conceptual de los mismos, así como seleccionar la metodología más adecuada en función de las características del proyecto. La aplicación de técnicas y herramientas específicas no solo mejora la calidad del producto final, sino que también contribuye a la eficiencia y satisfacción de los interesados.

Referencias

- Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., & Grenning, J. (2020). *The Agile Manifesto*. Recuperado de https://www.agilealliance.org/agile101/the-agile-manifesto/.
- Bourque, P., & Fairley, R. E. (2019). *Guide to the Software Engineering Body of Knowledge (SWEBOK)*. ACM Press. Recuperado de https://dl.acm.org/doi/10.5555/3346862.
- Glinz, M. (2019). *A Glossary of Requirements Engineering Terminology*. arXiv. Recuperado de https://arxiv.org/pdf/1902.03461.pdf.
- IEEE (2018). *IEEE Standard for Systems and Software Engineering--Life Cycle Processes--Requirements Engineering*. IEEE Xplore. Recuperado de https://ieeexplore.ieee.org/document/8290593.
- Sommerville, I. (2019). *Software Engineering*. Pearson Education. Recuperado de https://www.pearson.com/store/p/software-engineering/P100000737911.