Datos Generales de la asignatura

Nombre de la asignatura:	**Ingeniería de Requerimientos**
Clave de la asignatura:	**ISC-2603**
SATCA[1]:	**2-2 4**
Carrera:	**Ingeniería en Sistemas Computacionales**

2. Presentación

Caracterización de la asignatura

La asignatura **Ingeniería de Requerimientos** constituye un componente fundamental dentro del área de ingeniería de software, cuya finalidad es dotar al estudiante de competencias avanzadas para la especificación, análisis, validación y gestión sistemática de los requerimientos de software y sistemas, como parte integral del ciclo de vida del desarrollo.

Este curso proporciona los principios teóricos y metodológicos que sustentan la definición de los requerimientos funcionales y no funcionales, orientados al aseguramiento de calidad, cumplimiento normativo, trazabilidad y alineación con los objetivos de negocio y necesidades de los stakeholders.

La asignatura enfatiza el uso de modelos formales y semi-formales, técnicas de elicitation estructuradas, mecanismos de gestión de cambios y herramientas especializadas para la documentación, análisis y verificación de requisitos. Asimismo, se estudian estándares internacionales como el **IEEE 830**, **ISO/IEC/IEEE 29148:2018**, el marco **BABOK v3** (Estandar global para la práctica de análisis de negocios) y principios del **CMMI-DEV** para prácticas de madurez en ingeniería de requisitos.

Su enfoque es metodológicamente agnóstico, permitiendo la aplicación en contextos predictivos (waterfall) y adaptativos (ágiles), haciendo uso de técnicas como casos de uso, historias de usuario, prototipos, entrevistas, análisis de tareas y modelado UML/BPMN, fomentando un perfil profesional capaz de afrontar problemáticas reales en el desarrollo de software con un alto grado de formalidad y precisión técnica.

[1] Sistema de Asignación y Transferencia de Créditos Académicos

Intención didáctica
La intención didáctica de la asignatura **Ingeniería de Requerimientos** es el desarrollo progresivo de competencias cognitivas, procedimentales y actitudinales que permitan al estudiante: 1. **Comprende y aplica marcos teóricos y normativos** vinculados con la especificación y gestión de requisitos de software, en concordancia con buenas prácticas internacionales y estándares de la industria. 2. **Diseña y ejecuta procesos de elicitation de requerimientos**, empleando metodologías cualitativas y cuantitativas para la captura estructurada de necesidades desde múltiples perspectivas (técnica, operativa, organizacional y de usuario final). 3. **Modela requerimientos utilizando notaciones estandarizadas** como UML, BPMN y plantillas formales (IEEE), garantizando claridad, no ambigüedad y validación efectiva por parte de los stakeholders. 4. **Aplica técnicas avanzadas de análisis, priorización, verificación y validación**, con enfoque en mitigación de riesgos, gestión de inconsistencias y soporte a decisiones arquitectónicas y de negocio. 5. **Desarrolla habilidades para la gestión del cambio y trazabilidad**, asegurando el alineamiento continuo de los requisitos con los objetivos del proyecto durante todo el ciclo de vida del software. 6. **Integra herramientas CASE y plataformas colaborativas**, como JIRA, ReqView, Enterprise Architect, Jama o DOORS, en escenarios simulados o reales, orientados al control documental, seguimiento de versiones y automatización de tareas relacionadas. La estrategia pedagógica se sustenta en metodologías activas de aprendizaje, como aprendizaje basado en proyectos (ABP), simulaciones, estudio de casos reales, peer review técnico y evaluación continua mediante entregables progresivos, fomentando la autonomía, el pensamiento crítico, la rigurosidad técnica y la responsabilidad profesional.

3. Participantes en el diseño y seguimiento curricular del programa

Lugar y fecha de elaboración o revisión	Participantes	Observaciones
Dirección General de Institutos Descentralizados, México D.F. Fecha: 12 al 14 de septiembre de 2012.	Representantes: Tecnológico de Estudios Superiores de Coacalco. Instituto Tecnológico Superior de Atlixco, Comalcalco, Fresnillo, Santiago Papasquiaro, Tepexi de Rodríguez, Zapopan.	Análisis y adecuación por competencias del módulo de la especialidad “Ingeniería de Software" de la carrera de Ingeniería en Sistemas Computacionales.
Tecnológico de Estudios Superiores de Coacalco Fecha: 10 al 14 de agosto de 2015.	Representantes: Tecnológico de Estudios Superiores de Coacalco.	Actualización del módulo de la especialidad “Ingeniería de Software" de la carrera de Ingeniería en Sistemas Computacionales
Tecnológico de Estudios Superiores de Coacalco Fecha: 21 al 22 de junio de 2019.	Representantes: Tecnológico de Estudios Superiores de Coacalco.	Actualización de asignaturas del módulo de la especialidad "Ingeniería de Software" de la carrera de Ingeniería en Sistemas Computacionales.
Tecnológico de Estudios Superiores de Coacalco Octubre del 2025.	Representantes: Docentes de la Academia de Ingeniería en Sistemas Computacionales del Tecnológico de Estudios Superiores de Coacalco.	Análisis, formulación de competencias, actualización de temario, formulación de actividades y prácticas de la especialidad de "Ingeniería de Software y Arquitectura de Sistemas” de la carrera de Ingeniería en Sistemas Computacionales.

4. Competencia(s) a desarrollar

Competencia(s) específica(s) de la asignatura
Analizar, modelar y gestionar requerimientos funcionales y no funcionales de sistemas de software, mediante la aplicación de metodologías estructuradas, técnicas de elicitation, herramientas UMLy estándares internacionales, con el fin de asegurar la

trazabilidad, validación y alineación de los requerimientos con los objetivos del proyecto y las necesidades de los stakeholders.

5. Competencias previas

a) **Comprende los fundamentos de la ingeniería de software y el ciclo de vida del desarrollo de sistemas.**

Permite al estudiante situar la ingeniería de requerimientos dentro del proceso general de desarrollo y entender su impacto en las etapas de análisis, diseño y mantenimiento.

b) **Domina principios básicos de programación y estructuras de datos.**

Facilita la interpretación técnica de los requerimientos y la traducción de necesidades del usuario en soluciones viables desde el punto de vista del desarrollo.

c) **Conoce metodologías de análisis y diseño de sistemas.**

Ayuda a comprender la modelación con UML, BPMN y otras herramientas visuales que se emplean para representar requerimientos y procesos.

d) **Posee habilidades de comunicación efectiva y trabajo colaborativo.**

Son esenciales para realizar entrevistas, talleres y revisiones con los *stakeholders*, además de participar en equipos multidisciplinarios durante la elicitation y validación de requerimientos.

e) **Aplica pensamiento crítico y capacidad de análisis para la resolución de problemas.**

Permite evaluar la consistencia, viabilidad y prioridad de los requerimientos, así como proponer soluciones alineadas con los objetivos del negocio.

6. Temario

Unidad	Temas	Subtemas
1	**Técnicas de elicitation**	1.1 **Fundamentos de la elicitation de requerimientos** 1.1.1 Tipos de requerimientos: funcionales vs. no funcionales. 1.1.2 Principios y desafíos de la elicitation. 1.2 **Técnicas de elicitation** 1.2.1 Entrevistas estructuradas y semi-estructuradas. 1.2.2 Cuestionarios y encuestas.

		1.2.3 Workshops y sesiones JAD (Joint Application Design). 1.2.4 Observación directa y análisis de tareas. 1.2.5 Análisis de documentación existente. 1.3 **Priorización inicial de requerimientos** 1.3.1 Técnicas como MoSCoW, Kano, 100-points. (herramientas de priorización utilizadas en la gestión de proyectos y productos, especialmente en entornos ágiles)
2	**Modelos de requerimientos**	2.1 **Introducción al modelado de requerimientos** 2.1.1 Importancia del modelado para análisis y verificación. 2.1.2 Tipos de modelos: estructurales, de comportamiento, de datos. 2.2 **Modelado con UML** 2.2.1 Diagramas de casos de uso. 2.2.2 Diagramas de secuencia. 2.2.3 Diagramas de actividades. 2.3 **Modelado con BPMN** 2.3.1 Procesos de negocio vs. procesos de sistema. 2.3.2 Roles y eventos.
3	**Especificaciones de los requerimientos**	3.1 **Estándares para documentación de requerimientos** 3.1.1 IEEE 830, ISO/IEC/IEEE 29148:2018. 3.1.2 Plantillas y estructuras típicas (SRS – Software Requirements Specification). 3.2 **Escritura efectiva de requerimientos (RF / RNF)** 3.2.1 Criterios de calidad: claros, medibles, verificables. 3.2.2 Lenguaje natural controlado vs. lenguaje formal. 3.3 **Historias de usuario y criterios de aceptación** 3.3.1 Formato: “Como [rol], quiero [funcionalidad] para [beneficio]”. 3.3.2 INVEST y BDD (Given–When–Then). 3.4 **Verificación de requisitos** 3.4.1 Revisión cruzada, inspecciones, checklist de validación.

4	**Trazabilidad y control de los requerimientos**	4.1 **Gestión de requerimientos** 4.1.1 Ciclo de vida del requerimiento. 4.1.2 Control de versiones. 4.2 **Trazabilidad** 4.2.1 Matrices de trazabilidad (RTM). 4.2.2 Relación entre requerimientos, diseño, código y pruebas. 4.3. **Gestión de cambios** 4.3.1 Flujo de aprobación de cambios. 4.3.2 Herramientas de soporte (JIRA, ReqView, DOORS). 4.4 **Impacto del cambio en la arquitectura y el cronograma** 4.4.1 Análisis de impacto.
5	**Validación y Verificación de requerimientos**	5.1 **Validación vs. verificación** 5.1.1 Diferencias clave. 5.1.2 Técnicas: revisión, simulación, prototipado. 5.2 **Criterios de aceptación** 5.2.1 Técnicas BDD (Behavior Driven Development). 5.2.2 Definición de “Done”. 5.3 **Técnicas participativas** 5.3.1 Reuniones de validación. 5.3.2 Roles de los stakeholders. **5.4 Pruebas de aceptación y casos de prueba derivados de requerimientos** 5.4.1 Trazabilidad entre pruebas y requisitos.

7. Actividades de aprendizaje de los temas

Unidad 1. Técnicas de elicitation	
Competencias	Actividades de aprendizaje
Específica(s): Aplica técnicas de *elicitation* de requerimientos para identificar, analizar y documentar necesidades funcionales y no funcionales de los usuarios, empleando métodos estructurados como entrevistas, cuestionarios, observación y análisis documental, con el fin de garantizar la precisión, claridad y completitud de los requerimientos del sistema de software Genéricas: • Capacidad de organizar y planificar • Comunicación oral y escrita • Capacidad de análisis y síntesis • Habilidad para buscar y analiza	1. **Análisis teórico y discusión guiada sobre los fundamentos de la elicitation.** • Revisar los conceptos de requerimientos funcionales y no funcionales. • Identificar los principales desafíos de la elicitation mediante el análisis de casos reales. • Realizar una lluvia de ideas sobre errores comunes en la captura de requerimientos. 2. **Diseño y aplicación de una entrevista estructurada a un "cliente simulado".** • Elaborar un guion de preguntas abiertas y cerradas. • Aplicar la entrevista a un compañero o actor que represente al usuario final. • Registrar y clasificar los requerimientos obtenidos según su tipo. 3. **Desarrollo de un ejercicio práctico de priorización inicial de requerimientos.** • Utilizar las técnicas **MoSCoW**, **Kano** o **100-points** para jerarquizar requerimientos obtenidos de un caso de estudio. • Justificar las decisiones de priorización con base en criterios de valor y factibilidad. 4. **Elaboración de un informe de resultados de elicitation.**

	• Documentar los requerimientos funcionales y no funcionales identificados. • Incluir evidencias del proceso: entrevistas, encuestas o análisis de documentos. • Presentar el informe en clase, destacando los retos enfrentados y las lecciones aprendidas.
Unidad 2. Modelos de requerimientos	
Competencias	Actividades de aprendizaje
Específica(s): Modela requerimientos de sistemas de software mediante notaciones formales y semi-formales, como UML y BPMN, para representar de forma estructurada los procesos, interacciones y comportamientos del sistema, favoreciendo la detección de inconsistencias y la comunicación efectiva entre analistas, desarrolladores y stakeholders. Genéricas: • Solución de problemas • Capacidad crítica y autocrítica • Trabajo en equipo	1. **Análisis y debate sobre la importancia del modelado de requerimientos.** • Identificar los tipos de modelos (estructurales, de comportamiento, de datos) y su función en el ciclo de vida del software. • Analizar ejemplos reales donde el modelado ayudó a detectar ambigüedades o errores en requerimientos. • Elaborar un resumen comparativo de modelos formales y semi-formales. 2. **Diseño de diagramas UML para representar requerimientos funcionales.** • Elaborar **diagramas de casos de uso** que describan actores y funcionalidades del sistema. • Complementar con **diagramas de secuencia** que muestren la interacción entre objetos. • Representar **diagramas de actividades** para visualizar flujos de procesos y decisiones.

	3. **Modelado de procesos de negocio utilizando BPMN.** • Crear diagramas BPMN para representar procesos del usuario y procesos del sistema. • Identificar roles, eventos, compuertas y tareas. • Comparar los diagramas BPMN con los modelos UML elaborados, destacando su propósito y alcance. 4. **Presentación y validación de modelos ante el grupo o stakeholders simulados.** • Exponer los diagramas elaborados y explicar su interpretación. • Recibir retroalimentación sobre claridad, coherencia y completitud. • Ajustar los modelos según observaciones para garantizar su validez y trazabilidad.
Unidad 3. Especificaciones de los requerimientos	
Competencias	Actividades de aprendizaje
Específica(s): Redacta especificaciones de requerimientos funcionales y no funcionales conforme a estándares internacionales como IEEE 830 e ISO/IEC/IEEE 29148:2018, aplicando principios de claridad, verificabilidad y	1. **Investigación y análisis comparativo de estándares internacionales de documentación.**

trazabilidad, para garantizar la calidad, comprensión y mantenibilidad de la documentación técnica del software. Genéricas: • Capacidad de aplicar los conocimientos en la práctica • Capacidad de generar nuevas ideas(creatividad) • Habilidad para trabajar en forma Autónoma • Capacidad para diseñar y gestionar proyectos.	• Revisar los estándares **IEEE 830** y **ISO/IEC/IEEE 29148:2018**. • Elaborar una tabla comparativa de sus secciones, estructura y propósitos. • Identificar buenas prácticas de documentación en organizaciones reales de desarrollo de software. 2. **Redacción práctica de requerimientos funcionales y no funcionales.** • Aplicar criterios de calidad: claridad, consistencia, verificabilidad y medibilidad. • Usar lenguaje natural controlado para evitar ambigüedades. • Redactar ejemplos de requerimientos mal escritos y mejorarlos según las normas estudiadas. 3. **Elaboración de historias de usuario con criterios de aceptación.** • Utilizar el formato: *"Como [rol], quiero [funcionalidad] para [beneficio]"*. • Incorporar los principios **INVEST** y **BDD (Given–When–Then)** para estructurar requisitos ágiles. • Documentar al menos tres historias de usuario con sus criterios de validación. 4. **Verificación y revisión cruzada de especificaciones.** • Formar equipos para intercambiar y revisar documentos SRS. • Utilizar listas de chequeo (checklists) para evaluar

	calidad, completitud y trazabilidad. • Corregir y actualizar los documentos con base en la retroalimentación del grupo o del docente.
Unidad 4. Trazabilidad y control de los requerimientos	
Competencias	Actividades de aprendizaje
Específica(s): Gestiona la trazabilidad y el control de cambios de los requerimientos durante todo el ciclo de vida del software, aplicando metodologías y herramientas especializadas para mantener la coherencia entre requerimientos, diseño, código y pruebas, garantizando la calidad, estabilidad y alineación del producto con los objetivos del proyecto. Genéricas: • Capacidad de aplicar losconocimientos en la práctica • Capacidad de generar nuevas ideas(creatividad) • Habilidad para trabajar en forma Autónoma • Capacidad para diseñar y gestionarproyectos • Iniciativa y espíritu emprendedor	1. **Análisis del ciclo de vida de los requerimientos y control de versiones.** • Elaborar un esquema que muestre las etapas del ciclo de vida del requerimiento (creación, revisión, aprobación, implementación, validación y mantenimiento). • Analizar ejemplos de cómo los cambios en los requerimientos impactan el desarrollo y la documentación. • Simular la creación de un control de versiones básico en un documento compartido. 2. **Diseño de una Matriz de Trazabilidad de Requerimientos (RTM).** • Identificar la relación entre requerimientos, componentes de diseño, código fuente y casos de prueba. • Construir una RTM para un sistema de ejemplo, usando hojas de cálculo o herramientas CASE (como ReqView o Jira). • Verificar la completitud de los enlaces entre los

	diferentes elementos del proyecto. 3. **Simulación de gestión de cambios en requerimientos.** • Aplicar un flujo de aprobación de cambios mediante roles (analista, desarrollador, cliente). • Registrar solicitudes de cambio (Change Requests) con su justificación y evaluación de impacto. • Documentar las decisiones y actualizaciones en una bitácora de control. 4. **Análisis del impacto de los cambios en la arquitectura y cronograma del proyecto.** • Evaluar cómo una modificación de requerimiento afecta el diseño, las pruebas y los plazos del proyecto. • Elaborar un informe de impacto con recomendaciones técnicas y de gestión. • Presentar los hallazgos ante el grupo simulando una reunión de control de cambios.
Unidad 5. Validación y Verificación de requerimientos	
Competencias	Actividades de aprendizaje
Específica(s): Valida y verifica los requerimientos de software aplicando técnicas de revisión, prototipado y pruebas de aceptación, en colaboración con los stakeholders, para garantizar que el sistema desarrollado cumpla con los criterios de aceptación, las	1. Análisis comparativo entre validación y verificación de requerimientos. • Elaborar un cuadro que distinga objetivos, responsables, momentos y resultados de cada proceso.

necesidades del usuario y los objetivos del proyecto. Genéricas: • Capacidad de aplicar losconocimientos en la práctica • Capacidad de generar nuevas ideas(creatividad) • Habilidad para trabajar en forma Autónoma • Capacidad para diseñar y gestionarproyectos • Iniciativa y espíritu emprendedor	• Analizar ejemplos de proyectos en los que se confundieron ambas prácticas y sus consecuencias. • Reflexionar en grupo sobre la importancia de ambas etapas dentro del ciclo de vida del software. 2. Aplicación de técnicas de revisión, simulación y prototipado. • Desarrollar un prototipo de baja fidelidad (mockup o wireframe) que represente un conjunto de requerimientos. • Realizar una sesión de revisión con compañeros simulando roles de stakeholders. • Registrar observaciones y sugerencias derivadas del proceso de validación. 3. Diseño de criterios de aceptación basados en BDD (Behavior Driven Development). • Redactar criterios utilizando el formato Given–When–Then. • Definir la “Definición de Done” para un conjunto de historias de usuario. • Discutir cómo estos criterios permiten medir objetivamente el cumplimiento de los requerimientos. 4. Elaboración y ejecución de pruebas de aceptación. • Derivar casos de prueba directamente de los requerimientos y criterios de aceptación. • Documentar la trazabilidad entre pruebas, requerimientos y resultados. • Simular una reunión de validación final con stakeholders para

	presentar los resultados y obtener aprobación.

8. Práctica(s)

1. **Caso 1: Sistema de gestión escolar (Elicitación de requerimientos)**

    - Realizar entrevistas y encuestas a “stakeholders” simulados (docentes, alumnos, administrativos).
    - Identificar requerimientos funcionales y no funcionales.
    - Elaborar un documento inicial de necesidades del cliente.

2. **Caso 2: Plataforma de comercio electrónico (Priorización de requerimientos)**

    - Aplicar técnicas **MoSCoW** y **Kano** para priorizar funcionalidades.
    - Justificar la selección en base al valor de negocio y viabilidad técnica.

3. **Caso 3: Sistema de reservas de hospital (Modelado UML)**

    - Crear diagramas de **casos de uso**, **secuencia** y **actividades** para representar interacciones del sistema.
    - Detectar inconsistencias o redundancias entre los modelos.

4. **Caso 4: Proceso de admisión universitaria (Modelado BPMN)**

    - Representar el flujo de negocio con **BPMN**, incluyendo roles, tareas y eventos.
    - Analizar puntos críticos y oportunidades de mejora del proceso.

5. **Caso 5: Sistema de inventarios (Especificación de requerimientos con estándar IEEE 830)**

    - Redactar un **SRS (Software Requirements Specification)** completo.
    - Aplicar criterios de claridad, verificabilidad y trazabilidad.

6. **Caso 6: Aplicación móvil de servicios urbanos (Historias de usuario y criterios de aceptación)**

    - Formular historias de usuario usando el formato “Como [rol], quiero [funcionalidad] para [beneficio]”.
    - Definir criterios de aceptación con metodología **BDD (Given–When–Then)**.

7. **Caso 7: Sistema bancario en línea (Matriz de trazabilidad)**

- Elaborar una **RTM (Requirements Traceability Matrix)** que vincule requerimientos con diseño, código y pruebas.
- Evaluar la completitud y consistencia de las relaciones.

8. **Caso 8: Portal de servicios gubernamentales (Gestión de cambios)**

    - Simular solicitudes de cambio de requerimientos y aplicar un **flujo de aprobación**.
    - Documentar los impactos en la arquitectura, cronograma y presupuesto.

9. **Caso 9: Aplicación de gestión de proyectos (Validación y verificación)**

    - Prototipar interfaces y realizar revisiones con stakeholders.
    - Verificar el cumplimiento de criterios de aceptación y documentar resultados.

10. **Caso 10: Sistema integral de transporte urbano (Proyecto integrador final)**

- Integrar todos los aprendizajes: elicitation, modelado, documentación, trazabilidad y validación.
- Presentar el caso completo con su documentación técnica y justificación metodológica.

9. Proyecto de asignatura

El objetivo del proyecto que planteé el docente que imparta esta asignatura, es demostrar el desarrollo y alcance de la(s) competencia(s) de la asignatura, considerando las siguientes fases:

- **Fundamentación:** marco referencial (teórico, conceptual, contextual, legal) en el cual se fundamenta el proyecto de acuerdo con un diagnóstico realizado, mismo que permite a los estudiantes lograr la comprensión de la realidad o situación objeto de estudio para definir un proceso de intervención o hacer el diseño de un modelo.

- **Planeación:** con base en el diagnóstico en esta fase se realiza el diseño del proyecto por parte de los estudiantes con asesoría del docente; implica planificar un proceso: de intervención empresarial, social o comunitario, el diseño de un modelo, entre otros, según el tipo de proyecto, las actividades a realizar los recursos requeridos y el cronograma de trabajo.

- **Ejecución:** consiste en el desarrollo de la planeación del proyecto realizada por parte de los estudiantes con asesoría del docente, es decir en la intervención (social, empresarial), o construcción del modelo propuesto según el tipo de proyecto, es la fase de mayor duración que implica el desempeño de las competencias genéricas y especificas a desarrollar.

- **Evaluación:** es la fase final que aplica un juicio de valor en el contexto laboral-profesión, social e investigativo, ésta se debe realizar a través del reconocimiento de logros y aspectos a mejorar se estará promoviendo el concepto de "evaluación para la mejora continua", la metacognición, el desarrollo del pensamiento crítico y reflexivo en los estudiantes.

10. Evaluación por competencias

Son las técnicas, instrumentos y herramientas sugeridas para constatar los desempeños académicos de las actividades de aprendizaje.

11. Fuentes de información

1. **Sommerville, Ian.**
   Ingeniería de Software. 10ª edición. Pearson Educación, 2020.

   Obra fundamental que aborda el ciclo de vida del software, requerimientos, validación, mantenimiento y gestión del cambio.
2. **Pressman, Roger S. & Maxim, Bruce R.**
   Ingeniería del Software: Un Enfoque Práctico. 9ª edición. McGraw-Hill, 2021.

   Referencia clave para metodologías de desarrollo, análisis de requerimientos y documentación profesional.

3. **Booch, Grady; Rumbaugh, James; Jacobson, Ivar.**
   El Lenguaje Unificado de Modelado (UML): Guía del Usuario. 2ª edición. Addison-Wesley, 2005.

   Texto esencial para comprender el origen y la aplicación de UML en modelado estructural y de comportamiento.

4. **Dennis, Alan; Wixom, Haley; Tegarden, David.**
   Sistemas de Información: Análisis y Diseño Estructurado y Orientado a Objetos. 7ª edición. Wiley, 2020.

   Incluye casos de estudio reales aplicando UML, modelos de requerimientos y herramientas CASE

5. **Object Management Group (OMG).**
   UML 2.5.1 Specification. OMG, 2017. Disponible en:

https://www.omg.org/spec/UML

Documento oficial que define formalmente la notación UML, sus diagramas, relaciones y semántica.

6. **Wiegers, Karl & Beatty, Joy.**
   Software Requirements. 3ª edición. Microsoft Press, 2013.

   Guía completa sobre elicitation, análisis, validación y gestión de requerimientos con ejemplos industriales.

7. **Pohl, Klaus.**
   Requirements Engineering: Fundamentals, Principles, and Techniques. Springer, 2010.

   Explica con rigor académico el proceso completo de ingeniería de requerimientos y su aplicación en entornos ágiles.

8. **OMG.**
   Business Process Model and Notation (BPMN) 2.0.2 Specification. Object Management Group, 2014. Disponible en: https://www.omg.org/spec/BPMN/2.0.2

   Estándar oficial para modelar procesos de negocio y su relación con los sistemas de información.

9. **IEEE Computer Society.**
   ISO/IEC/IEEE 29148:2018 — Systems and Software Engineering — Life Cycle Processes — Requirements Engineering. IEEE, 2018.

   Norma internacional que define buenas prácticas para documentación y gestión de requerimientos.

10. **Arlow, Jim & Neustadt, Ila.**
    UML 2 and the Unified Process: Practical Object-Oriented Analysis and Design. 2ª edición. Addison-Wesley, 2005.

    Expone la integración del modelado UML con procesos unificados (RUP), ideal para proyectos académicos o industriales.