

## 1. APLICACIÓN DEL ANÁLISIS FUNCIONAL DE LA FIGURA PROFESIONAL “DESARROLLO DE SOFTWARE”

El análisis funcional (AF) constituye una técnica metodológica orientada para identificar, organizar y estructurar las actividades de un proceso productivo o de prestación de servicios. Representa el punto de partida para elaborar el perfil profesional en términos de competencias y la base para el diseño curricular. En este marco, se presentan los elementos que integran el análisis funcional aplicados a la figura profesional de “Desarrollo de Software”.

### a) Identificación del Objetivo:

La figura profesional se orienta a formar bachilleres capaces de desarrollar soluciones informáticas mediante la aplicación de metodologías ágiles, principios de programación, diseño, tecnologías emergentes, dando respuesta a necesidades de automatización, innovación, transformación digital en diversos sectores productivos, sociales industriales y educativos.

### b) Deducción de las actividades profesionales

El o la Bachiller Técnico en Desarrollo de Software posee las competencias necesarias para participar en el ciclo de construcción de aplicaciones informáticas orientadas a escritorio, web y móviles, desde el análisis, diseño, desarrollo, pruebas, mantenimiento y documentación, utilizando lenguajes de programación, bases de datos y herramientas tecnológicas actuales, en la creación de software funcional, trabajo en equipo aplicando metodologías de desarrollo de software y contribuyendo con soluciones a problemáticas reales del entorno.

Entre las actividades profesionales se detallan las siguientes:

- Diseño, desarrollo y mantenimiento de aplicaciones informáticas básicas y funcionales, aplicando lógica computacional y programación estructurada.
- Construcción de soluciones digitales utilizando programación orientada a objetos e integración de bases de datos, según especificaciones funcionales técnicas.
- Diseño e implementación de aplicaciones web y móviles de nivel intermedio, considerando la accesibilidad, experiencia de usuario y compatibilidad con diversos dispositivos.
- Realizar pruebas de funcionamiento, aplicando medidas de seguridad y elaboración de documentos técnicos de las soluciones desarrolladas.
- Colaboración en proyectos de desarrollo de software aplicando metodologías ágiles con herramientas digitales y buenas prácticas que fomenten el trabajo organizado y en equipo.
- Proponer soluciones tecnológicas con enfoque innovador y emprendedor, solucionando problemáticas del entorno.

Además, de sus competencias técnicas, estos bachilleres desarrollan habilidades blandas como trabajo en equipo, comunicación efectiva, liderazgo, creatividad y adaptabilidad, que les permiten resolver problemas con responsabilidad y aportar soluciones innovadoras en un mercado tecnológico dinámico y competitivo.

### c) Desagregación de las actividades

La desagregación de las actividades profesionales permite identificar las acciones clave que debe dominar el bachiller técnico en desarrollo de software. Estas tareas reflejan los conocimientos y habilidades necesarios para participar activamente en el ciclo de vida de una solución digital, desde la concepción de la idea hasta su implementación, testeo y documentación, dentro de contextos de aprendizaje y aplicación.

- Diseño, desarrollo y mantenimiento de aplicaciones informáticas básicas y funcionales, aplicando lógica computacional y programación estructurada.
  - Analizar un problema o necesidad y plantear una solución lógica.
  - Escribir algoritmos y diagramas de flujo.
  - Codificar en un lenguaje estructurado.
  - Probar y depurar el código asegurando su funcionalidad.
  - Documentar el código fuente y su propósito.
- Construcción de soluciones digitales utilizando programación orientada a objetos e integración de bases de datos, según especificaciones funcionales técnicas.
  - Analizar especificaciones funcionales.
  - Crear y administrar bases de datos.
  - Diseñar clases, objetos y estructuras de datos.
  - Codificar en lenguajes de programación orientados a objetos.
  - Establecer la conexión entre la aplicación y la base de datos.
  - Realizar pruebas de persistencia y recuperación de datos.
- Diseño e implementación de aplicaciones web y móviles de nivel intermedio, considerando criterios de accesibilidad, experiencia de usuario y compatibilidad con diversos dispositivos.
  - Utilizar frameworks.
  - Crear interfaces gráficas responsivas.
  - Aplicar principios de diseño centrado en el usuario (UX/UI).
  - Adaptar funcionalidades para dispositivos móviles.
  - Realizar pruebas de compatibilidad en distintos navegadores o plataformas.
- Realizar pruebas de funcionamiento, aplicando medidas de seguridad y elaboración de documentos técnicos de las soluciones desarrolladas.
  - Aplicar pruebas unitarias y funcionales.
  - Verificar que la aplicación cumpla con los requerimientos.
  - Detectar errores y realizar correcciones.
  - Implementar controles de seguridad (validaciones, acceso, cifrado de datos).
  - Elaborar documentación técnica y manuales de usuario.
- Colaboración en proyectos de desarrollo de software aplicando metodologías ágiles con herramientas digitales y buenas prácticas que fomenten el trabajo organizado y en equipo.
  - Participar y colaborar en la planificación y seguimiento de tareas mediante la implementación de metodologías de desarrollo de software.
  - Aplicar control de versiones.
  - Utilizar herramientas de gestión de proyectos.

- Comunicar ideas de forma clara y trabajar en equipo.
- Proponer soluciones tecnológicas con enfoque innovador y emprendedor, solucionando problemáticas del entorno.
  - Formular estrategias tecnológicas en la resolución de problemas.
  - Diseñar prototipos funcionales de soluciones digitales.
  - Evaluar su viabilidad técnica y operativa.
  - Aplicar la solución tecnológica en contextos reales o simulados.

Además de sus competencias técnicas en programación, diseño, análisis y desarrollo de aplicaciones, estos bachilleres fortalecen habilidades blandas como el liderazgo, la comunicación efectiva, el trabajo en equipo y la resolución de problemas. Se distinguen por su responsabilidad y compromiso con la calidad en la creación de soluciones tecnológicas, así como por su capacidad de adaptación frente a los constantes avances y exigencias del entorno digital y del mercado laboral en el área del desarrollo de software.

## 2. PERFIL PROFESIONAL DE LA FIGURA “DESARROLLO DE SOFTWARE”

### 1. Caracterización

El Desarrollo de Software constituye un área estratégica en el ámbito tecnológico, impulsa la creación de soluciones digitales para responder a las necesidades de los sectores educativos, empresariales, comunitarios, industriales y de servicios. Esta figura profesional busca formar Bachilleres Técnicos con competencias en el análisis, diseño, programación, prueba, implementación y mantenimiento de aplicaciones y sistemas informáticos, aplicando metodologías de desarrollo y estándares de calidad. Se enfatiza el uso de lenguajes de programación, entornos de desarrollo, bases de datos y herramientas de gestión de proyectos, así como la incorporación de tecnologías emergentes que favorecen la innovación y la transformación digital. Estos bachilleres desempeñan un rol esencial en la modernización tecnológica, aportando al desarrollo de soluciones eficientes, seguras y escalables, que fortalecen la competitividad y el crecimiento de los diferentes sectores económicos y sociales.

### 2. Definición

Al egresar, la/el Bachiller técnico en Desarrollo de Software; es capaz de adquirir una comprensión integral del ciclo de vida del software, desde la identificación de necesidades hasta la entrega de productos digitales funcionales. Esto incluye el dominio de principios de programación, estructuras de datos, arquitectura de software, bases de datos, diseño de interfaces, pruebas, documentación y despliegue de soluciones digitales, tanto en entorno de escritorio, web y móvil.

Deben familiarizarse con el funcionamiento del ecosistema digital, la lógica detrás de los sistemas informáticos, la relación entre el hardware, software y los procesos organizacionales, así como el impacto social, económico y ético del desarrollo tecnológico. En ese marco, se incluye la comprensión de metodologías de desarrollo de software, gestión de versiones, seguridad informática y estándares de calidad aplicables al desarrollo.

Además, deben desarrollar habilidades blandas que potencien su capacidad de trabajo en equipo, liderazgo técnico, pensamiento crítico y comunicación efectiva, tanto con usuarios técnicos como con público no especializado. Es fundamental que puedan adaptarse a entornos dinámicos, aprender de forma continua y colaborar en la creación de soluciones innovadoras con enfoque emprendedor.

### Aspectos destacados del perfil profesional

- **Competencias técnicas específicas:** Tendrán capacidad para el análisis y desarrollo de aplicaciones empleando lenguajes de programación, integración con bases de datos, gestión de información, diseño e implementación de aplicaciones de escritorio, web y móvil, revisión y depuración de código, pruebas y validación, documentación técnica, prácticas de seguridad, comprensión de arquitecturas de software, entre otras.
- **Gestión de proyectos y metodologías de desarrollo de software:** Aplicarán modelos colaborativos para la planificación, seguimiento y entrega de productos de software, incluyendo herramientas de control de versiones, plataformas de gestión de tareas, marcos de trabajo, integración continua.
- **Innovación y emprendimiento tecnológico:** Desarrollarán propuestas emprendedoras con enfoque creativo y viable, soluciones que integren la inteligencia artificial.
- **Pensamiento computacional y resolución de problemas:** Demostrarán pensamiento lógico algorítmico y modelado de soluciones que permiten transformar requerimientos en funcionalidades concretas.
- **Comunicación y liderazgo:** Tendrán habilidades para trabajar colaborativamente, comunicar soluciones tecnológicas con claridad, adaptarse con resiliencia a entornos dinámicos y nuevas tecnologías, mantener atención al detalle para prevenir errores, y demostrarán empatía al interactuar con distintos perfiles de usuario.

### 3. Campo ocupacional

Los y las estudiantes del Bachillerato Técnico de la figura profesional de Desarrollo de Software podrán insertarse en el campo ocupacional que abarca desde roles técnicos, operativos o de apoyo hasta aquellos que involucran gestión, y son los siguientes:

- **Ocupaciones y puestos de trabajo relacionados.**
  - Auxiliar de desarrollo de software.
  - Asistente programador junior (web, móvil, escritorio).
  - Asistente de pruebas y control de calidad de software.
  - Asistente de bases de datos.
  - Testeador básico de software (QA Quality Assurance).
  - Diseñador de interfaces básicas (UI/UX User Interface/User Experience).
  - Colaborador en proyectos de tecnología.

- Asistente de documentación técnica y/o manuales de usuario.
- Desarrollador de soluciones informáticas orientadas a pequeños negocios o emprendimientos.
- **Sectores productivos y organizaciones**
  - **Sector privado:** El campo ocupacional tiene una amplia proyección, especialmente en empresas dedicadas a tecnologías de la información, entidades de desarrollo de software, agencias digitales, startups tecnológicas y consultoras de desarrollo. Además, puede integrarse a empresas de otros sectores productivos como comercio, industria, agricultura, educación, salud, turismo y finanzas, donde se requiera desarrollo y/o mantenimiento de aplicaciones, automatización de procesos o gestión de plataformas digitales. Estas oportunidades se extienden tanto a empresas grandes como medianas y pequeñas, brindando al bachiller técnico un campo de acción diverso y dinámico.
  - **Sector público:** Los y las estudiantes pueden vincularse a instituciones educativas, gobiernos locales o entidades públicas a través de proyectos o programas de vinculación, colaborando en mejorar el acceso y la calidad de los servicios mediante herramientas digitales, aportando en el manejo básico de sistemas o apoyando iniciativas tecnológicas con adaptación e implementación.
  - **Ámbito comunitario y no gubernamental:** Este perfil puede aportar en organizaciones sociales, espacios de formación juvenil, laboratorios tecnológicos o iniciativas que promuevan el uso responsable e innovador de la tecnología. Además, los y las estudiantes pueden desarrollar emprendimientos personales o brindar servicios digitales básicos de forma independiente, promoviendo así el autoempleo y el desarrollo local.

#### 4. Competencia general

Desarrollar soluciones informáticas mediante el análisis, diseño, codificación, pruebas, mantenimiento y documentación de software, con enfoque en aplicaciones web y móviles, considerando el desarrollo de aplicaciones de escritorio como base formativa, aplicando metodologías de desarrollo, conexión con bases de datos, marcos de trabajo, herramientas colaborativas, entornos de programación, requerimientos técnicos, conforme a la normativa vigente, demostrando habilidades de comunicación, trabajo en equipo, pensamiento crítico, responsabilidad y adaptabilidad a entornos tecnológicos cambiantes.

##### 4.1. Unidades de competencia

UNIDAD DE COMPETENCIA (UC)	
<b>UC 1:</b> Aplicar programación estructurada en la resolución de problemas computacionales, mediante el análisis de requerimientos, el diseño de algoritmos y flujogramas, uso de lenguajes estructurados en la construcción de soluciones funcionales, empleando buenas prácticas de codificación y lógica computacional.	
Elementos de competencia (EC)	Criterios de desempeño (CD)
<b>EC1:</b> Formular el problema informático a resolver, mediante el análisis de requerimientos, considerando entradas, procesos y salidas.	CD1.1: Reconoce las necesidades del usuario a partir de una descripción escrita o verbal del problema.
	CD1.2: Distingue los elementos clave del problema utilizando herramientas básicas de análisis como tablas de entrada/salida o diagramas simples.
	CD1.3: Establece con claridad las entradas, procesos y salidas necesarias brindando soluciones a problemas planteados.
	CD1.4: Define correctamente el propósito de la solución esperada, según las condiciones del contexto.
<b>EC2:</b> Diseñar soluciones a problemas mediante flujogramas y pseudocódigos, mediante el análisis estructurado de problemas y la organización lógica de instrucciones.	CD2.1: Elabora flujogramas utilizando simbología técnica y lógica coherente con la secuencia del algoritmo.
	CD2.2: Transforma el flujograma a pseudocódigos estructurados empleando lenguaje técnico y estructuras de control adecuadas.
	CD2.3: Compara la validez de flujogramas y pseudocódigos frente a distintas condiciones de entrada.
	CD2.4: Evalúa la solución de forma secuencial y ordenada, manteniendo la correspondencia entre flujograma y pseudocódigo.
<b>EC3:</b> Codificar algoritmos en un lenguaje de programación estructurado, asegurando la funcionalidad, claridad y mantenibilidad del código, conforme a estándares técnicos y buenas prácticas de programación.	CD3.1: Construye soluciones aplicando correctamente la sintaxis y semántica del lenguaje seleccionado.
	CD3.2: Integra estructuras condicionales y repetitivas modelando procesos según el flujo lógico del algoritmo.



	CD3.3: Verifica la funcionalidad del programa mediante pruebas que cubran distintos escenarios de entrada.
	CD3.4: Argumenta el uso de estructuras y operadores implementados con base en su funcionalidad dentro del algoritmo.
<b>EC4:</b> Validar el funcionamiento de los programas desarrollados, ejecutando casos de prueba que permitan verificar su comportamiento esperado y detectar posibles errores.	CD4.1: Ejecuta el programa con datos de prueba representativos comprobando su comportamiento funcional.
	CD4.2: Identifica errores de tipo sintáctico, semántico y lógico mediante técnicas de depuración manual o automatizada.
	CD4.3: Ajusta el código corrigiendo fallas e inconsistencias cumpliendo con los resultados previstos.
	CD4.4: Evalúa el rendimiento del programa aplicando mejoras que optimicen su ejecución y comprensión.
<b>EC5:</b> Documentar programas estructurados utilizando comentarios descriptivos y nomenclatura significativa con el fin de facilitar la comprensión de la lógica implementada.	CD5.1: Analiza el flujo del programa identificando los puntos clave donde se requiere explicar la lógica mediante comentarios
	CD5.2: Emplea comentarios en líneas y bloques de código explicando la funcionalidad de estructuras condicionales, bucles y variables.
	CD5.3: Organiza el código aplicando convenciones de indentación, uso de identificadores claros y agrupación lógica de secciones.
	CD5.4: Evalúa la claridad del programa comentado, proponiendo mejoras y siguiendo estándares técnicos
<b>Condiciones de ejecución de la Unidad de Competencia:</b>	
Espacios e instalaciones:	Entorno de aprendizaje Laboratorio informático
Insumos y recursos:	Computador Proyector

	Internet Aplicaciones ofimáticas Software para crear diagramas de flujo y pseudocódigo (Free DFD, PSeInt, entre otros) Lenguajes de programación estructuradas (C, Python) Entornos de desarrollo Guías, manuales, tutoriales, videotutoriales, documentales, entre otros.
Información utilizada:	Deitel, P., & Deitel, H. (2016). <i>C: Cómo programar</i> (8.ª ed.). Pearson. Joyanes Aguilar, L. (2012). Fundamentos de programación. McGraw-Hill. Conde, J. (2010). Programación en C para principiantes. Anaya Multimedia. Perry, G. (2011). Aprenda a programar con C. Alfaomega. Bosch, J. (2015). Java 2: Curso de programación. Marcombo. Ariño, M. Á. (2018). Trabajo en equipo y liderazgo. Pearson Educación.



UNIDAD DE COMPETENCIA (UC)	
<b>UC 2:</b> Desarrollar aplicaciones informáticas aplicando programación orientada a objetos, integrando clases, objetos y relaciones entre componentes, incorporando diseño modular, encapsulamiento, herencia, polimorfismo y reutilización de código, con el uso de buenas prácticas de codificación.	
Elementos de competencia (EC)	Criterios de desempeño (CD)
<b>EC1:</b> Analizar un problema computacional identificando los objetos, clases y relaciones necesarias estructurando la solución conforme a los principios de la programación orientada a objetos.	CD1.1: Distingue los objetos relevantes del problema según su comportamiento y atributos, a partir de un caso práctico.
	CD1.2: Explica relaciones jerárquicas entre clases considerando herencia y composición, con base en requerimientos funcionales.
	CD1.3: Muestra responsabilidades de clases conforme al principio de responsabilidad única, según la lógica del dominio.
	CD1.4: Estructura el modelo orientado a objetos mediante diagramas UML u otra notación técnica, de forma clara y estructurada.
<b>EC2:</b> Diseñar la arquitectura de una aplicación orientada a objetos estructurando módulos, clases y métodos según principios de herencia, encapsulamiento y polimorfismo.	CD2.1: Clasifica convenciones de diseño orientado a objetos según buenas prácticas técnicas, verificando la coherencia del modelo.
	CD2.2: Determina atributos y métodos de cada clase diferenciando accesos públicos, privados y protegidos, de acuerdo con la funcionalidad requerida.
	CD2.3: Plantea interfaces o clases abstractas que soporte extensibilidad y reutilización, según patrones de diseño conocidos.
	CD2.4: Ilustra estructuras modulares que organizan el código en componentes reutilizables, respetando la lógica de separación de responsabilidades.
<b>EC3:</b> Implementar clases, objetos y relaciones en un entorno de desarrollo integrando herencia, polimorfismo y encapsulamiento según el diseño previamente definido.	CD3.1: Codifica clases con constructores, atributos y métodos utilizando la sintaxis y convenciones del lenguaje orientado a objetos.
	CD3.2: Aplica herencia extendiendo funcionalidades sin duplicar código, asegurando el cumplimiento del principio de sustitución.

	CD3.3: Usa métodos mediante polimorfismo con referencias genéricas o interfaces, validando el comportamiento dinámico del sistema.
	CD3.4: Controla el acceso a los datos mediante modificadores de visibilidad encapsulando la información y protegiendo el estado del objeto.
<b>EC4:</b> Verificar el comportamiento de una aplicación orientada a objetos mediante pruebas de ejecución y revisión del código asegurando la funcionalidad y mantenibilidad.	CD4.1: Identifica errores lógicos y de sintaxis en el código fuente mediante la depuración con herramientas del entorno de desarrollo.
	CD4.2: Ejecuta pruebas funcionales sobre métodos y clases desarrolladas en condiciones controladas validando el cumplimiento de los requerimientos.
	CD4.3: Organiza los resultados obtenidos de las pruebas de ejecución utilizando registros estructurados facilitando el funcionamiento.
	CD4.4: Evalúa la implementación de principios de POO en el código fuente considerando encapsulamiento, herencia y polimorfismo.
<b>EC5:</b> Documentar aplicaciones orientadas a objetos empleando buenas prácticas de codificación, convenciones de estilo y uso de herramientas especializadas, facilitando la comprensión, mantenimiento y evolución.	CD5.1: Analiza la estructura del sistema orientado a objetos identificando clases, métodos, atributos y relaciones en su correcta descripción técnica.
	CD5.2: Emplea convenciones de estilo, nomenclatura y estándares en la escritura del código mejorando su legibilidad y mantenibilidad.
	CD5.3: Genera documentación del programa mediante el uso de herramientas de software especializadas incluyendo descripciones claras de la funcionalidad de cada componente.
	CD5.4: Evalúa la coherencia y completitud de la documentación técnica generada, proponiendo mejoras de forma colaborativa según estándares de calidad.
<b>Condiciones de ejecución de la Unidad de Competencia:</b>	

Espacios e instalaciones:	Entorno de aprendizaje Laboratorio informático
Insumos y recursos:	Computador Proyector Internet Aplicaciones ofimáticas Software para crear diagramas de flujo y pseudocódigo (Free DFD, PSeInt, entre otros) Lenguajes de programación estructurados (C, Python) Entornos de desarrollo Guías, manuales, tutoriales, videotutoriales, documentales, entre otros.
Información utilizada:	Deitel, P., & Deitel, H. (2016). <i>C: Cómo programar</i> (8.ª ed.). Pearson. Joyanes Aguilar, L. (2012). Fundamentos de programación. McGraw-Hill. Conde, J. (2010). Programación en C para principiantes. Anaya Multimedia. Perry, G. (2011). Aprenda a programar con C. Alfaomega. Bosch, J. (2015). Java 2: Curso de programación. Marcombo. Ariño, M. Á. (2018). Trabajo en equipo y liderazgo. Pearson Educación.

UNIDAD DE COMPETENCIA (UC)	
<b>UC 3:</b> Gestionar bases de datos relacionales y no relacionales enlazadas al desarrollo de aplicaciones informáticas, mediante el modelado de datos, diseño conceptual, lógico y físico, implementación de estructuras relacionales y operaciones CRUD, garantizando la integridad, consistencia y disponibilidad de la información, conforme a estándares técnicos y buenas prácticas de administración de datos.	
Elementos de competencia (EC)	Criterios de desempeño (CD)
<b>EC1:</b> Analizar los requerimientos de almacenamiento de información determinando las entidades, atributos y relaciones del modelo de datos.	CD1.1: Distingue entidades, atributos y relaciones a partir de especificaciones funcionales de una aplicación.
	CD1.2: Utiliza simbología estándar en la representación de modelos entidad-relación (ER) en función de los requerimientos.
	CD1.3: Establece claves primarias y foráneas considerando unicidad y vínculos lógicos entre entidades.
	CD1.4: Verifica la completitud del modelo de datos contrastándolo con las necesidades de la aplicación.
<b>EC2:</b> Diseñar la estructura lógica de la base de datos aplicando principios de normalización y consistencia referencial.	CD2.1: Define esquemas relacionales representando adecuadamente las tablas, campos, tipos de datos y restricciones.
	CD2.2: Aplica reglas de normalización (1FN, 2FN, 3FN) eliminando redundancias y dependencias inadecuadas.
	CD2.3: Establece relaciones entre tablas utilizando claves primarias y foráneas.
	CD2.4: Evalúa la integridad referencial y lógica del diseño mediante pruebas conceptuales de inserción y consulta.
<b>EC3:</b> Implementar bases de datos en sistemas de información manipulando datos con estructuras de control.	CD3.1: Crea bases de datos, tablas y restricciones utilizando comandos DDL en un SGBD.
	CD3.2: Realiza operaciones de insertado, actualización y eliminación de registros mediante comandos DML respetando las relaciones definidas.

	CD3.3: Consulta información mediante sentencias SELECT, utilizando filtros, funciones, ordenamientos y agrupaciones.
	CD3.4: Verifica la correcta ejecución de las operaciones CRUD asegurando la integridad de la información.
<b>EC4:</b> Verificar el funcionamiento y desempeño de la base de datos mediante consultas, pruebas de validación y respaldo de la información.	CD4.1: Revisa la consistencia de los datos frente a escenarios de actualización y eliminación.
	CD4.2: Ejecuta pruebas de recuperación de datos validando la exactitud de las consultas.
	CD4.3: Realiza copias de seguridad y restauración utilizando herramientas del sistema gestor.
	CD4.4: Documenta incidencias, mejoras o ajustes necesarios en el manejo de la base de datos.
<b>Condiciones de ejecución de la Unidad de Competencia:</b>	
Espacios e instalaciones:	Entorno de aprendizaje Laboratorio informático
Insumos y recursos:	Computador Proyector Internet Aplicaciones ofimáticas Software para crear diagramas de flujo y pseudocódigo (Free DFD, PSeInt, entre otros) Lenguajes de programación Motores de Base de Datos Entornos de desarrollo Guías, manuales, tutoriales, videotutoriales, documentales, entre otros.
Información utilizada:	Bosch, J. (2015). <i>Java 2: Curso de programación</i> . Marcombo.

	<p>Duckett, J. (2014). Desarrollo web con HTML, CSS y JavaScript. Anaya Multimedia.</p> <p>Bonilla, M. del P. (2019). Programación en JavaScript para novatos. Alfaomega.</p> <p>Thomson, L., &amp; Welling, L. (2017). PHP y MySQL: Desarrollo web (5.<sup>a</sup> ed.). Anaya Multimedia.</p> <p>Portales, R. (2020). Desarrollo de aplicaciones móviles con Android <i>Studio</i>. Marcombo.</p> <p>Sánchez, J. (2018). Acceso a datos con Java: JDBC y JPA. Ra-Ma Editorial.</p>
--	--



UNIDAD DE COMPETENCIA (UC)	
<b>UC 4:</b> Implementar aplicaciones de escritorio gestionando el ciclo de vida del software, desde el análisis de requerimientos, diseño, codificación, pruebas, mantenimiento y documentación, diseñando interfaces gráficas funcionales, integrando bases de datos y empleando herramientas colaborativas, control de versiones, así como metodologías de desarrollo asegurando la calidad técnica y usabilidad.	
Elementos de la competencia (EC)	Criterios de desempeño (CD)
<b>EC1:</b> Analizar los requerimientos funcionales y no funcionales del aplicativo a desarrollar, mediante técnicas de recolección de información y alineada a las necesidades del usuario.	CD1.1: Interpreta los requerimientos del usuario mediante entrevistas, encuestas o documentos de especificación, según los objetivos del proyecto.
	CD1.2: Clasifica los requerimientos en funcionales y no funcionales considerando criterios técnicos y de negocio.
	CD1.3: Representa los requerimientos mediante modelos o herramientas como historias de usuario, casos de uso o diagramas.
	CD1.4: Contrasta los requerimientos con los criterios de factibilidad técnica y recursos disponibles del entorno.
<b>EC2:</b> Diseñar la estructura y la interfaz gráfica de la solución, aplicando principios de usabilidad, organización modular y patrones de diseño orientados a la experiencia del usuario.	CD2.1: Estructura las pantallas, formularios y componentes gráficos en función del flujo lógico de la aplicación.
	CD2.2: Aplica principios de diseño visual como jerarquía, coherencia y accesibilidad en la elaboración de interfaces gráficas.
	CD2.3: Define el diseño modular de la aplicación con base en patrones de diseño.
	CD2.4: Justifica las decisiones de diseño con prototipos, mockups o wireframes acorde estándares de diseño.
<b>EC3:</b> Desarrollar la funcionalidad de una aplicación de escritorio integrando bases de datos y conectividad, empleando lenguajes de programación, entornos de desarrollo y bibliotecas gráficas.	CD3.1: Implementa módulos funcionales utilizando eventos, formularios, validaciones y estructuras de control.
	CD3.2: Conecta la aplicación a una base de datos mediante controladores, objetos de acceso a datos y aplicando buenas prácticas.

	CD3.3: Integra componentes visuales y elementos interactivos según la lógica de negocio establecida.
	CD3.4: Evalúa el cumplimiento funcional del software verificando el flujo de datos entre la interfaz, la lógica y la base de datos.
<b>EC4:</b> Evaluar la calidad funcional de la aplicación de escritorio mediante pruebas técnicas, ajustes correctivos y elaboración de documentación, asegurando su mantenibilidad y comprensión técnica.	CD4.1: Ejecuta pruebas unitarias, funcionales e integradas sobre módulos y formularios, utilizando casos de prueba definidos.
	CD4.2: Corrige errores de lógica, diseño o conexión identificados durante el proceso de prueba, asegurando el funcionamiento estable de la aplicación.
	CD4.3: Refactoriza el código optimizando rendimiento, legibilidad y mantenimiento, siguiendo buenas prácticas de desarrollo.
	CD4.4: Elabora documentación técnica y de usuario que describa la instalación, estructura, uso y mantenimiento del sistema.
	CD4.5: Publica los entregables y registros técnicos en plataformas colaborativas empleando herramientas de control de versiones.
<b>UC5:</b> Aplicar metodologías ágiles en el desarrollo de la aplicación siguiendo principios técnicos de organización, control y documentación del proceso.	CD5.1: Organiza tareas del proyecto en tableros digitales siguiendo los principios de una metodología de desarrollo.
	CD5.2: Gestiona versiones del código fuente utilizando un sistema de control bajo convenciones técnicas y mensajes estandarizados.
	CD5.3: Integra control de versiones respetando la asignación de tareas y revisiones colaborativas dentro de ciclos definidos.
	CD5.4: Documenta el proceso de desarrollo utilizando formatos técnicos estandarizados asegurando trazabilidad de cambios, decisiones técnicas y criterios de aceptación.
<b>Condiciones de ejecución de la Unidad de Competencia:</b>	

Espacios e instalaciones:	Entorno de aprendizaje Laboratorio informático
Insumos y recursos:	Computador Proyector Internet Aplicaciones ofimáticas Software para crear diagramas de flujo y pseudocódigo (Free DFD, PSeInt, entre otros) Lenguajes de programación (Java, Python, C#) Entornos de desarrollo Guías, manuales, tutoriales, videotutoriales, documentales, entre otros.
Información utilizada:	Larman, C. (2005). <i>UML y patrones</i> . Pearson Educación. Niemeyer, P., & Leuck, D. (2014). <i>Aprendiendo Java</i> . O'Reilly Media, en español. Coronel, C., & Morris, S. (2016). Base de datos: diseño y programación (11.ª ed.). Cengage Learning. Sánchez, J. (2018). Acceso a datos con Java: JDBC y JPA. Ra-Ma Editorial. Sommerville, I. (2011). Ingeniería del Software (9.ª ed.). Pearson Educación. Niemeyer, P., & Leuck, D. (2014). <i>Aprendiendo Java</i> . O'Reilly Media, en español.

UNIDAD DE COMPETENCIA (UC)	
<b>UC 5:</b> Crear aplicaciones para entornos web y móvil utilizando tecnologías, marcos de trabajo actuales, a través del diseño de interfaces adaptables, uso de lenguajes de marcado y de programación, integración de librerías, bases de datos modularizando el desarrollo del frontend y backend.	
Elementos de competencia (EC)	Criterios de desempeño (CD)
<b>EC1:</b> Emplear interfaces gráficas adaptables en entornos web y móviles considerando principios de usabilidad y experiencia de usuario con lenguajes de marcado e interactividad mediante scripts.	CD1.1: Aplica estructuras visuales y eventos dinámicos con librerías y estándares web, según principios de diseño centrado en el usuario.
	CD1.2: Organiza la estructura de la interfaz, siguiendo convenciones de estilo y estructuras modulares.
	CD1.3: Integra herramientas o frameworks de diseño con el fin de mejorar la interfaz y experiencia de usuario.
	CD1.4: Adapta el diseño de la interfaz a diferentes tamaños de pantalla empleando técnicas de diseño responsivo, verificando el comportamiento en diferentes navegadores y/o dispositivos.
<b>EC2:</b> Implementar lógica de negocio y gestión de datos, aplicando buenas prácticas de desarrollo, utilizando lenguajes de programación de servidor, frameworks y entornos adecuados.	CD2.1: Diferencia las responsabilidades de los componentes del sistema, empleando patrones y principios de arquitectura según el tipo de aplicación.
	CD2.2: Implementa funcionalidades del backend que respondan a solicitudes desde el frontend.
	CD2.3: Establece la conexión con la base de datos según el lenguaje de programación, entorno o framework, asegurando la correcta ejecución de las operaciones CRUD.
	CD2.4: Estructura el diseño del frontend empleando librerías o frameworks.
	CD2.5: Evalúa el comportamiento del sistema mediante pruebas funcionales y validaciones técnicas.

<b>EC3:</b> Diseñar aplicaciones web o móviles en entornos de prueba o producción utilizando herramientas y plataformas adecuadas.	CD3.1: Analiza el entorno de despliegue seleccionando configuraciones compatibles con los requerimientos técnicos del proyecto.
	CD3.2: Publica archivos de la aplicación en servidores locales o en la nube utilizando herramientas o plataformas específicas.
	CD3.3: Evalúa el funcionamiento de la aplicación en entornos de producción asegurado funcionalidad.
	CD3.4: Documenta el proceso de despliegue especificando herramientas utilizadas, configuración aplicada y pasos seguidos.
<b>EC4:</b> Desarrollar aplicaciones web y móviles, considerando estándares de codificación, seguridad y mantenimiento.	CD4.1: Modifica el código fuente siguiendo convenciones de nomenclatura, indentación y organización según el lenguaje o framework utilizado.
	CD4.2: Aplica principios básicos de seguridad como sanitización de datos, autenticación o gestión segura de contraseñas.
	CD4.3: Documenta las funciones y estructuras del código utilizando comentarios técnicos claros y coherentes.
<b>Condiciones de ejecución de la Unidad de Competencia:</b>	
Espacios e instalaciones:	Entorno de aprendizaje Laboratorio informático
Insumos y recursos:	Computador Proyector Internet Aplicaciones ofimáticas Software para crear diagramas de flujo y pseudocódigo (Free DFD, PSeInt, entre otros) Lenguajes de programación Entornos de desarrollo

	Guías, manuales, tutoriales, videotutoriales, documentales, entre otros.
Información utilizada:	<p>Bosch, J. (2015). <i>Java 2: Curso de programación</i>. Marcombo.</p> <p>Duckett, J. (2014). <i>Desarrollo web con HTML, CSS y JavaScript</i>. Anaya Multimedia.</p> <p>Bonilla, M. del P. (2019). <i>Programación en JavaScript para novatos</i>. Alfaomega.</p> <p>Thomson, L., &amp; Welling, L. (2017). <i>PHP y MySQL: Desarrollo web</i> (5.<sup>a</sup> ed.). Anaya Multimedia.</p> <p>Portales, R. (2020). <i>Desarrollo de aplicaciones móviles con Android Studio</i>. Marcombo.</p> <p>Sánchez, J. (2018). <i>Acceso a datos con Java: JDBC y JPA</i>. Ra-Ma Editorial.</p>



## 5. Relación de las Unidades de competencia de la Figura profesional Redes y módulos de especialización

**Tabla 1. Relación Unidades de competencia – módulos de especialización**

No.	Unidad de Competencia	Módulo de especialización
1	Aplicar programación estructurada en la resolución de problemas computacionales, mediante el análisis de requerimientos, el diseño de algoritmos y flujogramas, uso de lenguajes estructurados en la construcción de soluciones funcionales, empleando buenas prácticas de codificación y lógica computacional.	Programación Estructurada
2	Desarrollar aplicaciones informáticas aplicando programación orientada a objetos, integrando clases, objetos y relaciones entre componentes, incorporando diseño modular, encapsulamiento, herencia, polimorfismo y reutilización de código, con el uso de buenas prácticas de codificación.	Programación Orientada a Objetos
3	Gestionar bases de datos relacionales y no relacionales enlazadas al desarrollo de aplicaciones informáticas, mediante el modelado de datos, diseño conceptual, lógico y físico, implementación de estructuras relacionales y operaciones CRUD, garantizando la integridad, consistencia y disponibilidad de la información, conforme a estándares técnicos y buenas prácticas de administración de datos.	Base de Datos
4	Implementar aplicaciones de escritorio gestionando el ciclo de vida del software, desde el análisis de requerimientos, diseño, codificación, pruebas, mantenimiento y documentación, diseñando interfaces gráficas funcionales, integrando bases de datos y empleando herramientas colaborativas, control de versiones, así como metodologías de desarrollo asegurando la calidad técnica y usabilidad.	Aplicaciones de Escritorio
5	Crear aplicaciones para entornos web y móvil utilizando tecnologías y marcos de trabajo actuales, a través del diseño de interfaces adaptables, uso de lenguajes de marcado y de programación, integración de librerías, bases de datos modularizando el desarrollo del frontend y backend.	Aplicaciones Web y Móviles