

|  |  |  |
| --- | --- | --- |
|  | INGENIERÍA EN DESARROLLO Y GESTIÓN DE SOFTWARE  EN COMPETENCIAS PROFESIONALES | descarga |

**ASIGNATURA DE ARQUITECTURAS DE SOFTWARE**

|  |  |
| --- | --- |
| 1. **Competencias** | Construir soluciones de software y sistemas inteligentes mediante la gestión de proyectos, integración de metodologías, modelos y herramientas de desarrollo bajo la normatividad aplicable para la optimización de proyectos de investigación, innovación, desarrollo tecnológico y de emprendimiento. |
| 1. **Cuatrimestre** | Séptimo |
| 1. **Horas Teóricas** | 21 |
| 1. **Horas Prácticas** | 54 |
| 1. **Horas Totales** | 75 |
| 1. **Horas Totales por Semana Cuatrimestre** | 5 |
| 1. **Objetivo de aprendizaje** | El alumno establecerá arquitecturas de software con base en el análisis de requerimientos para satisfacer los atributos de calidad del software y servir como guía en el desarrollo. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Unidades de Aprendizaje** | **Horas** | | |
| **Teóricas** | **Prácticas** | **Totales** |
| I. Fundamentos de arquitecturas de software | 10 | 5 | 15 |
| **II. Modelado de arquitecturas de software** | 5 | 25 | 30 |
| III. Patrones de diseño | 6 | 24 | 30 |
| **Totales** | **21** | **54** | **75** |

**ARQUITECTURAS DE SOFTWARE**

*UNIDADES DE APRENDIZAJE*

|  |  |
| --- | --- |
| 1. **Unidad de aprendizaje** | **I. Fundamentos de arquitecturas de software** |
| 1. **Horas Teóricas** | 10 |
| 1. **Horas Prácticas** | 5 |
| 1. **Horas Totales** | 15 |
| 1. **Objetivo de la Unidad de Aprendizaje** | El alumno determinará la arquitectura de software de acuerdo a requerimientos para guiar la construcción de los componentes de software. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Temas** | **Saber** | **Saber hacer** | **Ser** |
| Introducción a las arquitecturas de software. | Definir el concepto de arquitectura de software.  Describir las características de las arquitecturas de software. |  | Analítico  Razonamiento deductivo  Uso de procesos cognitivos  Razonamiento lógico |
| Frameworks de arquitecturas. | Describir los frameworks de arquitectura de software. |  | Analítico  Razonamiento deductivo  Uso de procesos cognitivos  Razonamiento lógico |
| Estilos de arquitectura de software. | Identificar requerimientos funcionales y no funcionales.  Distinguir los estilos de arquitecturas de software:  - Cliente-Servidor  - Microservicios  - N capas  - Orientada a servicios  - Dirigida a eventos  - Basada en espacio  - Microkernel  - Serverless | Establecer la arquitectura de software cumpliendo con los requerimientos funcionales y no funcionales. | Analítico  Razonamiento deductivo  Uso de procesos cognitivos  Razonamiento lógico |

**ARQUITECTURAS DE SOFTWARE**

*PROCESO DE EVALUACIÓN*

|  |  |  |
| --- | --- | --- |
| **Resultado de aprendizaje** | **Secuencia de aprendizaje** | **Instrumentos y tipos de reactivos** |
| Elabora un documento que incluya:  - Estilos de arquitectura de software  - Características de las arquitecturas de software  Elaborar un documento a partir de casos prácticos que incluya:  - Requerimientos Funcionales y No Funcionales  - Justificación de la arquitectura seleccionada | 1. Identificar los estilos de arquitecturas de software.  2. Comprender las características de las arquitecturas de software.  3. Analizar los requerimientos funcionales y no funcionales.  4. Explicar el uso de las arquitecturas de software. | - Ejercicios prácticos  - Lista de cotejo |

**ARQUITECTURAS DE SOFTWARE**

*PROCESO ENSEÑANZA APRENDIZAJE*

|  |  |
| --- | --- |
| **Métodos y técnicas de enseñanza** | **Medios y materiales didácticos** |
| - Prácticas en laboratorios dirigidas y no dirigidas  - Solución de problemas  - Tareas de investigación | - Equipos de cómputo  - Proyector  - Internet  - Pizarrón y marcadores  - Plataformas virtuales |

*ESPACIO FORMATIVO*

|  |  |  |
| --- | --- | --- |
| **Aula** | **Laboratorio / Taller** | **Empresa** |
|  | **X** |  |

**ARQUITECTURAS DE SOFTWARE**

*UNIDADES DE APRENDIZAJE*

|  |  |
| --- | --- |
| 1. **Unidad de aprendizaje** | **II. Modelado de arquitecturas de software** |
| 1. **Horas Teóricas** | 5 |
| 1. **Horas Prácticas** | 25 |
| 1. **Horas Totales** | 30 |
| 1. **Objetivo de la Unidad de Aprendizaje** | El alumno elaborará el modelo de arquitecturas de software para establecer el marco de referencia y guiar el proceso de desarrollo de software. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Temas** | **Saber** | **Saber hacer** | **Ser** |
| Modelos y lenguajes de descripción de arquitectura. | Identificar modelos de arquitectura.  Identificar lenguajes de descripción de arquitectura. | Seleccionar modelos y lenguajes de descripción de arquitectura. | Analítico  Razonamiento deductivo  Uso de procesos cognitivos  Razonamiento lógico  Trabajo en equipo |
| Diagramas y herramientas de diseño arquitectónico de software. | Describir diagramas de diseño arquitectónico de software.  Identificar herramientas de modelado de arquitecturas de software. | Diseñar arquitecturas de software con herramientas de modelado.  Implementar arquitecturas en el desarrollo de software. | Analítico  Razonamiento deductivo  Uso de procesos cognitivos  Razonamiento lógico  Trabajo en equipo |

**ARQUITECTURAS DE SOFTWARE**

*PROCESO DE EVALUACIÓN*

|  |  |  |
| --- | --- | --- |
| **Resultado de aprendizaje** | **Secuencia de aprendizaje** | **Instrumentos y tipos de reactivos** |
| Elabora un documento a partir de un caso de estudio que incluya:  - Modelo y lenguaje seleccionado  - Diagramas de la arquitectura de software  Elabora un prototipo funcional de software en el que aplique una arquitectura. | 1. Identificar los modelos.  2. Identificar los lenguajes de descripción.  3. Comprender las primitivas de los diagramas de arquitectura.  4. Explicar el uso de las herramientas de modelado.  5 Relacionar los diagramas de arquitectura con el prototipo funcional de software. | - Estudio de casos  - Lista de cotejo |

**ARQUITECTURAS DE SOFTWARE**

*PROCESO ENSEÑANZA APRENDIZAJE*

|  |  |
| --- | --- |
| **Métodos y técnicas de enseñanza** | **Medios y materiales didácticos** |
| - Prácticas en laboratorios dirigidas y no dirigidas  - Estudio de casos  - Solución de problemas | - Equipos de cómputo  - Proyector  - Internet  - Pizarrón y marcadores  - Plataformas virtuales  - Herramientas de modelado  - IDE de desarrollo  - Manejadores de base de datos |

*ESPACIO FORMATIVO*

|  |  |  |
| --- | --- | --- |
| **Aula** | **Laboratorio / Taller** | **Empresa** |
|  | **X** |  |

# ARQUITECTURAS DE SOFTWARE

*UNIDADES DE APRENDIZAJE*

|  |  |
| --- | --- |
| 1. Unidad de aprendizaje | **III. Patrones de diseño** |
| 2. Horas Teóricas | 6 |
| 3. Horas Prácticas | 24 |
| 4. Horas Totales | 30 |
| 5. Objetivo de la Unidad de Aprendizaje | El alumno determinará patrones de diseño para resolver problemas específicos en el desarrollo de software. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Temas** | **Saber** | **Saber hacer** | **Ser** |
| Introducción a los patrones de diseño. | Definir el concepto de patrón de diseño.  Enlistar los patrones de diseño:  - Creacionales  - Estructurales  - De comportamientos  - Emergentes  Definir los antipatrones y los patrones GRASP |  | Analítico  Razonamiento deductivo  Uso de procesos cognitivos  Razonamiento lógico  Razonamiento inductivo  Trabajo en equipo |
| Patrones de diseño GoF. | Describir las características de los patrones de diseño:  - Creacionales  - Estructurales  - De comportamiento | Realizar diagramas de los patrones de diseño:  - Creacionales  - Estructurales  - De comportamiento  Implementar patrones de diseño en el desarrollo de software:  - Creacionales  - Estructurales  - De comportamiento | Analítico  Razonamiento deductivo  Uso de procesos cognitivos  Razonamiento lógico  Razonamiento inductivo  Trabajo en equipo |
| Patrones de diseño emergentes. | Describir las características de los patrones de diseño emergentes:  - MVC  - DAO  - CQRS  - DDD  - MVVM  - MVP | Realizar diagramas de los patrones de diseño:  - MVC  - DAO  - CQRS  - DDD  - MVVM  - MVP  Implementar patrones de diseño en el desarrollo de software:  - MVC  - DAO  - CQRS  - DDD  - MVVM  - MVP | Analítico  Razonamiento deductivo  Uso de procesos cognitivos  Razonamiento lógico  Razonamiento inductivo  Trabajo en equipo |

**ARQUITECTURAS DE SOFTWARE**

*PROCESO DE EVALUACIÓN*

|  |  |  |
| --- | --- | --- |
| **Resultado de aprendizaje** | **Secuencia de aprendizaje** | **Instrumentos y tipos de reactivos** |
| Elabora un documento a partir de casos prácticos que incluya:  - Justificación de Patrones de diseño seleccionados  - Diagramas de los patrones de diseño seleccionados  Elabora un prototipo funcional de software utilizando al menos un patrón de diseño:  - Creacional.  - Estructural.  - De comportamiento. | 1. Identificar los patrones de diseño.  2. Comprender las características de los patrones de diseño.  4. Explicar el uso de las herramientas de modelado.  5 Relacionar los patrones de diseño con el prototipo funcional de software. | - Ejercicios prácticos  - Lista de cotejo |

**ARQUITECTURAS DE SOFTWARE**

*PROCESO ENSEÑANZA APRENDIZAJE*

|  |  |
| --- | --- |
| **Métodos y técnicas de enseñanza** | **Medios y materiales didácticos** |
| - Prácticas en laboratorios dirigidas y no dirigidas  - Solución de problemas  - Tareas de investigación | - Equipos de cómputo  - Proyector  - Internet  - Pizarrón y marcadores  - Plataformas virtuales  - herramientas de modelado  - IDE de desarrollo  - Manejadores de base de datos  - Frameworks |

*ESPACIO FORMATIVO*

|  |  |  |
| --- | --- | --- |
| **Aula** | **Laboratorio / Taller** | **Empresa** |
|  | **X** |  |

**ARQUITECTURAS DE SOFTWARE**

*CAPACIDADES DERIVADAS DE LAS COMPETENCIAS PROFESIONALES A LAS QUE CONTRIBUYE LA ASIGNATURA*

|  |  |
| --- | --- |
| **Capacidad** | **Criterios de Desempeño** |
| Determinar procesos y herramientas de machine learning, data mining y big data mediante el análisis del problema de acuerdo a las características, ubicación de los datos y normativa aplicable para establecer un plan de desarrollo e integración que cubra las necesidades de información. | Entrega un plan de desarrollo e integración que incluya:  - Planteamiento del problema.  - Definición de necesidades de información.  - Descripción de la ubicación y características de los datos.  - Justificación de la normativa aplicable.  - Justificación de las herramientas y procesos a utilizar. |
| Implementar modelos de procesamiento mediante la incorporación de API's y frameworks de desarrollo tomando en cuenta la normatividad aplicable para la integración de sistemas inteligentes . | Entrega productos generados que incluyan:  - Diagrama de arquitectura software.  - Requerimientos técnicos para integrar el modelo a la aplicación.  - Descripción de los datos de entrada y salida. |
| Determinar la arquitectura de software mediante el análisis de los requerimientos, la definición de los componentes y la relación entre ellos para satisfacer los atributos de calidad del software y servir como guía en el desarrollo | Entrega un documento de arquitectura de software que incluya:  - Requerimientos funcionales y no funcionales de desarrollo de software.  - Justificación de la arquitectura seleccionada.  - Diagrama de arquitectura de software.  - Características y funciones de los componentes.  - Diagramas necesarios de acuerdo a la arquitectura seleccionada. |
| Implementar modelos de datos mediante herramientas y sistemas gestores para garantizar la disponibilidad e integridad de la información | 1. Entrega un documento que incluya:  - Justificación de las herramientas a utilizar para los modelos de datos.  - Modelos de datos.  - Descripción de metadatos.  2. Entrega archivos, credenciales de registro y secuencia de configuración para la creación de los modelos de datos. |
| Desarrollar componentes mediante el uso de patrones de diseño, APIs y frameworks de desarrollo conforme a la arquitectura establecida para la integración continua de unidades del software | 1. Entrega un documento que incluya:  - Justificación de las herramientas para el desarrollo de componentes.  - Justificación de los patrones de diseño a utilizar.  - Justificación de los lenguajes de programación a utilizar.  2. Entrega archivos de código fuente y configuración de los componentes desarrollados.  3. Entrega evidencia de almacenamiento de archivos de código fuente y actualizaciones en la documentación del proyecto en la plataforma de versionamiento seleccionada. |
| Implementar soluciones de software mediante la integración de componentes de acuerdo a la arquitectura definida para su liberación en un ambiente de producción | 1. Entrega manuales del software establecidos en el plan de desarrollo.  2. Entrega la versión estable del software integrando los componentes de acuerdo al plan de entregas continuas.  3. Entrega evidencia de almacenamiento de archivos de código fuente y actualizaciones de la documentación del proyecto en la plataforma de versionamiento establecida. |

**ARQUITECTURAS DE SOFTWARE**

*FUENTES BIBLIOGRÁFICAS*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Autor** | **Año** | **Título del Documento** | **Ciudad** | **País** | **Editorial** |
| Mark Richards, Neal Ford | 2020 ISBN:97:81-492043-454 | *Fundamentals of Software Architecture* | New York | United States | O'Reilly Media |
| Robert C. Martin | 2018 ISBN:013-449416-4-860 | *Clean Architecture* | Boston | United States | Addison - Wesley - Pearson |
| Angel Arias, Alicia Durango | 2016 ISBN:97:81-52336-5-487 | *Ingeniería y Arquitectura del Software* | Vigo | España | IT Campus Academy |
| Humberto Cervantes, Rick Kazman | 2016 ISBN:97:80-13439-0-789 | *Designing Software Architectures: A Practical Approach* | Boston | United States | Addison - Wesley - Pearson |
| Laurent Debrauwer | 2018 ISBN:97:82-40901-6-349 | *Patrones de Diseño en Java* | Barcelona | España | Eni Ediciones |
| David Roldan Martinez, Pedro J. Valderas | 2018 ISBN:97:88-49964-7-654 | *Microservicios, un enfoque integrado* | Jarama | España | Ra-Ma |
| Fien Van Der Heyde | 2019 ISBN:97:82-40901-8-046 | *UML 2.5 Domine el diseño con los Patrones de diseño* | Barcelona | España | Eni Ediciones |
| IEEE/ISO/IEC | 2011 ISBN(s):9781504413671, 9780738171678, 9780738171425 | *ISO/IEC/IEEE Std 42010:2011(E), Systems and software engineering -- Architecture description* | New York | United States | ISO/IEC/IEEE |

CIZG