Grand Master
Ingeniería de Software



tech universidad tecnológica



tech universidad tecnológica

Grand Master

Ingeniería de Software

Modalidad: **Online**
Duración: **2 años**
Titulación: **TECH Universidad Tecnológica**
Horas lectivas: **3.000 h.**
Acceso web: **www.techtitute.com/informatica/grand-master/grand-master-ingenieria-software**

Índice

01 Presentación — pág. 4

02 Objetivos — pág. 8

03 Competencias — pág. 14

04 Estructura y contenido — pág. 18

05 Metodología — pág. 40

06 Titulación — pág. 48

01

Presentación

La demanda de software en los últimos años se ha disparado. Con el surgimiento de nuevas plataformas digitales, hardware más sofisticado y una virtualización de procesos cotidianos cada vez mayor, los ingenieros de *software* se enfrentan a nuevos retos continuamente. El público está cada vez más acostumbrado a las nuevas tecnologías y sus exigencias se vuelven mayores, por lo que los profesionales del desarrollo de *software* deben adaptarse a estas demandas y crear productos que estén a la altura de las expectativas del mercado. Esto requiere un nivel técnico elevado en múltiples campos del conocimiento informático.





Vas a tener un papel clave en el futuro tecnológico de muchas empresas. Especialízate en Ingeniería de Software y empieza a desarrollar los sistemas que marquen un antes y un después"

La industria tecnológica es de las más relevantes hoy en día, pues casi todo el mundo interactúa a diario con alguna clase de dispositivo digital. En este contexto, los ingenieros de software son la primera línea de batalla de todo el proceso de desarrollo tecnológico, ya que son los que constantemente tienen que estar actualizando sistemas, desarrollando otros nuevos y ofreciendo soluciones inteligentes a las problemáticas que se vayan presentando. Visto de esta forma, los profesionales de la ingeniería informática deben ser personas altamente resolutivas, con grandes conocimientos técnicos y una capacidad de adaptación destacada a todo tipo de desarrollo y entornos.

Con este objetivo en mente, TECH ha diseñado este Grand Master en Ingeniería de Software, ofreciendo una capacitación completa y de alto nivel a todos los desarrolladores que quieran especializar su carrera y dirigirla a la creación de sistemas. Por un lado, el programa trata las diferentes metodologías para crear y dirigir un proyecto de desarrollo de software, así como todos los aspectos a tener en cuenta respecto a la computación, requisitos y plataformas. Por otro lado, también se incide en de la seguridad tanto del propio software en sí mismo como de los sistemas de información y entorno de trabajo empleados durante el proceso. Al egresar, el alumno tendrá todos los conocimientos necesarios para ser un experto en Ingeniería de Software eficaz y altamente competente.

A todo ello hay que sumar una de las principales ventajas de este programa: su naturaleza 100% online. Esto implica que el alumno no ha de adaptarse a horarios fijos ni tiene la obligación de asistir a un centro físico concreto. Así, el estudiante tiene la libertad para gestionar el estudio de la materia que elija, a su propio ritmo y teniendo en cuenta sus obligaciones, planificando los horarios como mejor le convenga.

Este **Grand Master en Ingeniería de Software** contiene el programa más completo y actualizado del mercado. Sus características más destacadas son:

- El desarrollo de casos prácticos presentados por expertos en desarrollo de software
- Los contenidos gráficos, esquemáticos y eminentemente prácticos con los que están concebidos recogen una información científica y práctica sobre aquellas disciplinas indispensables para el ejercicio profesional
- Los ejercicios prácticos donde realizar el proceso de autoevaluación para mejorar el aprendizaje
- Su especial hincapié en metodologías innovadoras en el campo de la Ingeniería de Software
- Las lecciones teóricas, preguntas al experto, foros de discusión de temas controvertidos y trabajos de reflexión individual
- La disponibilidad de acceso a los contenidos desde cualquier dispositivo, fijo o portátil, con conexión a internet

¿Te imaginas haber participado en el desarrollo de Netflix? Es el momento de dejar de imaginar y enfocar tu carrera hacia los mejores proyectos de software"

Tu experiencia y conocimientos pueden marcar la diferencia en grandes proyectos que impliquen muchos requisitos. No pierdas la oportunidad de distinguirte en tu carrera y matricúlate ya en este Grand Master en Ingeniería de Software"

El objetivo de TECH es conseguir que seas un gran ingeniero informático. Tienes garantizado el acceso al mejor material y enseñanza posible para ello.

Estúdialo cuando, donde y como quieras. El programa es 100% online y se adapta a tus necesidades, no al revés.

Incluye en su cuadro docente a profesionales pertenecientes al ámbito de la Ingeniería de Software, que vierten en este programa la experiencia de su trabajo, además de reconocidos especialistas de sociedades de referencia y universidades de prestigio.

Su contenido multimedia, elaborado con la última tecnología educativa, permitirá al profesional un aprendizaje situado y contextual, es decir, un entorno simulado que proporcionará un estudio inmersivo programado para entrenarse ante situaciones reales.

El diseño de este programa se centra en el Aprendizaje Basado en Problemas, mediante el cual el alumno deberá tratar de resolver las distintas situaciones de práctica profesional que se le planteen a lo largo del programa. Para ello, el profesional contará con la ayuda de un novedoso sistema de vídeo interactivo realizado por reconocidos expertos.



02

Objetivos

El presente Grand Master en Ingeniería de Software se ha elaborado con el objetivo de ofrecer a todo profesional del ámbito informático la capacitación superior necesaria para enfocar su carrera hacia el desarrollo de software moderno y adaptado a las nuevas realidades fluctuantes del mercado. Con los conocimientos altamente técnicos instruidos durante toda la enseñanza, el alumno verá incrementadas en gran medida sus opciones de ascender profesionalmente y acceder a puestos de trabajo en las grandes compañías del sector.



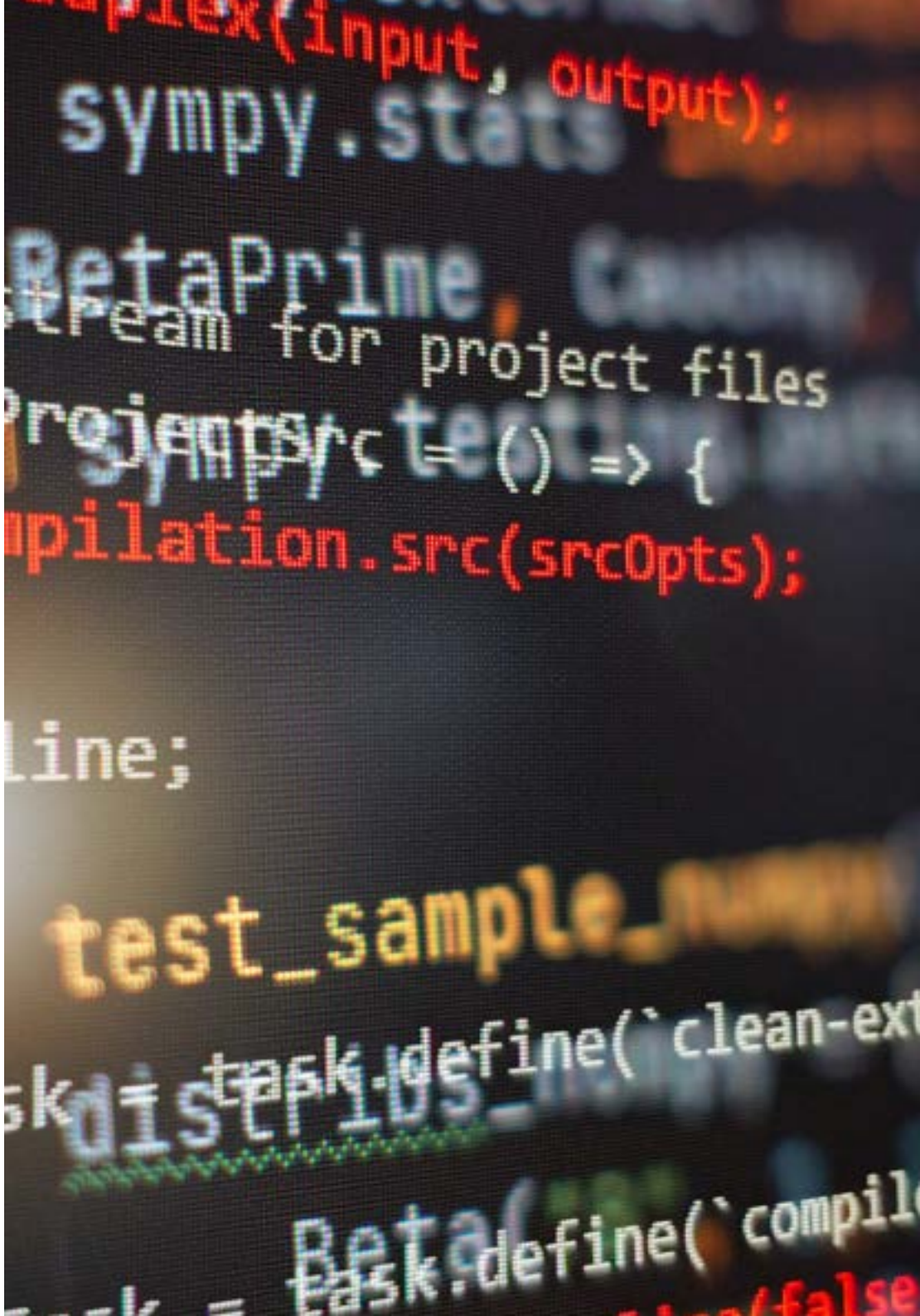


Un Grand Master que supondrá el mayor impulso positivo que le puedas dar a tu carrera hacia el éxito profesional"

Objetivos generales

- Adquirir nuevas competencias necesarias y demandadas en cuanto a nuevas tecnologías y últimas novedades en software
- Complementar los conocimientos adquiridos con habilidades en el campo de la computación y la estructura de computadoras, todo ello incluyendo la base matemática, estadística y física imprescindible en una ingeniería
- Ampliar los conocimientos en Ingeniería de Software y Sistemas Informáticos con las últimas novedades y metodología más innovadora
- Abordar proyectos y entornos de software complejos, sabiendo aportar soluciones inteligentes a problemáticas diversas

Una especialización que te ayudará a dominar el desarrollo de software con un set de competencias únicas y demandadas por toda empresa puntera del sector"



Objetivos específicos

- Conocer las bases de la Ingeniería de Software, así como el conjunto de normas o principios éticos y de responsabilidad profesional durante y después del desarrollo
- Comprender el proceso de desarrollo de software, bajo los diferentes modelos de programación y el paradigma de la programación orientada a objetos
- Entender los diferentes tipos de modelados de aplicaciones y patrones de diseño en el lenguaje unificado de modelamiento (UML)
- Conocer los conceptos fundamentales de la dirección de proyectos y el ciclo de vida de la gestión de proyectos
- Comprender el funcionamiento de la gestión de la calidad en los proyectos, incluyendo la planificación, el aseguramiento, el control, los conceptos estadísticos y las herramientas disponibles
- Adquirir los conocimientos esenciales relacionados con la responsabilidad profesional derivada de la gestión de proyectos
- Comprender las diferentes plataformas de desarrollo de software
- Adquirir los conocimientos necesarios para el desarrollo de aplicaciones e interfaces gráficas en los lenguajes *Java* y *.NET*
- Aprender los entornos de desarrollo de aplicaciones móviles en *Android* y los procesos de depuración y publicación
- Entender el desarrollo de aplicaciones basada en la nube y determinar los correctos procedimientos para su implementación
- Comprender los procedimientos y técnicas para mejor la apariencia de un documento escrito en HTML
- Adquirir los conocimientos necesarios para el desarrollo de aplicaciones en el lado del cliente web
- Desarrollar aplicaciones de estructuras complejas, mediante el uso de los diferentes procedimientos, funciones y objetos que integran el *JavaScript*
- Aprender a utilizar la interfaz de programación DOM para los documentos HTML y XML, al fin de modificar, tanto su estructura, estilo y contenido
- Conocer el concepto de usabilidad web, sus ventajas, principios, métodos y técnicas para hacer un sitio web usable por el usuario
- Entender la arquitectura de software del Modelo Vista Controlador (MVC) que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos
- Adquirir las destrezas para el uso de los servicios web, mediante el uso de XML, SOA y REST
- Conocer el proceso de seguridad de la información, sus implicaciones en la confidencialidad, integridad, disponibilidad y costos económicos
- Aprender el uso de las buenas prácticas de la seguridad en la gestión de los servicios de tecnologías de información
- Adquirir los conocimientos para la correcta certificación de los procesos de seguridad
- Comprender los mecanismos y métodos de autenticación para el control de acceso, así como el proceso de auditoría de accesos
- Entender los programas de gestión de la seguridad, la gestión de riesgo y el diseño de políticas de seguridad
- Aprender los planes de continuidad de negocio, sus fases y proceso de mantenimiento
- Conocer los procedimientos para la correcta protección de la empresa a través, de las redes DMZ, el uso de sistemas de detección de intrusos y otras metodologías
- Entender los problemas relacionados con la seguridad en el software, sus vulnerabilidades y como se clasifican

- Analizar los diferentes servidores web que son tendencia en el mercado actual
- Entender el proceso de estadísticas de uso y balanceo de cargas en los servidores web
- Adquirir los conocimientos requeridos para la correcta ejecución del proceso de auditoría y control interno informático
- Entender los conceptos y procesos del diseño de software, aprendiendo también sobre el diseño de la arquitectura y sobre el diseño a nivel de componentes y basado en patrones
- Comprender los distintos patrones de arquitecturas de sistemas y de diseño de software, así como la arquitectura de las aplicaciones en la nube
- Profundizar en la mejora del proceso de desarrollo de software y de calidad del software usando los estándares ISO/IEC
- Comprender la importancia de la ingeniería de requisitos en el proceso de desarrollo de software
- Profundizar en las fuentes de requisitos y las técnicas de elicitación de requisitos, ya que son parte esencial del proceso
- Entender y aplicar la realización de prototipos como parte esencial del proceso de desarrollo
- Sentar las bases para el análisis forense en el mundo del software y de las auditorías informáticas
- Conocer los conceptos fundamentales de la dirección de proyectos y el ciclo de vida de la gestión de proyectos
- Aprender el desarrollo del cronograma para la gestión del tiempo, el desarrollo del presupuesto y la respuesta ante los riesgos
- Comprender el funcionamiento de la gestión de la calidad en los proyectos, incluyendo la planificación, el aseguramiento, el control, los conceptos estadísticos y las herramientas disponibles





Una capacitación completa que te llevará a través de los conocimientos necesarios, para competir entre los mejores"

03

Competencias

Los ingenieros de software están en constante actualización de conocimientos ya que las propias herramientas y realidades en las que trabajan se modernizan día a día. Esto exige un proceso de aprendizaje regular para el que son necesarias diversas competencias generales, tanto en materia de puro desarrollo de software como en otras disciplinas como la gestión de equipos. Comprendiendo esta circunstancia, TECH ha elaborado el Grand Master en Ingeniería de Software pensando en dotar al alumno de todas las competencias posibles y necesarias para que su propio proceso de aprendizaje continuo sea más ligero y automático.





Con todas las competencias que vas a adquirir estudiando este Grand Master en Ingeniería de Software vas a ser el mejor candidato para cualquier puesto al que postules"

Competencias generales

- Desarrollar un sistema software teniendo en cuenta todas las etapas de desarrollo, plataformas de seguridad y cuestiones de seguridad
- Tratar correcta y profesionalmente todos los datos generados durante el desarrollo
- Aplicar la mejor metodología de trabajo según sea conveniente para el proyecto o las personas que lo integran
- Conocer la realidad completa de la Ingeniería de Software y prevenir posibles riesgos o problemas de manera rápida y eficaz

Puedes dar el paso hacia un futuro laboral mejor. Hazlo y matricúlate ya en este Grand Master que abrirá muchas puertas para tu carrera profesional"



Competencias específicas

- Entender los diferentes tipos de modelados de aplicaciones y patrones de diseño en el lenguaje unificado de modelamiento (UML)
- Comprender el funcionamiento de la gestión de la calidad en los proyectos, incluyendo la planificación, el aseguramiento, el control, los conceptos estadísticos y las herramientas disponibles
- Emplear los conocimientos necesarios para el desarrollo de aplicaciones e interfaces gráficas en los lenguajes *Java y .NET*
- Comprender los procedimientos y técnicas para mejor la apariencia de un documento escrito en HTML
- Dominar el proceso de interacciones con el cliente, mediante el uso de: formularios, *Cookies* y manejo de sesiones
- Comprender los mecanismos y métodos de autenticación para el control de acceso, así como el proceso de auditoría de accesos
- Comprender la aplicación de la seguridad, en las diferentes fases del ciclo de vida del software
- Conocer el concepto, funcionamiento, arquitectura, recursos y contenidos de un servidor web
- Comprender las diferentes herramientas de apoyo, metodologías y el análisis posterior durante la auditoría de seguridad en internet y en los dispositivos móviles
- Entender las políticas y estándares de la seguridad a aplicar en las aplicaciones online
- Ser capaz de redactar, planificar, desarrollar y firmar proyectos en el ámbito de la ingeniería en informática que tengan como objetivo el desarrollo o la explotación de sistemas, servicios y aplicaciones informáticas
- Dirigir las actividades de los proyectos informáticos
- Ser capaz de definir, evaluar y seleccionar plataformas hardware y software para el desarrollo y la ejecución de sistemas, servicios y aplicaciones informáticas
- Saber desarrollar usando las técnicas de *Scrum*, programación extrema y de desarrollo de software basado en reutilización
- Tener la capacidad para concebir, desarrollar y mantener sistemas, servicios y aplicaciones informáticas empleando los métodos de la ingeniería del software como instrumento para el aseguramiento de su calidad
- Emplear los fundamentos de la criptografía simétrica y de la criptografía asimétrica, así como sus principales algoritmos
- Aplicar los conceptos esenciales relacionados con los sistemas de información en la empresa, así como identificar las oportunidades y necesidades de los sistemas de información
- Saber desarrollar el cronograma para la gestión del tiempo, el presupuesto y la respuesta ante los riesgos
- Comprender el funcionamiento del gobierno y gestión de las TIC, las normas ISO/IEC que lo rigen y las buenas prácticas a llevar a cabo
- Planificar la gestión de la seguridad y a manejar los principales mecanismos para la protección de activos información

04

Estructura y contenido

El material didáctico de este Grand Master en Ingeniería de Software está elaborado para cubrir toda la enseñanza necesaria y complementaria en la materia, con las metodologías, herramientas y conocimientos más actualizados posibles del mercado. Un temario extenso y exhaustivo donde se imparten el uso de lenguajes de programación y entornos avanzados, administración de servidores web y su integración en el proceso de desarrollo o la gestión en sí de un proyecto. Esto asegura la mejor oportunidad posible para el alumno de especializarse en Ingeniería de Software y destacar rápidamente en ese campo.





Demuestra que tus conocimientos están a la par con tu actitud por ser el mejor profesional y añade un gran valor a tu currículo con este Grand Master en Ingeniería de Software"

Módulo 1. Metodologías, desarrollo y calidad en la Ingeniería de Software

1.1. Introducción a la Ingeniería de Software
- 1.1.1. Introducción
- 1.1.2. La crisis del software
- 1.1.3. Diferencias entre la Ingeniería de Software y la ciencia de la computación
- 1.1.4. Ética y responsabilidad profesional en la ingeniería del software
- 1.1.5. Fábricas de software

1.2. El proceso de desarrollo de software
- 1.2.1. Definición
- 1.2.2. Modelo de proceso software
- 1.2.3. El proceso unificado de desarrollo de software

1.3. Desarrollo de software orientado a objetos
- 1.3.1. Introducción
- 1.3.2. Principios de la orientación a objetos
- 1.3.3. Definición de objeto
- 1.3.4. Definición de clase
- 1.3.5. Análisis orientado a objetos vs. Diseño orientado a objetos

1.4. Desarrollo de software basado en modelos
- 1.4.1. La necesidad de modelar
- 1.4.2. Modelado de sistemas software
- 1.4.3. Modelado de objetos
- 1.4.4. UML
- 1.4.5. Herramientas CASE

1.5. Modelado de aplicaciones y patrones de diseño con UML
- 1.5.1. Modelado avanzado de requisitos
- 1.5.2. Modelado estático avanzado
- 1.5.3. Modelado dinámico avanzado
- 1.5.4. Modelado de componentes
- 1.5.5. Introducción a los patrones de diseño con UML
- 1.5.6. *Adapter*
- 1.5.7. *Factory*
- 1.5.8. *Singleton*
- 1.5.9. *Strategy*
- 1.5.10. *Composite*
- 1.5.11. *Facade*
- 1.5.12. *Observer*

1.6. Ingeniería dirigida por modelos
- 1.6.1. Introducción
- 1.6.2. Metamodelado de sistemas
- 1.6.3. MDA
- 1.6.4. DSL
- 1.6.5. Refinamientos de modelos con OCL
- 1.6.6. Transformaciones de modelos

1.7. Ontologías en la Ingeniería de Software
- 1.7.1. Introducción
- 1.7.2. Ingeniería de la ontología
- 1.7.3. Aplicación de las ontologías en la Ingeniería de Software

1.8. Metodologías ágiles para el desarrollo de software, *Scrum*
- 1.8.1. ¿Qué es la agilidad en el software?
- 1.8.2. El manifiesto ágil
- 1.8.3. La hoja de ruta de un proyecto ágil
- 1.8.4. El *Product Owner*
- 1.8.5. Las historias de usuario
- 1.8.6. Planificación y estimación ágil
- 1.8.7. Mediciones en desarrollos ágiles
- 1.8.8. Introducción al Scrum
- 1.8.9. Los roles
- 1.8.10. El *Product Backlog*
- 1.8.11. El *Sprint*
- 1.8.12. Las reuniones

1.9. La metodología de desarrollo de software *Lean*
- 1.9.1. Introducción
- 1.9.2. *Kanban*

1.10. Calidad y mejora del proceso software
- 1.10.1. Introducción
- 1.10.2. Medición del software
- 1.10.3. Pruebas del software
- 1.10.4. Modelo de calidad de procesos software: CMMI

Módulo 2. Gestión de proyectos de software

2.1. Conceptos fundamentales de la dirección de proyectos y el ciclo de vida de la gestión de proyectos
- 2.1.1. ¿Qué es un proyecto?
- 2.1.2. Metodología común
- 2.1.3. ¿Qué es la dirección/gestión de proyectos?
- 2.1.4. ¿Qué es un plan de proyecto?
- 2.1.5. Beneficios
- 2.1.6. Ciclo de vida del proyecto
- 2.1.7. Grupos de procesos o ciclo de vida de la gestión de los proyectos
- 2.1.8. La relación entre los grupos de procesos y las áreas de conocimiento
- 2.1.9. Relaciones entre el ciclo de vida del producto y del proyecto

2.2. El inicio y la planificación
- 2.2.1. De la idea al proyecto
- 2.2.2. Desarrollo del acta de proyecto
- 2.2.3. Reunión de arranque del proyecto
- 2.2.4. Tareas, conocimientos y habilidades en el proceso de inicio
- 2.2.5. El plan de proyecto
- 2.2.6. Desarrollo del plan básico. Pasos
- 2.2.7. Tareas, conocimientos y habilidades en el proceso de planificación

2.3. La gestión de los *Stakeholders* y del alcance
- 2.3.1. Identificar a los interesados
- 2.3.2. Desarrollar el plan para la gestión de los interesados
- 2.3.3. Gestionar el compromiso de los interesados
- 2.3.4. Controlar el compromiso de los interesados
- 2.3.5. El objetivo del proyecto
- 2.3.6. La gestión del alcance y su plan
- 2.3.7. Recopilar los requisitos
- 2.3.8. Definir el enunciado del alcance
- 2.3.9. Crear la WBS (EDT)
- 2.3.10. Verificar y controlar el alcance

2.4. El desarrollo del cronograma
- 2.4.1. La gestión del tiempo y su plan
- 2.4.2. Definir las actividades
- 2.4.3. Establecimiento de la secuencia de las actividades
- 2.4.4. Estimación de recursos de las actividades
- 2.4.5. Estimación de la duración de las actividades
- 2.4.6. Desarrollo del cronograma y cálculo del camino crítico
- 2.4.7. Control del cronograma

2.5. El desarrollo del presupuesto y la respuesta a los riesgos
- 2.5.1. Estimar los costes
- 2.5.2. Desarrollar el presupuesto y la curva S
- 2.5.3. Control de costes y método del valor ganado
- 2.5.4. Los conceptos de riesgo
- 2.5.5. ¿Cómo hacer un análisis de riesgos?
- 2.5.6. El desarrollo del plan de respuesta

2.6. La gestión de la calidad
- 2.6.1. Planificación de la calidad
- 2.6.2. Aseguramiento de la calidad
- 2.6.3. Control de la calidad
- 2.6.4. Conceptos estadísticos básicos
- 2.6.5. Herramientas de la gestión de la calidad

2.7. La comunicación y los recursos humanos
- 2.7.1. Planificar la gestión de las comunicaciones
- 2.7.2. Análisis de requisitos de comunicaciones
- 2.7.3. Tecnología de las comunicaciones
- 2.7.4. Modelos de comunicación
- 2.7.5. Métodos de comunicación
- 2.7.6. Plan de gestión de las comunicaciones

2.7.7. Gestionar las comunicaciones
2.7.8. La gestión de los recursos humanos
2.7.9. Principales actores y sus roles en los proyectos
2.7.10. Tipos de organizaciones
2.7.11. Organización del proyecto
2.7.12. El equipo de trabajo

2.8. El aprovisionamiento
2.8.1. El proceso de adquisiciones
2.8.2. Planificación
2.8.3. Búsqueda de suministradores y solicitud de ofertas
2.8.4. Adjudicación del contrato
2.8.5. Administración del contrato
2.8.6. Los contratos
2.8.7. Tipos de contratos
2.8.8. Negociación del contrato

2.9. Ejecución, monitorización y control y cierre
2.9.1. Los grupos de procesos
2.9.2. La ejecución del proyecto
2.9.3. La monitorización y control del proyecto
2.9.4. El cierre del proyecto

2.10. Responsabilidad profesional
2.10.1. Responsabilidad profesional
2.10.2. Características de la responsabilidad social y profesional
2.10.3. Código deontológico del líder de proyectos
2.10.4. Responsabilidad vs. PMP®
2.10.5. Ejemplos de responsabilidad
2.10.6. Beneficios de la profesionalización



Módulo 3. Plataformas de desarrollo del software

3.1. Introducción al desarrollo de aplicaciones
- 3.1.1. Aplicaciones de escritorio
- 3.1.2. Lenguaje de programación
- 3.1.3. Entornos de desarrollo integrado
- 3.1.4. Aplicaciones web
- 3.1.5. Aplicaciones móviles
- 3.1.6. Aplicaciones en la nube

3.2. Desarrollo de aplicaciones e interfaz gráfica en *Java*
- 3.2.1. Entornos de desarrollo integrados para *Java*
- 3.2.2. Principales IDE para *Java*
- 3.2.3. Introducción a la plataforma de desarrollo *Eclipse*
- 3.2.4. Introducción a la plataforma de desarrollo *NetBeans*
- 3.2.5. Modelo Vista Controlador para las interfaces gráficas de usuario
- 3.2.6. Diseñar una interfaz gráfica en *Eclipse*
- 3.2.7. Diseñar una interfaz gráfica en *NetBeans*

3.3. Depuración y pruebas en *Java*
- 3.3.1. Pruebas y depuración de programas en *Java*
- 3.3.2. Depuración en *Eclipse*
- 3.3.3. Depuración en *NetBeans*

3.4. Desarrollo de aplicaciones e interfaz gráfica en .NET
- 3.4.1. *Net Framework*
- 3.4.2. Componentes de la plataforma de desarrollo .NET
- 3.4.3. Visual Studio .NET
- 3.4.4. Herramientas de .NET para GUI
- 3.4.5. La GUI con *Windows Presentation Foundation*
- 3.4.6. Depurar y compilar una aplicación de WPF

3.5. Programación para redes .NET
- 3.5.1. Introducción a la programación para redes en .NET
- 3.5.2. Peticiones y respuestas en .NET
- 3.5.3. Uso de protocolos de aplicación en .NET
- 3.5.4. Seguridad en la programación para redes en .NET

3.6. Entornos de desarrollo de aplicaciones móviles
- 3.6.1. Aplicaciones móviles
- 3.6.2. Aplicaciones móviles *Android*
- 3.6.3. Pasos para el desarrollo en *Android*
- 3.6.4. El IDE *Android Studio*

3.7. Desarrollo de aplicaciones en el entorno *Android Studio*
- 3.7.1. Instalar e iniciar Android Studio
- 3.7.2. Ejecución de una aplicación *Android*
- 3.7.3. Desarrollo de la interfaz gráfica en *Android Studio*
- 3.7.4. Iniciando actividades en *Android Studio*

3.8. Depuración y publicación de aplicaciones *Android*
- 3.8.1. Depuración de una aplicación en *Android Studio*
- 3.8.2. Memorizar aplicaciones en *Android Studio*
- 3.8.3. Publicación de una aplicación en *Google Play*

3.9. Desarrollo de aplicaciones para la nube
- 3.9.1. *Cloud computing*
- 3.9.2. Niveles de *Cloud*: SaaS, PaaS, IaaS
- 3.9.3. Principales plataformas de desarrollo en la nube
- 3.9.4. Referencias bibliográficas

3.10. Introducción a Google *Cloud Platform*
- 3.10.1. Conceptos básicos de Google *Cloud Platform*
- 3.10.2. Servicios de Google *Cloud Platform*
- 3.10.3. Herramientas de Google *Cloud Platform*

Módulo 4. Computación en el cliente web

4.1. Introducción a HTML
- 4.1.1. Estructura de un documento
- 4.1.2. Color
- 4.1.3. Texto
- 4.1.4. Enlaces de hipertexto
- 4.1.5. Imágenes
- 4.1.6. Listas
- 4.1.7. Tablas
- 4.1.8. Marcos (*Frames*)
- 4.1.9. Formularios
- 4.1.10. Elementos específicos para tecnologías móviles
- 4.1.11. Elementos en desuso

4.2. Hojas de estilo web (CSS)
- 4.2.1. Elementos y estructura de una hoja de estilos
  - 4.2.1.1. Creación de hojas de estilo
  - 4.2.1.2. Aplicación de estilos. Selectores
  - 4.2.1.3. Herencia de estilos y aplicación en cascada
  - 4.2.1.4. Formateado de páginas mediante estilos
  - 4.2.1.5. Estructura de páginas mediante estilos. El modelo de cajas
- 4.2.2. Diseño de estilos para diferentes dispositivos
- 4.2.3. Tipos de hojas de estilos: estáticas y dinámicas. Las pseudo-clases
- 4.2.4. Buenas prácticas en el uso de hojas de estilo

4.3. Introducción e historia de *JavaScript*
- 4.3.1. Introducción
- 4.3.2. Historia de *JavaScript*
- 4.3.3. Entorno de desarrollo que vamos a usar

4.4. Nociones básicas de programación web
- 4.4.1. Sintaxis básica de *JavaScript*
- 4.4.2. Tipos de datos primitivos y operadores
- 4.4.3. Variables y ámbitos
- 4.4.4. Cadenas de texto y *Template Literals*
- 4.4.5. Números y booleanos
- 4.4.6. Comparaciones

4.5. Estructuras complejas en *JavaScript*
- 4.5.1. Vectores o *Arrays* y objetos
- 4.5.2. Conjuntos
- 4.5.3. Mapas
- 4.5.4. Disyuntivas
- 4.5.5. Bucles

4.6. Funciones y objetos
- 4.6.1. Definición e invocación de funciones
- 4.6.2. Argumentos
- 4.6.3. Funciones flecha
- 4.6.4. Funciones de retrollamada o *Callback*
- 4.6.5. Funciones de orden superior
- 4.6.6. Objetos literales
- 4.6.7. El objeto *This*
- 4.6.8. Objetos como espacios de nombres: el objeto *Math* y el objeto *Date*

4.7. El modelo de objetos del documento (DOM)
- 4.7.1. ¿Qué es el DOM?
- 4.7.2. Un poco de historia
- 4.7.3. Navegación y obtención de elementos
- 4.7.4. Un DOM virtual con JSDOM
- 4.7.5. Selectores de consulta o *Query Selectors*
- 4.7.6. Navegación mediante propiedades
- 4.7.7. Asignación de atributos a los elementos
- 4.7.8. Creación y modificación de nodos
- 4.7.9. Actualización del estilo de los elementos del DOM

4.8. Desarrollo web moderno
- 4.8.1. Flujo basado en eventos y *Listeners*
- 4.8.2. *Toolkits* web modernos y sistemas de alineamiento
- 4.8.3. Modo estricto de *JavaScript*
- 4.8.4. Algo más sobre funciones
- 4.8.5. Promesas y funciones asíncronas
- 4.8.6. *Closures*
- 4.8.7. Programación funcional
- 4.8.8. POO en *JavaScript*

4.9. Usabilidad web
- 4.9.1. Introducción a la usabilidad
- 4.9.2. Definición de usabilidad
- 4.9.3. Importancia del diseño web centrado en el usuario
- 4.9.4. Diferencias entre accesibilidad y usabilidad
- 4.9.5. Ventajas y problemas en la combinación de accesibilidad y usabilidad
- 4.9.6. Ventajas y dificultades en la implantación de sitios web usables
- 4.9.7. Métodos de usabilidad
- 4.9.8. Análisis de requerimiento de usuario
- 4.9.9. Principios del diseño conceptual. Creación de prototipos orientados al usuario
- 4.9.10. Pautas para la creación de sitios web usables
  - 4.9.10.1. Pautas de usabilidad de *Jakob Nielsen*
  - 4.9.10.2. Pautas de usabilidad de Bruce Tognazzini
- 4.9.11. Evaluación de la usabilidad

4.10. Accesibilidad web
- 4.10.1. Introducción
- 4.10.2. Definición de accesibilidad web
- 4.10.3. Tipos de discapacidades
  - 4.10.3.1. Discapacidades temporales o permanentes
  - 4.10.3.2. Discapacidades visuales
  - 4.10.3.3. Discapacidades auditivas
  - 4.10.3.4. Discapacidades motrices
  - 4.10.3.5. Discapacidad neurológicas o cognitivas
  - 4.10.3.6. Dificultades derivadas del envejecimiento
  - 4.10.3.7. Limitaciones derivadas del entorno
  - 4.10.3.8. Barreras que impiden el acceso a la web
- 4.10.4. Ayudas técnicas y productos de apoyo para superar las barreras
  - 4.10.4.1. Ayudas para personas ciegas
  - 4.10.4.2. Ayudas para persona con baja visión
  - 4.10.4.3. Ayudas para personas con daltonismo
  - 4.10.4.4. Ayudas para personas con discapacidad auditiva
  - 4.10.4.5. Ayudas para personas con discapacidad motriz
  - 4.10.4.6. Ayudas para personas con discapacidad cognitiva y neurológica
- 4.10.5. Ventajas y dificultades en la implantación de la accesibilidad web
- 4.10.6. Normativa y estándares sobre accesibilidad web
- 4.10.7. Organismos regulatorios de la accesibilidad web
- 4.10.8. Comparativa de normas y estándares
- 4.10.9. Guías para el cumplimiento de normativas y estándares
  - 4.10.9.1. Descripción de las pautas principales (imágenes, enlaces videos, etc.)
  - 4.10.9.2. Pautas para una navegación accesible
    - 4.10.9.2.1. Perceptibilidad
    - 4.10.9.2.2. Operatividad
    - 4.10.9.2.3. Comprensibilidad
    - 4.10.9.2.4. Robustez
- 4.10.10. Descripción del proceso de la conformidad en accesibilidad web
- 4.10.11. Niveles de conformidad
- 4.10.12. Criterios de conformidad
- 4.10.13. Requisitos de conformidad
- 4.10.14. Metodología de evaluación de la accesibilidad en sitios web

Módulo 5. Computación en servidor web

5.1. Introducción a la programación en el servidor: PHP
- 5.1.1. Conceptos básicos de programación en el servidor
- 5.1.2. Sintaxis básica de PHP
- 5.1.3. Generación de contenido HTML con PHP
- 5.1.4. Entornos de desarrollo y pruebas: XAMPP

5.2. PHP avanzado
- 5.2.1. Estructuras de control con PHP
- 5.2.2. Funciones en PHP
- 5.2.3. Manejo de *Arrays* en PHP
- 5.2.4. Manejo de cadenas con PHP
- 5.2.5. Orientación a objetos en PHP

5.3. Modelos de datos
- 5.3.1. Concepto de dato. Ciclo de vida de los datos
- 5.3.2. Tipos de datos
  - 5.3.2.1. Básicos
  - 5.3.2.2. Registros
  - 5.3.2.3. Dinámicos

5.4. El modelo relacional
- 5.4.1. Descripción
- 5.4.2. Entidades y tipos de entidades
- 5.4.3. Elementos de datos. Atributos
- 5.4.4. Relaciones: tipos, subtipos, cardinalidad
- 5.4.5. Claves. Tipos de claves
- 5.4.6. Normalización. Formas normales

5.5. Construcción del modelo lógico de datos
- 5.5.1. Especificación de tablas
- 5.5.2. Definición de columnas
- 5.5.3. Especificación de claves
- 5.5.4. Conversión a formas normales. Dependencias

5.6. El modelo físico de datos. Ficheros de datos
- 5.6.1. Descripción de los ficheros de datos
- 5.6.2. Tipos de ficheros
- 5.6.3. Modos de acceso
- 5.6.4. Organización de ficheros

5.7. Acceso a bases de datos desde PHP
- 5.7.1. Introducción a *MariaDB*
- 5.7.2. Trabajar con una base de datos *MariaDB*: el lenguaje SQL
- 5.7.3. Acceder a la base de datos *MariaDB* desde PHP
- 5.7.4. Introducción a MySql
- 5.7.5. Trabajar con una base de datos MySql: el lenguaje SQL
- 5.7.6. Acceder a la base de datos MySql desde PHP

5.8. Interacción con el cliente desde PHP
- 5.8.1. Formularios PHP
- 5.8.2. *Cookies*
- 5.8.3. Manejo de sesiones

5.9. Arquitectura de aplicaciones web
- 5.9.1. El patrón Modelo Vista Controlador
- 5.9.2. Controlador
- 5.9.3. Modelo
- 5.9.4. Vista

5.10. Introducción a los servicios web
- 5.10.1. Introducción a XML
- 5.10.2. Arquitecturas orientas a servicios (SOA): servicios web
- 5.10.3. Creación de servicios web SOAP y REST
- 5.10.4. El protocolo SOAP
- 5.10.5. El protocolo REST

Módulo 6. Gestión de la seguridad

6.1. La seguridad de la información
- 6.1.1. Introducción
- 6.1.2. La seguridad de la información implica la confidencialidad, integridad y disponibilidad
- 6.1.3. La seguridad es un asunto económico
- 6.1.4. La seguridad es un proceso
- 6.1.5. La clasificación de la información
- 6.1.6. La seguridad en la información implica la gestión de los riesgos
- 6.1.7. La seguridad se articula con controles de seguridad
- 6.1.8. La seguridad es tanto física como lógica
- 6.1.9. La seguridad implica a las personas

6.2. El profesional de la seguridad de la información
- 6.2.1. Introducción
- 6.2.2. La seguridad de la información como profesión
- 6.2.3. Las certificaciones (ISC)2
- 6.2.4. El estándar ISO 27001
- 6.2.5. Buenas prácticas de seguridad en la gestión de servicios TI
- 6.2.6. Modelos de madurez para la seguridad de la información
- 6.2.7. Otras certificaciones, estándares y recursos profesionales

6.3. Control de accesos
- 6.3.1. Introducción
- 6.3.2. Requisitos del control de accesos
- 6.3.3. Mecanismos de autenticación
- 6.3.4. Métodos de autorización
- 6.3.5. Contabilidad y auditoría de accesos
- 6.3.6. Tecnologías «Triple A»

6.4. Programas, procesos y políticas de seguridad de la información
- 6.4.1. Introducción
- 6.4.2. Programas de gestión de la seguridad
- 6.4.3. La gestión de riesgos
- 6.4.4. Diseño de políticas de seguridad

6.5. Planes de continuidad de negocio
- 6.5.1. Introducción a los PCN
- 6.5.2. Fase I y II
- 6.5.3. Fase III y IV
- 6.5.4. Mantenimiento del PCN

6.6. Procedimientos para a correcta protección de la empresa
- 6.6.1. Redes DMZ
- 6.6.2. Sistemas de detección de intrusos
- 6.6.3. Listas de control de accesos
- 6.6.4. Aprender del atacante: *Honeypot*

6.7. Arquitectura de seguridad. Prevención
- 6.7.1. Visión general. Actividades y modelo de capas
- 6.7.2. Defensa perimetral (*Firewalls, WAFs*, IPS, etc.)
- 6.7.3. Defensa del punto final (equipos, servidores y servicios)

6.8. Arquitectura de seguridad. Detección
- 6.8.1. Visión general detección y supervisión
- 6.8.2. Logs, ruptura de tráfico cifrado, grabación y Siems
- 6.8.3. Alertas e inteligencia

6.9. Arquitectura de seguridad. Reacción
- 6.9.1. Reacción. Productos, servicios y recursos
- 6.9.2. Gestión de incidentes
- 6.9.3. CERTS y CSIRTs

6.10. Arquitectura de seguridad. Recuperación
- 6.10.1. Resiliencia, conceptos, requerimientos de negocio y normativa
- 6.10.2. Soluciones IT de resiliencia
- 6.10.3. Gestión y gobierno de las crisis

Módulo 7. Seguridad en los sistemas de información

7.1. Una perspectiva global de la seguridad, la criptografía y los criptoanálisis clásicos
- 7.1.1. La seguridad informática: perspectiva histórica
- 7.1.2. Pero, ¿qué se entiende exactamente por seguridad?
- 7.1.3. Historia de la criptografía
- 7.1.4. Cifradores de sustitución
- 7.1.5. Caso de estudio: la máquina Enigma

7.2. Criptografía simétrica
- 7.2.1. Introducción y terminología básica
- 7.2.2. Cifrado simétrico
- 7.2.3. Modos de operación
- 7.2.4. DES
- 7.2.5. El nuevo estándar AES
- 7.2.6. Cifrado en flujo
- 7.2.7. Criptoanálisis

7.3. Criptografía asimétrica
- 7.3.1. Orígenes de la criptografía de clave pública
- 7.3.2. Conceptos básicos y funcionamiento
- 7.3.3. El algoritmo RSA
- 7.3.4. Certificados digitales
- 7.3.5. Almacenamiento y gestión de claves

7.4. Ataques en redes
- 7.4.1. Amenazas y ataques de una red
- 7.4.2. Enumeración
- 7.4.3. Interceptación de tráfico: *Sniffers*
- 7.4.4. Ataques de denegación de servicio
- 7.4.5. Ataques de envenenamiento ARP

7.5. Arquitecturas de seguridad
- 7.5.1. Arquitecturas de seguridad tradicionales
- 7.5.2. *Secure Socket Layer*: SSL
- 7.5.3. Protocolo SSH
- 7.5.4. Redes Privadas Virtuales (VPNs)
- 7.5.5. Mecanismos de protección de unidades de almacenamiento externo
- 7.5.6. Mecanismos de protección hardware

7.6. Técnicas de protección de sistemas y desarrollo de código seguro
- 7.6.1. Seguridad en operaciones
- 7.6.2. Recursos y controles
- 7.6.3. Monitorización
- 7.6.4. Sistemas de detección de intrusión
- 7.6.5. IDS de *Host*
- 7.6.6. IDS de red
- 7.6.7. IDS basados en firmas
- 7.6.8. Sistemas señuelos
- 7.6.9. Principios de seguridad básicos en el desarrollo de código
- 7.6.10. Gestión del fallo
- 7.6.11. Enemigo público número 1: el desbordamiento de búfer
- 7.6.12. Chapuzas criptográficas

7.7. *Botnets y Spam*
- 7.7.1. Origen del problema
- 7.7.2. Proceso del spam
- 7.7.3. Envío del spam
- 7.7.4. Refinamiento de las listas de direcciones de correo
- 7.7.5. Técnicas de protección
- 7.7.6. Servicio anti-spam ofrecidos por terceros
- 7.7.7. Casos de estudio
- 7.7.8. Spam exótico

7.8. Auditoría y ataques Web
- 7.8.1. Recopilación de información
- 7.8.2. Técnicas de ataque
- 7.8.3. Herramientas

7.9. Malware y código malicioso
- 7.9.1. ¿Qué es el malware?
- 7.9.2. Tipos de malware
- 7.9.3. Virus
- 7.9.4. Criptovirus
- 7.9.5. Gusanos
- 7.9.6. Adware
- 7.9.7. *Spyware*
- 7.9.8. *Hoaxes*
- 7.9.9. *Pishing*
- 7.9.10. Troyanos
- 7.9.11. La economía del malware
- 7.9.12. Posibles soluciones

7.10. Análisis forense
- 7.10.1. Recolección de evidencias
- 7.10.2. Análisis de las evidencias
- 7.10.3. Técnicas anti-forenses
- 7.10.4. Caso de estudio práctico

Módulo 8. Seguridad en el software

8.1. Problemas de la seguridad en el software
- 8.1.1. Introducción al problema de la seguridad en el software
- 8.1.2. Vulnerabilidades y su clasificación
- 8.1.3. Propiedades software seguro
- 8.1.4. Referencias

8.2. Principios de diseño seguridad del software
- 8.2.1. Introducción
- 8.2.2. Principios de diseño seguridad del software
- 8.2.3. Tipos de S-SDLC
- 8.2.4. Seguridad del software en las fases del S-SDLC
- 8.2.5. Metodologías y estándares
- 8.2.6. Referencias

8.3. Seguridad en el ciclo de vida del software en las fases de requisitos y diseño
- 8.3.1. Introducción
- 8.3.2. Modelado de ataques
- 8.3.3. Casos de abuso
- 8.3.4. Ingeniería de requisitos de seguridad
- 8.3.5. Análisis de riesgo. Arquitectónico
- 8.3.6. Patrones de diseño
- 8.3.7. Referencias

8.4. Seguridad en el ciclo de vida del software en las fases de codificación, pruebas y operación
- 8.4.1. Introducción
- 8.4.2. Pruebas de seguridad basadas en riesgo
- 8.4.3. Revisión de código
- 8.4.4. Test de penetración
- 8.4.5. Operaciones de seguridad
- 8.4.6. Revisión externa
- 8.4.7. Referencias

8.5. Codificación segura aplicaciones I
- 8.5.1. Introducción
- 8.5.2. Prácticas de codificación segura
- 8.5.3. Manipulación y validación de entradas
- 8.5.4. Desbordamiento de memoria
- 8.5.5. Referencias

8.6. Codificación segura aplicaciones II
- 8.6.1. Introducción
- 8.6.2. *Integers Overflows*, errores de truncado y problemas con conversiones de tipo entre números enteros
- 8.6.3. Errores y excepciones
- 8.6.4. Privacidad y confidencialidad
- 8.6.5. Programas privilegiados
- 8.6.6. Referencias

8.7. Seguridad en el desarrollo y en la nube
- 8.7.1. Seguridad en el desarrollo; metodología y práctica
- 8.7.2. Modelos PaaS, IaaS,CaaS y SaaS
- 8.7.3. Seguridad en la nube y para servicios en la nube

8.8. Automatización y orquestación de seguridad (SOAR)
- 8.9.1. Complejidad del tratamiento manual; necesidad de automatizar las tareas
- 8.9.2. Productos y servicios
- 8.9.3. Arquitectura SOAR

8.9. Seguridad en el teletrabajo
- 8.9.1. Necesidad y escenarios
- 8.9.2. Productos y servicios
- 8.9.3. Seguridad en el teletrabajo

Módulo 9. Calidad y auditoría de sistemas de información

9.1. Introducción a los Sistemas de Gestión de Seguridad de la Información
- 9.1.1. Principios fundamentales de los SGSI
- 9.1.2. Reglas de oro de los SGSI
- 9.1.3. Papel de la auditoría informática en los SGSI

9.2. Planificación en la gestión de la seguridad
- 9.2.1. Conceptos relativos a la gestión de la seguridad
- 9.2.2. Clasificación de la información: objetivos, conceptos y roles
- 9.2.3. Implementación de las políticas de seguridad: políticas de seguridad, estándares y procedimientos
- 9.2.4. Gestión del riesgo: principios y análisis del riesgo de los activos de información

9.3. Principales mecanismos para la protección de activos información (I)
- 9.3.1. Resumen de las principales herramientas criptográficas para la protección de la triada CID
- 9.3.2. Consideración de los requisitos de privacidad, anonimato y gestión adecuada de la trazabilidad de usuarios

9.4. Principales mecanismos para la protección de activos información (II)
- 9.4.1. Seguridad de las comunicaciones: protocolos, dispositivos y arquitecturas de seguridad
- 9.4.2. Seguridad de los sistemas operativos

9.5. Controles internos de los SGSI
- 9.5.1. Taxonomía de los controles SGSI: controles administrativos, lógicos y físicos
- 9.5.2. Clasificación de los controles en función del modo de abordar la amenaza: controles para la prevención, la detección y la corrección de amenazas
- 9.5.3. Implantación de sistemas de control interno en los SGSI

9.6. Tipos de auditoría
- 9.6.1. Diferencia entre auditoría y control interno
- 9.6.2. Auditoría interna frente a auditoría externa
- 9.6.3. Clasificación de la auditoría en función del objetivo y el tipo de análisis

9.7. Guionista y guion: sujeto y objeto protegido por la propiedad intelectual
- 9.7.1. Introducción a los test de penetración y al análisis forense
- 9.7.2. Definición y relevancia de los conceptos de *Fingerprinting* y *Footprinting*

9.8. Análisis de vulnerabilidades y monitorización de tráfico de red
- 9.8.1. Herramientas para el análisis de vulnerabilidades en sistemas
- 9.8.2. Principales vulnerabilidades en el contexto de las aplicaciones web
- 9.8.3. Análisis de protocolos de comunicaciones

9.9. El proceso de la auditoría informática
- 9.9.1. Concepto de ciclo de vida en el desarrollo de sistemas
- 9.9.2. Monitorización de actividad y de procesos: recolección y tratamiento de evidencias
- 9.9.3. Metodología de la auditoría informática
- 9.9.4. Proceso de una auditoría informática
- 9.9.5. Identificación de los principales delitos y faltas en el contexto de las tecnologías de la información
- 9.9.6. Investigación de delitos informáticos: introducción al análisis forense y su relación con la auditoría informática

9.10. Planes de continuidad de negocio y de recuperación frente a desastres
- 9.10.1. Definición de plan de continuidad de negocio y del concepto de interrupción del negocio
- 9.10.2. Recomendación NIST sobre los planes de continuidad de negocio
- 9.10.3. Plan de recuperación ante desastres
- 9.10.4. Proceso de plan de recuperación ante desastres

Módulo 10. Administración de servidores web

10.1. Introducción a servidores web
- 10.1.1. ¿Que es un servidor web?
- 10.1.2. Arquitectura y funcionamiento de un servidor web
- 10.1.3. Recursos y contenidos en un servidor web
- 10.1.4. Servidores de aplicaciones
- 10.1.5. Servidores Proxy
- 10.1.6. Principales servidores web del mercado
- 10.1.7. Estadística de uso servidores web
- 10.1.8. Seguridad en servidores web
- 10.1.9. Balanceo de carga en servidores web
- 10.1.10. Referencias

10.2. Manejo del protocolo HTTP
- 10.2.1. Funcionamiento y estructura
- 10.2.2. Descripción de peticiones o *Request Methods*
- 10.2.3. Códigos de estado
- 10.2.4. Cabeceras
- 10.2.5. Codificación del contenido. Páginas de códigos
- 10.2.6. Realización de peticiones HTTP en Internet mediante un proxy, *Livehttpheaders* o método similar, analizando el protocolo utilizado

10.3. Descripción de arquitecturas distribuidas en múltiples servidores
- 10.3.1. Modelo de 3 capas
- 10.3.2. Tolerancia a fallos
- 10.3.3. Reparto de carga
- 10.3.4. Almacenes de estado de sesión
- 10.3.5. Almacenes de caché

10.4. *Internet Information Services* (IIS)
- 10.4.1. ¿Que es IIS?
- 10.4.2. Historia y evolución de IIS
- 10.4.3. Principales ventajas y características de IIS y posteriores
- 10.4.4. Arquitectura IIS y posteriores

10.5. Instalación, administración y configuración de IIS
- 10.5.1. Preámbulo
- 10.5.2. Instalación de *Internet Information Services* (IIS)
- 10.5.3. Herramientas de administración de IIS
- 10.5.4. Creación, configuración y administración de sitios web
- 10.5.5. Instalación y manejo de extensiones en IIS

10.6. Seguridad avanzada en IIS
- 10.6.1. Preámbulo
- 10.6.2. Autentificación, autorización, y control de acceso en IIS
- 10.6.3. Configuración de un sitio web seguro en IIS con SSL
- 10.6.4. Políticas de seguridad implementada en IIS 10.x

10.7. Introducción a Apache
- 10.7.1. ¿Qué es Apache?
- 10.7.2. Principales ventajas de Apache
- 10.7.3. Características principales de Apache
- 10.7.4. Arquitectura

10.8. Instalación y configuración de Apache
- 10.8.1. Instalación inicial de Apache
- 10.8.2. Configuración de Apache

10.9. Instalación y configuración de los diferentes módulos en Apache
- 10.9.1. Instalación de módulos en Apache
- 10.9.2. Tipos de módulos
- 10.9.3. Configuración segura de Apache

10.10. Seguridad avanzada
- 10.10.1. Autenticación, autorización y control de acceso
- 10.10.2. Métodos de autenticación
- 10.10.3. Configuración segura de Apache con SSL

Módulo 11. Seguridad en aplicaciones online

11.1. Vulnerabilidades y problemas de seguridad en las aplicaciones online
- 11.1.1. Introducción a la seguridad en las aplicaciones online
- 11.1.2. Vulnerabilidades de seguridad en el diseño de las aplicaciones web
- 11.1.3. Vulnerabilidades de seguridad en la implementación de las aplicaciones web
- 11.1.4. Vulnerabilidades de seguridad en el despliegue de las aplicaciones web
- 11.1.5. Listas oficiales de vulnerabilidades de seguridad

11.2. Políticas y estándares para la seguridad de las aplicaciones online
- 11.2.1. Pilares para la seguridad de las aplicaciones online
- 11.2.2. Política de seguridad
- 11.2.3. Sistema de gestión de seguridad de la información
- 11.2.4. Ciclo de vida de desarrollo seguro de software
- 11.2.5. Estándares para la seguridad de las aplicaciones

11.3. Seguridad en el diseño de las aplicaciones web
- 11.3.1. Introducción a la seguridad de las aplicaciones web
- 11.3.2. Seguridad en el diseño de las aplicaciones web

11.4. Test de la seguridad y protección online de las aplicaciones web
- 11.4.1. Análisis y test de la seguridad de las aplicaciones web
- 11.4.2. Seguridad en el despliegue y producción de las aplicaciones web

11.5. Seguridad de los servicios web
- 11.5.1. Introducción a la seguridad de los servicios web
- 11.5.2. Funciones y tecnologías de la seguridad de los servicios web

11.6. Test de la seguridad y protección online de los servicios web
- 11.6.1. Evaluación de la seguridad de los servicios web
- 11.6.2. Protección online. *Firewalls* y *Gateways* XML

11.7. *Hacking* ético, malware y *Forensic*
- 11.7.1. *Hacking* ético
- 11.7.2. Análisis de Malware
- 11.7.3. Análisis forense

11.8. Resolución de incidentes sobre servicios web
- 11.8.1. Monitorización
- 11.8.2. Herramientas de medición del rendimiento
- 11.8.3. Medidas de contención
- 11.8.4. Análisis causa–raíz
- 11.8.5. Gestión proactiva de problemas

11.9. Buenas prácticas para garantizar la seguridad en las aplicaciones
- 11.9.1. Manual de buenas prácticas en el desarrollo de las aplicaciones online
- 11.9.2. Manual de buenas prácticas en la implementación de las aplicaciones online

11.10. Errores comunes que perjudican la seguridad de las aplicaciones
- 11.10.1. Errores comunes en el desarrollo
- 11.10.2. Errores comunes en el hospedaje
- 11.10.3. Errores comunes en la producción

Módulo 12. Ingeniería del software

12.1. Introducción a la ingeniería del software y al modelado
- 12.1.1. La naturaleza del software
- 12.1.2. La naturaleza única de las *Webapps*
- 12.1.3. Ingeniería del software
- 12.1.4. El proceso del software
- 12.1.5. La práctica de la ingeniería del software
- 12.1.6. Mitos del software
- 12.1.7. ¿Cómo comienza todo?
- 12.1.8. Conceptos orientados a objetos
- 12.1.9. Introducción a UML

12.2. El proceso del software
- 12.2.1. Un modelo general de proceso
- 12.2.2. Modelos de proceso prescriptivos
- 12.2.3. Modelos de proceso especializado
- 12.2.4. El proceso unificado
- 12.2.5. Modelos del proceso personal y del equipo
- 12.2.6. ¿Qué es la agilidad?
- 12.2.7. ¿Qué es un proceso ágil?
- 12.2.8. *Scrum*
- 12.2.9. Conjunto de herramientas para el proceso ágil

12.3. Principios que guían la práctica de la ingeniería del software
- 12.3.1. Principios que guían el proceso
- 12.3.2. Principios que guían la práctica
- 12.3.3. Principios de comunicación
- 12.3.4. Principios de planificación
- 12.3.5. Principios de modelado
- 12.3.6. Principios de construcción
- 12.3.7. Principios de despliegue

12.4. Comprensión de los requisitos
- 12.4.1. Ingeniería de requisitos
- 12.4.2. Establecer las bases
- 12.4.3. Indagación de los requisitos
- 12.4.4. Desarrollo de casos de uso
- 12.4.5. Elaboración del modelo de los requisitos
- 12.4.6. Negociación de los requisitos
- 12.4.7. Validación de los requisitos

12.5. Modelado de los requisitos: escenarios, información y clases de análisis
- 12.5.1. Análisis de los requisitos
- 12.5.2. Modelado basado en escenarios
- 12.5.3. Modelos UML que proporcionan el caso de uso
- 12.5.4. Conceptos de modelado de datos
- 12.5.5. Modelado basado en clases
- 12.5.6. Diagramas de clases

12.6. Modelado de los requisitos: flujo, comportamiento y patrones
- 12.6.1. Requisitos que modelan las estrategias
- 12.6.2. Modelado orientado al flujo
- 12.6.3. Diagramas de estado
- 12.6.4. Creación de un modelo de comportamiento
- 12.6.5. Diagramas de secuencia
- 12.6.6. Diagramas de comunicación
- 12.6.7. Patrones para el modelado de requisitos

12.7. Conceptos de diseño
- 12.7.1. Diseño en el contexto de la ingeniería del software
- 12.7.2. El proceso de diseño
- 12.7.3. Conceptos de diseño
- 12.7.4. Conceptos de diseño orientado a objetos
- 12.7.5. El modelo del diseño

12.8. Diseño de la arquitectura
- 12.8.1. Arquitectura del software
- 12.8.2. Géneros arquitectónicos
- 12.8.3. Estilos arquitectónicos
- 12.8.4. Diseño arquitectónico
- 12.8.5. Evolución de los diseños alternativos para la arquitectura
- 12.8.6. Mapeo de la arquitectura con el uso del flujo de datos

12.9. Diseño en el nivel de componentes y basado en patrones
- 12.9.1. ¿Qué es un componente?
- 12.9.2. Diseño de componentes basados en clase
- 12.9.3. Realización del diseño en el nivel de componentes
- 12.9.4. Diseño de componentes tradicionales
- 12.9.5. Desarrollo basado en componentes
- 12.9.6. Patrones de diseño
- 12.9.7. Diseño de software basado en patrones
- 12.9.8. Patrones arquitectónicos
- 12.9.9. Patrones de diseño en el nivel de componentes
- 12.9.10. Patrones de diseño de la interfaz de usuario

12.10. Calidad del software y administración de proyectos
- 12.10.1. Calidad
- 12.10.2. Calidad del software
- 12.10.3. El dilema de la calidad del software
- 12.10.4. Lograr la calidad del software
- 12.10.5. Aseguramiento de la calidad del software
- 12.10.6. El espectro administrativo
- 12.10.7. El personal
- 12.10.8. El producto
- 12.10.9. El proceso
- 12.10.10. El proyecto
- 12.10.11. Principios y prácticas

Módulo 13. Ingeniería del software avanzada

13.1. Introducción a las metodologías ágiles
- 13.1.1. Modelos de proceso y metodologías
- 13.1.2. Agilidad y procesos ágiles
- 13.1.3. Manifiesto ágil
- 13.1.4. Algunas metodologías ágiles
- 13.1.5. Ágil vs. Tradicional

13.2. Scrum
- 13.2.1. Orígenes y filosofía de Scrum
- 13.2.2. Valores de Scrum
- 13.2.3. Flujo del proceso Scrum
- 13.2.4. Los roles de Scrum
- 13.2.5. Los artefactos de Scrum
- 13.2.6. Los eventos de Scrum
- 13.2.7. Las historias de usuario
- 13.2.8. Extensiones de Scrum
- 13.2.9. Estimaciones ágiles
- 13.2.10. Escalado de Scrum

13.3. Programación extrema
- 13.3.1. Justificación y visión general de XP
- 13.3.2. El ciclo de vida en XP
- 13.3.3. Los cinco valores básicos
- 13.3.4. Las doce prácticas básicas en XP
- 13.3.5. Roles de los participantes
- 13.3.6. XP Industrial
- 13.3.7. Valoración crítica de XP

13.4. Desarrollo de software basado en reutilización
- 13.4.1. La reutilización del software
- 13.4.2. Niveles de reutilización de código
- 13.4.3. Técnicas concretas de reutilización
- 13.4.4. Desarrollo basado en componentes
- 13.4.5. Beneficios y problemas de la reutilización
- 13.4.6. Planificación de la reutilización

13.5. Patrones de arquitectura de sistemas y de diseño de software
- 13.5.1. El diseño arquitectónico
- 13.5.2. Patrones arquitectónicos generales
- 13.5.3. Arquitecturas tolerantes a fallos
- 13.5.4. Arquitecturas de sistemas distribuidos
- 13.5.5. Los patrones de diseño
- 13.5.6. Patrones de Gamma
- 13.5.7. Patrones de diseño de interacción

13.6. Arquitectura de aplicaciones en la nube
- 13.6.1. Fundamentos de *Cloud Computing*
- 13.6.2. Calidad de las aplicaciones en la nube
- 13.6.3. Estilos de arquitectura
- 13.6.4. Patrones de diseño

13.7. Pruebas del software: TDD, ATDD y BDD
- 13.7.1. Verificación y validación del software
- 13.7.2. Las pruebas de software
- 13.7.3. *Test Driven Development* (TDD)
- 13.7.4. *Acceptance Test Driven Development* (ATDD)
- 13.7.5. *Behavior Driven Development* (BDD)
- 13.7.6. BDD y Cucumber

13.8. La mejora del proceso de software
- 13.8.1. La mejora del proceso de software
- 13.8.2. El proceso de mejora de procesos
- 13.8.3. Modelos de madurez
- 13.8.4. El modelo CMMI
- 13.8.5. CMMI V13.0
- 13.8.6. CMMI y Ágil

13.9. La calidad del producto software: *SQuaRE*
- 13.9.1. La calidad del software
- 13.9.2. Modelos de calidad del producto software
- 13.9.3. Familia ISO/IEC 13.000
- 13.9.4. ISO/IEC 13.010: modelo y características de calidad
- 13.9.5. ISO/IEC 13.0113. la calidad de los datos
- 13.9.6. ISO/IEC 13.013.: medición de la calidad del software
- 13.9.7. ISO/IEC 13.013., 13.013. y 13.013.: métricas de calidad del software y de los datos
- 13.9.8. ISO/IEC 13.040: evaluación del software
- 13.9.9. El proceso de certificación

13.10. Introducción a *DevOps*
- 13.10.1. Concepto de DevOps
- 13.10.2. Prácticas principales

Módulo 14. Ingeniería de requisitos

14.1. Introducción a la ingeniería de requisitos
- 14.1.1. La importancia de los requisitos
- 14.1.2. Concepto de requisito
- 14.1.3. Dimensiones de los requisitos
- 14.1.4. Niveles y tipos de requisitos
- 14.1.5. Características de los requisitos
- 14.1.6. La ingeniería de requisitos
- 14.1.7. El proceso de Ingeniería de Requisitos
- 14.1.8. *Frameworks* para ingeniería de requisitos
- 14.1.9. Buenas prácticas en ingeniería de requisitos
- 14.1.10. El analista de negocio

14.2. Las fuentes de los requisitos
- 14.2.1. La red de requisitos
- 14.2.2. Los *Stakeholders*
- 14.2.3. Los requisitos de negocio
- 14.2.4. Documento de visión y alcance

14.3. Técnicas de elicitación de requisitos
- 14.3.1. La elicitación de requisitos
- 14.3.2. Problemas de la elicitación de requisitos
- 14.3.3. Contextos de descubrimiento
- 14.3.4. Entrevistas
- 14.3.5. Observación y «aprendizaje»
- 14.3.6. Etnografía
- 14.3.7. *Workshops*
- 14.3.8. *Focus groups*
- 14.3.9. Cuestionarios
- 14.3.10. *Brainstorming* y técnicas creativas
- 14.3.11. Medios grupales
- 14.3.12. Análisis de interfaces del sistema
- 14.3.13. Análisis de documentos y «arqueología»
- 14.3.14. Casos de uso y escenarios
- 14.3.15. Los prototipos
- 14.3.16. La ingeniería inversa
- 14.3.17. Reutilización de requisitos
- 14.3.18. Buenas prácticas de la elicitación

14.4. Requisitos de los usuarios
- 14.4.1. Personas
- 14.4.2. Casos de uso e historias de usuario
- 14.4.3. Escenarios
- 14.4.4. Tipos de escenarios
- 14.4.5. ¿Cómo descubrir escenarios?

14.5. Técnicas de prototipado
- 14.5.1. El prototipado
- 14.5.2. Prototipos según su alcance
- 14.5.3. Prototipos según su temporalidad

14.5.4. La fidelidad de un prototipo
14.5.5. Prototipos de interfaz de usuario
14.5.6. Evaluación de prototipos
14.6. Análisis de requisitos
14.6.1. El análisis de requisitos
14.6.2. Buenas prácticas del análisis de requisitos
14.6.3. El diccionario de datos
14.6.4. Priorización de requisitos
14.7. Documentación de los requisitos
14.7.1. El documento especificación de requisitos
14.7.2. Estructura y contenidos de un SRS
14.7.3. Documentación en lenguaje natural
14.7.4. EARS: *Easy Approach to Requirements Syntax*
14.7.5. Los requisitos no funcionales
14.7.6. Atributos y plantillas en forma de tabla
14.7.7. Buenas prácticas de especificación
14.8. Validación y negociación de requisitos
14.8.1. Validación de requisitos
14.8.2. Técnicas de validación de requisitos
14.8.3. Negociación de requisitos
14.9. Modelado y gestión de requisitos
14.9.1. El modelado de requisitos
14.9.2. La perspectiva del usuario
14.9.3. La perspectiva de los datos
14.9.4. La perspectiva funcional u orientada al flujo
14.9.5. La perspectiva del comportamiento
14.9.6. La volatilidad de los requisitos
14.9.7. Proceso de gestión de requisitos
14.9.8. Herramientas para gestión de requisitos
14.9.9. Buenas prácticas en la gestión de requisitos
14.10. Sistemas críticos y especificación formal
14.10.1. Los sistemas críticos
14.10.2. Especificación dirigida por riesgos
14.10.3. Especificación formal

Módulo 15. Procesos de ingeniería del software

15.1. Marco de Ingeniería software
15.1.1. Características del software
15.1.2. Los procesos principales en Ingeniería del software
15.1.3. Modelos de proceso de desarrollo software
15.1.4. Marco de referencia estándar para el proceso de desarrollo de software: la norma ISO/IEC 12207
15.2. Proceso Unificado de desarrollo software
15.2.1. Proceso unificado
15.2.2. Dimensiones del proceso unificado
15.2.3. Proceso de desarrollo dirigido por casos de uso
15.2.4. Flujos de trabajo fundamentales de procesos unificados
15.3. Planificación en el contexto de desarrollo de software ágil
15.3.1. Características del desarrollo software ágil
15.3.2. Diferentes horizontes temporales de planificación en el desarrollo ágil
15.3.3. Marco de desarrollo ágil Scrum y horizontes temporales de planificación
15.3.4. Historias de usuario como unidad de planificación y estimación
15.3.5. Técnicas comunes para derivar una estimación
15.3.6. Escalas para interpretar las estimaciones
15.3.7. *Planning Poker*
15.3.8. Tipos de planificaciones comunes: planificación de entregas y planificación de iteración
15.4. Estilos de diseño de software distribuido y arquitecturas software orientadas a servicios
15.4.1. Modelos de comunicación en sistemas software distribuidos
15.4.2. Capa intermedia o *Middleware*
15.4.3. Patrones de arquitectura para sistemas distribuidos
15.4.4. Proceso general de diseño de servicios software
15.4.5. Aspectos de diseño de servicios software
15.4.6. Composición de servicios
15.4.7. Arquitectura de servicios web
15.4.8. Componentes de Infraestructura y SOA

15.5. Introducción al desarrollo software dirigido por modelos
- 15.5.1. El concepto de modelo
- 15.5.2. Desarrollo software dirigido por modelos
- 15.5.3. Marco de referencia de desarrollo dirigido por modelos MDA
- 15.5.4. Elementos de un modelo de transformación

15.6. Diseño de interfaces gráficas de usuario
- 15.6.1. Principios de diseño de interfaces de usuario
- 15.6.2. Patrones de diseño arquitectónico para sistemas interactivos: Modelo Vista Controlador (MVC)
- 15.6.3. Experiencia de usuario (UX *User Experience*)
- 15.6.4. Diseño centrado en el usuario
- 15.6.5. Proceso de análisis y diseño de la interfaz gráfica de usuario
- 15.6.6. Usabilidad de interfaces de usuario
- 15.6.7. Accesibilidad en interfaces de usuario

15.7. Diseño de aplicaciones web
- 15.7.1. Características de las aplicaciones web
- 15.7.2. Interfaz de usuario de una aplicación web
- 15.7.3. Diseño de navegación
- 15.7.4 Protocolo de interacción base para aplicaciones web
- 15.7.5. Estilos de arquitectura para aplicaciones web

15.8. Estrategias y técnicas de pruebas software y factores de calidad del software
- 15.8.1. Estrategias de prueba
- 15.8.2. Diseños de casos de prueba
- 15.8.3. Relación coste calidad
- 15.8.4 Modelos de calidad
- 15.8.5. Familia de normas ISO/IEC 25000 (*SQuaRE*)
- 15.8.6. Modelo de calidad de producto (ISO 2501n)
- 15.8.7. Modelos de calidad de datos (ISO 2501n)
- 15.8.8. Gestión de la calidad del software

15.9. Introducción a las métricas en Ingeniería software
- 15.9.1. Conceptos básicos: medidas, métricas e indicadores
- 15.9.2. Tipos de métricas en Ingeniería software
- 15.9.3. El proceso de medición
- 15.9.4. ISO 250215. Métricas externas y de calidad en uso
- 15.9.5. Métrica orientada a objetos

15.10. Mantenimiento y reingeniería software
- 15.10.1. Proceso de mantenimiento
- 15.10.2. Marco estándar de proceso de mantenimiento. ISO/EIEC 115.64
- 15.10.3. Modelo de proceso de reIngeniería de Software
- 15.10.4. Ingeniería inversa

Módulo 16. Integración de sistemas

16.1. Introducción a los sistemas de información en la empresa
- 16.1.1. El papel de los sistemas de información
- 16.1.2. ¿Qué es un sistema de información?
- 16.1.3. Dimensiones de los sistemas de información
- 16.1.4. Procesos de negocio y sistemas de información
- 16.1.5. El departamento de SI/TI

16.2. Oportunidades y necesidades de los sistemas de información en la empresa
- 16.2.1. Organizaciones y sistemas de información
- 16.2.2. Características de las organizaciones
- 16.2.3. Impacto de los sistemas de información en la empresa
- 16.2.4. Sistemas de información para lograr una ventaja competitiva
- 16.2.5. Uso de los sistemas en la administración y gestión de la empresa

16.3. Conceptos básicos de sistemas y tecnologías de la información
- 16.3.1. Datos, información y conocimiento
- 16.3.2. Tecnología y sistemas de información
- 16.3.3. Componentes de la tecnología
- 16.3.4. Clasificación y tipos de sistemas de información
- 16.3.5. Arquitecturas basadas en servicios y procesos de negocio
- 16.3.6. Formas de integración de sistemas

16.4. Sistemas para la gestión integrada de recursos de la empresa
- 16.4.1. Necesidades de la empresa
- 16.4.2. Un sistema de información integrado para la empresa
- 16.4.3. Adquisición vs. Desarrollo
- 16.4.4. Implantación de un ERP
- 16.4.5. Implicaciones para la dirección
- 16.4.6. Principales proveedores de ERP

16.5. Sistemas de información para la gestión de la cadena de suministro y las relaciones con clientes
- 16.5.1. Definición de cadena de suministro
- 16.5.2. Gestión efectiva de la cadena de suministro
- 16.5.3. El papel de los sistemas de información
- 16.5.4. Soluciones para la gestión de cadena de suministro
- 16.5.5. La gestión de relaciones con los clientes
- 16.5.6. El papel de los sistemas de información
- 16.5.7. Implantación de un sistema CRM
- 16.5.8. Factores críticos de éxito en la implantación de CRM
- 16.5.9. CRM, e-CRM y otras tendencias

16.6. La toma de decisiones de inversión en TIC y planificación de sistemas de información
- 16.6.1. Criterios para la decisión de inversión en TIC
- 16.6.2. Vinculación del proyecto con la gerencia y plan de negocios
- 16.6.3. Implicaciones de la dirección
- 16.6.4. Rediseño de los procesos de negocio
- 16.6.5. Decisión de metodologías de implantación desde la dirección
- 16.6.6. Necesidad de planificación de los sistemas de información
- 16.6.7. Objetivos, participantes y momentos
- 16.6.8. Estructura y desarrollo del plan de sistemas
- 16.6.9. Seguimiento y actualización

16.7. Consideraciones de seguridad en el uso de las TIC
- 16.7.1. Análisis de riesgos
- 16.7.2. La seguridad en los sistemas de información
- 16.7.3. Consejos prácticos

16.8. Viabilidad de aplicación de proyectos de TIC y aspectos financieros en proyectos de sistemas de información
- 16.8.1. Descripción y objetivos
- 16.8.2. Participantes en el EVS
- 16.8.3. Técnicas y prácticas
- 16.8.4. Estructura de costes
- 16.8.5. La proyección financiera
- 16.8.6. Presupuestos

16.9. Business Intelligence
- 16.9.1. ¿Qué es la inteligencia de negocio?
- 16.9.2. Estrategia e implantación de BI
- 16.9.3. Presente y futuro en BI

16.10. ISO/IEC 12207
- 16.10.1. ¿Qué es «ISO/IEC 12207»?
- 16.10.2. Análisis de los Sistemas de Información
- 16.10.3. Diseño del Sistema de Información
- 16.10.4. Implantación y aceptación del Sistema de Información

Módulo 17. Reutilización de software

17.1. Panorama general de la reutilización de software
- 17.1.1. ¿En qué consiste la reutilización del software?
- 17.1.2. Ventajas e inconvenientes de la reutilización de software
- 17.1.3. Principales técnicas de reutilización de software

17.2. Introducción a los patrones de diseño
- 17.2.1. ¿Qué es un patrón de diseño?
- 17.2.2. Catálogo de los principales patrones de diseño
- 17.2.3. Cómo usar patrones para resolver problemas de diseño
- 17.2.4. Cómo seleccionar el mejor patrón de diseño

17.3. Patrones de creación
- 17.3.1. Patrones de creación
- 17.3.2. Patrón *Abstract Factory*
- 17.3.3. Ejemplo de implementación del Patrón *Abstract Factory*
- 17.3.4. Patrón *Builder*
- 17.3.5. Ejemplo de implementación del *Builder*
- 17.3.6. Patrón Abstract Factory vs. *Builder*

17.4. Patrones de creación (II)
- 17.4.1. Patrón *Factory Method*
- 17.4.2. *Factory Method* vs. *Abstract Factory*
- 17.4.3. Patrón *Singleton*

17.5. Patrones estructurales
- 17.5.1. Patrones estructurales
- 17.5.2. Patrón *Adapter*
- 17.5.3. Patrón *Bridge*

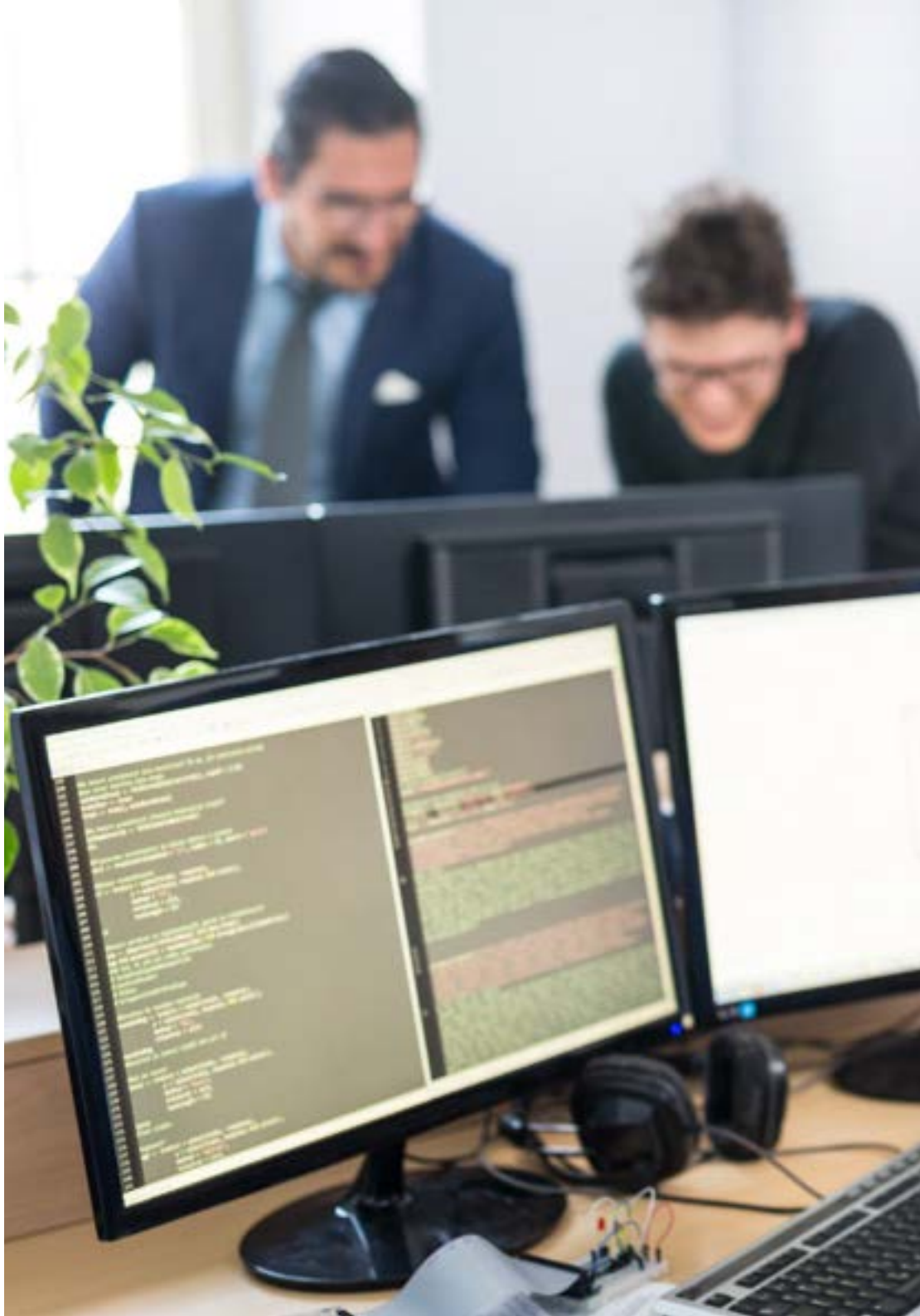
17.6. Patrones estructurales (II)
- 17.6.1. Patrón *Composite*
- 17.6.2. Patrón *Decorador*

17.7. Patrones estructurales (III)
- 17.7.1. Patrón *Facade*
- 17.7.2. Patrón *Proxy*

17.8. Patrones de comportamiento
- 17.8.1. Concepto de los patrones de comportamiento
- 17.8.2. Patrón de comportamiento: cadena de responsabilidad
- 17.8.3. Patrón de comportamiento Orden

17.9. Patrones de comportamiento (II)
- 17.9.1. Patrón Intérprete o *Interpreter*
- 17.9.2. Patrón Iterador
- 17.9.3. Patrón Observador
- 17.9.4. Patrón Estrategia

17.10. *Frameworks*
- 17.10.1. Concepto de *Framework*
- 17.10.2. Desarrollo mediante *Frameworks*
- 17.10.3. Patrón *Model View Controller*
- 17.10.4. *Framework* para diseño de interfaces gráficas de usuario
- 17.10.5. *Frameworks* para el desarrollo de aplicaciones web
- 17.10.6. *Frameworks* para la gestión de la persistencia de objetos en bases de datos



Módulo 18. Servicios de tecnología de la información

18.1. La transformación digital (I)
- 18.1.1. La innovación empresarial
- 18.1.2. La gestión de la producción
- 18.1.3. La gestión financiera

18.2. La transformación digital (II)
- 18.2.1. El marketing
- 18.2.2. La gestión de RRHH
- 18.2.3. Un sistema de información integrado

18.3. Caso de estudio
- 18.3.1. Presentación de la empresa
- 18.3.2. Metodologías para analizar la adquisición de TI
- 18.3.3. Determinación de costos, beneficios y riesgos
- 18.3.4. Evaluación económica de la inversión

18.4. El gobierno y la gestión de las TIC
- 18.4.1. Definición de gobierno de las tecnologías y sistemas de la información
- 18.4.2. Diferencia entre gobierno y gestión de las TSI
- 18.4.3. Marcos para el gobierno y la gestión de las TSI
- 18.4.4. Las normas y el gobierno y la gestión de las TSI

18.5. El gobierno corporativo de las TIC
- 18.5.1. ¿Qué es el buen gobierno corporativo?
- 18.5.2. Antecedentes de gobierno de las TIC
- 18.5.3. La Norma ISO/IEC 318.00:2008
- 18.5.4. Implementación de un buen gobierno TIC
- 18.5.5. Gobierno TIC y mejores prácticas
- 18.5.6. Gobierno corporativo. Resumen y tendencias

18.6. Objetivos de Control para la Información y Tecnologías Relacionadas (COBIT)
- 18.6.1. Marco de aplicación
- 18.6.2. Dominio: planificación y organización
- 18.6.3. Dominio: adquisición e implementación
- 18.6.4. Dominio: entrega y soporte
- 18.6.5. Dominio: supervisión y evaluación
- 18.6.6. Aplicación de la guía COBIT

18.7. La Biblioteca de Infraestructura de Tecnologías de Información (ITIL)
- 18.7.1. Introducción a ITIL
- 18.7.2. Estrategia del servicio
- 18.7.3. Diseño del servicio
- 18.7.4. Transición del servicio
- 18.7.5. Operación del servicio
- 18.7.6. Mejora del servicio

18.8. El sistema de gestión de servicios
- 18.8.1. Principios básicos de UNE-ISO/IEC 20000-1
- 18.8.2. La estructura de la serie de normas ISO/IEC 20000
- 18.8.3. Requisitos del Sistema de Gestión del Servicio (SGS)
- 18.8.4. Diseño y transición de servicios nuevos o modificados
- 18.8.5. Procesos de provisión del servicio
- 18.8.6. Grupos de procesos

18.9. El sistema de gestión de activos de software
- 18.9.1. Justificación de la necesidad
- 18.9.2. Antecedentes
- 18.9.3. Presentación de la norma 19770
- 18.9.4. Implantación de la gestión

18.10. Gestión de la continuidad del negocio
- 18.10.1. Plan de la continuidad del negocio
- 18.10.2. Implementación de un BCM

Un completísimo programa que será fundamental para tu desarrollo profesional"

05

Metodología

Este programa de capacitación ofrece una forma diferente de aprender. Nuestra metodología se desarrolla a través de un modo de aprendizaje de forma cíclica: ***el Relearning***.

Este sistema de enseñanza es utilizado, por ejemplo, en las facultades de medicina más prestigiosas del mundo y se ha considerado uno de los más eficaces por publicaciones de gran relevancia como el ***New England Journal of Medicine.***





Descubre el Relearning, un sistema que abandona el aprendizaje lineal convencional para llevarte a través de sistemas cíclicos de enseñanza: una forma de aprender que ha demostrado su enorme eficacia, especialmente en las materias que requieren memorización"

En TECH empleamos el Método del Caso

Nuestro programa ofrece un método revolucionario de desarrollo de habilidades y conocimientos. Nuestro objetivo es afianzar competencias en un contexto cambiante, competitivo y de alta exigencia.

Con TECH podrás experimentar una forma de aprender que está moviendo los cimientos de las universidades tradicionales de todo el mundo"



Somos la primera universidad online en español que combina los case studies de Harvard Business School con un sistema de aprendizaje 100% online basado en la reiteración.



El alumno aprenderá, mediante actividades colaborativas y casos reales, la resolución de situaciones complejas en entornos empresariales reales.

Un método de aprendizaje innovador y diferente

Este programa intensivo de Informática de TECH Universidad Tecnológica te prepara para afrontar todos los retos en esta área, tanto en el ámbito nacional como internacional. Tenemos el compromiso de favorecer el crecimiento personal y profesional, la mejor forma de caminar hacia el éxito, por eso, en TECH Universidad Tecnológica utilizarás los *case studies* de Harvard, con la cual tenemos un acuerdo estratégico, que nos permite acercar a nuestros alumnos los materiales de la mejor universidad del mundo.

> *Nuestro programa te prepara para afrontar nuevos retos en entornos inciertos y lograr el éxito en tu carrera"*

El método del caso ha sido el sistema de aprendizaje más utilizado por las mejores escuelas de Informática del mundo desde que éstas existen. Desarrollado en 1912 para que los estudiantes de Derecho no solo aprendiesen las leyes a base de contenidos teóricos, el método del caso consistió en presentarles situaciones complejas reales para que tomasen decisiones y emitiesen juicios de valor fundamentados sobre cómo resolverlas. En 1924 se estableció como método estándar de enseñanza en Harvard.

Ante una determinada situación, ¿qué debería hacer un profesional? Esta es la pregunta a la que te enfrentamos en el método del caso, un método de aprendizaje orientado a la acción. A lo largo del curso, los estudiantes se enfrentarán a múltiples casos reales. Deberán integrar todos sus conocimientos, investigar, argumentar y defender sus ideas y decisiones.

Relearning Methodology

Nuestra universidad es la primera en el mundo que combina los *case studies* de Harvard University con un sistema de aprendizaje 100% online basado en la reiteración, que combina elementos didácticos diferentes en cada lección.

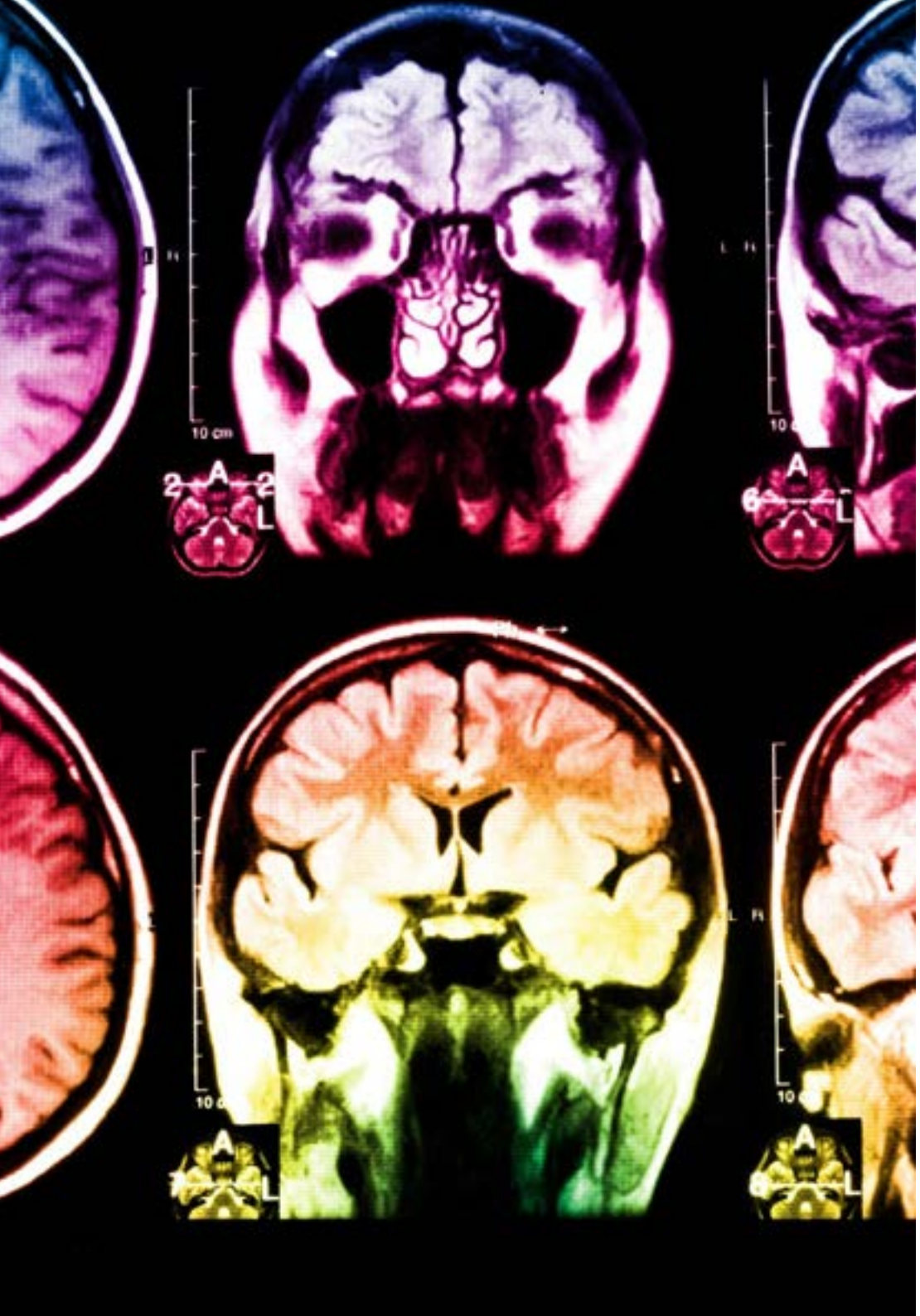
Potenciamos los *case studies* de Harvard con el mejor método de enseñanza 100% online: el Relearning.

En 2019 obtuvimos los mejores resultados de aprendizaje de todas las universidades online en español en el mundo.

En TECH aprenderás con una metodología vanguardista concebida para capacitar a los directivos del futuro. Este método, a la vanguardia pedagógica mundial, se denomina Relearning.

Nuestra universidad es la única en habla hispana licenciada para emplear este exitoso método. En 2019, conseguimos mejorar los niveles de satisfacción global de nuestros alumnos (calidad docente, calidad de los materiales, estructura del curso, objetivos…) con respecto a los indicadores de la mejor universidad online en español.





En nuestro programa, el aprendizaje no es un proceso lineal, sino que sucede en espiral (aprender, desaprender, olvidar y reaprender). Por eso, se combinan cada uno de estos elementos de forma concéntrica. Con esta metodología se han capacitado más de 650.000 graduados universitarios con un éxito sin precedentes en ámbitos tan distintos como la bioquímica, la genética, la cirugía, el derecho internacional, las habilidades directivas, las ciencias del deporte, la filosofía, el derecho, la ingeniería, el periodismo, la historia o los mercados e instrumentos financieros. Todo ello en un entorno de alta exigencia, con un alumnado universitario de un perfil socioeconómico alto y una media de edad de 43,5 años.

El Relearning te permitirá aprender con menos esfuerzo y más rendimiento, implicándote más en tu capacitación, desarrollando el espíritu crítico, la defensa de argumentos y el contraste de opiniones: una ecuación directa al éxito.

A partir de la última evidencia científica en el ámbito de la neurociencia, no solo sabemos organizar la información, las ideas, las imágenes y los recuerdos, sino que sabemos que el lugar y el contexto donde hemos aprendido algo es fundamental para que seamos capaces de recordarlo y almacenarlo en el hipocampo, para retenerlo en nuestra memoria a largo plazo.

De esta manera, y en lo que se denomina Neurocognitive context-dependent e-learning, los diferentes elementos de nuestro programa están conectados con el contexto donde el participante desarrolla su práctica profesional.

Este programa ofrece los mejores materiales educativos, preparados a conciencia para los profesionales:

Material de estudio

Todos los contenidos didácticos son creados por los especialistas que van a impartir el curso, específicamente para él, de manera que el desarrollo didáctico sea realmente específico y concreto.

Estos contenidos son aplicados después al formato audiovisual, para crear el método de trabajo online de TECH. Todo ello, con las técnicas más novedosas que ofrecen piezas de gran calidad en todos y cada uno los materiales que se ponen a disposición del alumno.

Clases magistrales

Existe evidencia científica sobre la utilidad de la observación de terceros expertos.

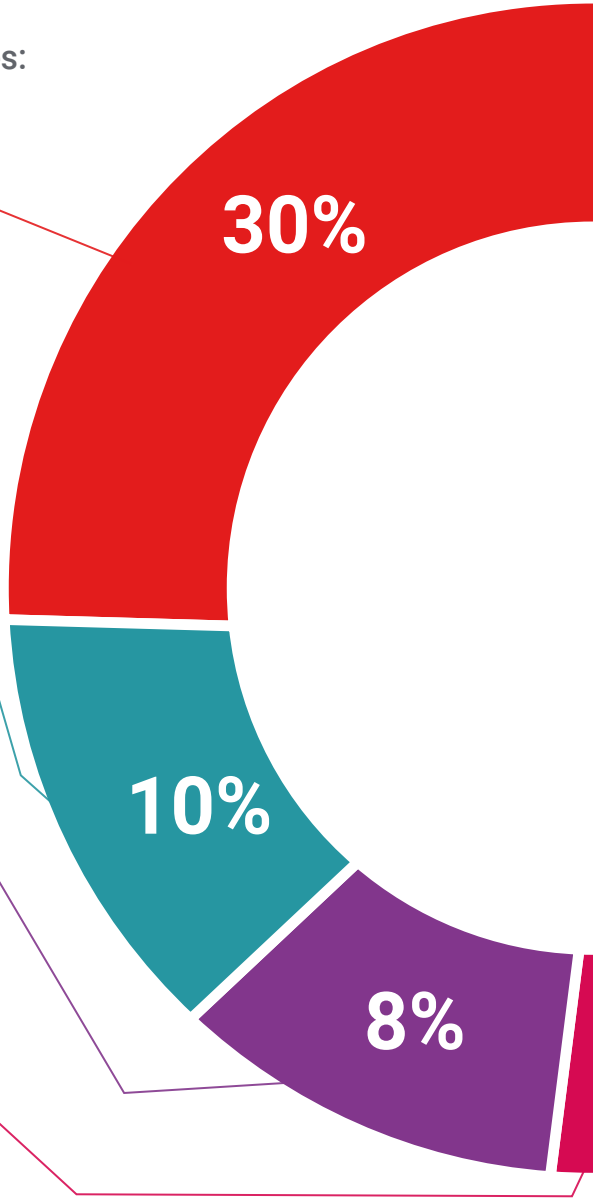
El denominado Learning from an Expert afianza el conocimiento y el recuerdo, y genera seguridad en las futuras decisiones difíciles.

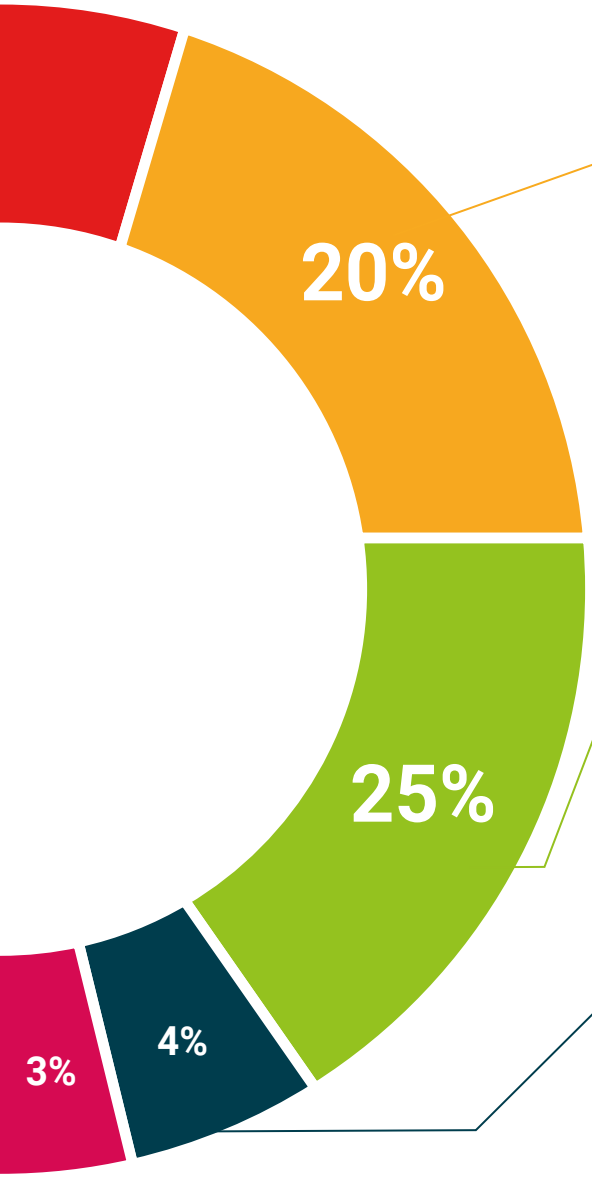
Prácticas de habilidades y competencias

Realizarán actividades de desarrollo de competencias y habilidades específicas en cada área temática. Prácticas y dinámicas para adquirir y desarrollar las destrezas y habilidades que un especialista precisa desarrollar en el marco de la globalización que vivimos.

Lecturas complementarias

Artículos recientes, documentos de consenso y guías internacionales, entre otros. En la biblioteca virtual de TECH el estudiante tendrá acceso a todo lo que necesita para completar su capacitación.





Case studies

Completarán una selección de los mejores cases studies de la materia que se emplean en Harvard. Casos presentados, analizados y tutorizados por los mejores especialistas del panorama internacional.

Resúmenes interactivos

El equipo de TECH presenta los contenidos de manera atractiva y dinámica en píldoras multimedia que incluyen audios, vídeos, imágenes, esquemas y mapas conceptuales con el fin de afianzar el conocimiento.

Este exclusivo sistema educativo para la presentación de contenidos multimedia fue premiado por Microsoft como "Caso de éxito en Europa".

Testing & Retesting

Se evalúan y reevalúan periódicamente los conocimientos del alumno a lo largo del programa, mediante actividades y ejercicios evaluativos y autoevaluativos para que, de esta manera, el estudiante compruebe cómo va consiguiendo sus metas.

06

Titulación

El Grand Master en Ingeniería de Software garantiza, además de la capacitación más rigurosa y actualizada, el acceso a un título de Grand Master expedido por TECH Universidad Tecnológica.





Supera con éxito este programa y recibe tu titulación universitaria sin desplazamientos ni farragosos trámites"

Este **Grand Master en Ingeniería de Software** contiene el programa más completo y actualizado del mercado.

Tras la superación de la evaluación, el alumno recibirá por correo postal* con cause de recibido su correspondiente título de **Grand Master** emitido por **TECH Universidad Tecnológica.**

El título expedido por **TECH Universidad Tecnlógica** expresará la calificación que haya obtenido en el Grand Master, y reunirá requisitos comúnmente exigidos por las bolsas de trabajo, oposiciones y comités evaluadores de carreras profesionales.

Título: **Grand Master en Ingeniería de Software**

N.º Horas Oficiales: **3.000 h.**


tech universidad tecnológica

Otorga la presente

CONSTANCIA

a

C. _____, con documento de identificación nº_____

Por haber superado con éxito y acreditado el programa de

GRAND MASTER

en

Ingeniería de Software

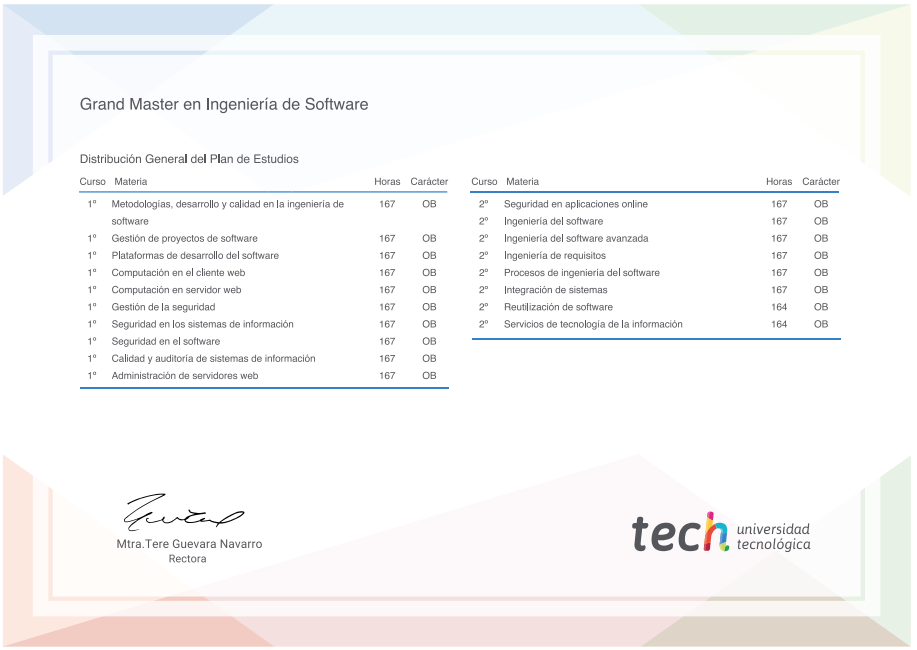
Se trata de un título propio de esta Universidad con una duración de 3.000 horas, con fecha de inicio dd/mm/aaaa y fecha de finalización dd/mm/aaaa.

TECH es una Institución Particular de Educación Superior reconocida por la Secretaría de Educación Pública a partir del 28 de junio de 2018.

A 17 de junio de 2020

Mtra.Tere Guevara Navarro
Rectora

Este título propio se deberá acompañar siempre del título universitario habilitante expedido por la autoridad competente para ejercer profesionalmente en cada país. código único TECH: AFWOR23S techtitute.com/titulos


Grand Master en Ingeniería de Software

Distribución General del Plan de Estudios

Curso	Materia	Horas	Carácter
1º	Metodologías, desarrollo y calidad en la ingeniería de software	167	OB
1º	Gestión de proyectos de software	167	OB
1º	Plataformas de desarrollo del software	167	OB
1º	Computación en el cliente web	167	OB
1º	Computación en servidor web	167	OB
1º	Gestión de la seguridad	167	OB
1º	Seguridad en los sistemas de información	167	OB
1º	Seguridad en el software	167	OB
1º	Calidad y auditoría de sistemas de información	167	OB
1º	Administración de servidores web	167	OB

Curso	Materia	Horas	Carácter
2º	Seguridad en aplicaciones online	167	OB
2º	Ingeniería del software	167	OB
2º	Ingeniería del software avanzada	167	OB
2º	Ingeniería de requisitos	167	OB
2º	Procesos de ingeniería del software	167	OB
2º	Integración de sistemas	167	OB
2º	Reutilización de software	164	OB
2º	Servicios de tecnología de la información	164	OB

Mtra.Tere Guevara Navarro
Rectora

tech universidad tecnológica

*Apostilla de La Haya. En caso de que el alumno solicite que su título en papel recabe la Apostilla de La Haya, TECH EDUCATION realizará las gestiones oportunas para su obtención, con un coste adicional.

tech universidad tecnológica

Grand Master

Ingeniería de Software

Modalidad: **Online**

Duración: **2 años**

Titulación: **TECH Universidad Tecnológica**

Horas lectivas: **3.000 h.**

Grand Master
Ingeniería de Software



tech universidad tecnológica